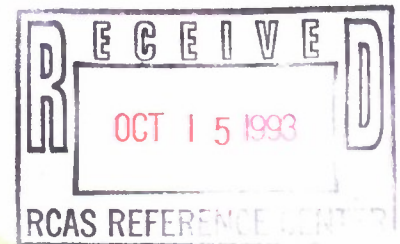


**NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY/
NATIONAL COMPUTER SECURITY CENTER**

16th NATIONAL COMPUTER SECURITY CONFERENCE

**September 20-23, 1993
Baltimore Convention Center
Baltimore, Maryland**



PROCEEDINGS

**Information Systems Security:
User Choices**

20090327404



DEFENSE TECHNICAL INFORMATION CENTER

Information for the Defense Community

DTIC® has determined on 04/10/2009 that this Technical Document has the Distribution Statement checked below. The current distribution for this document can be found in the DTIC® Technical Report Database.

☒ **DISTRIBUTION STATEMENT A.** Approved for public release; distribution is unlimited.

☐ **© COPYRIGHTED;** U.S. Government or Federal Rights License. All other rights and uses except those permitted by copyright law are reserved by the copyright owner.

☐ **DISTRIBUTION STATEMENT B.** Distribution authorized to U.S. Government agencies only (fill in reason) (date of determination). Other requests for this document shall be referred to (insert controlling DoD office)

☐ **DISTRIBUTION STATEMENT C.** Distribution authorized to U.S. Government Agencies and their contractors (fill in reason) (date of determination). Other requests for this document shall be referred to (insert controlling DoD office)

☐ **DISTRIBUTION STATEMENT D.** Distribution authorized to the Department of Defense and U.S. DoD contractors only (fill in reason) (date of determination). Other requests shall be referred to (insert controlling DoD office).

☐ **DISTRIBUTION STATEMENT E.** Distribution authorized to DoD Components only (fill in reason) (date of determination). Other requests shall be referred to (insert controlling DoD office).

☐ **DISTRIBUTION STATEMENT F.** Further dissemination only as directed by (inserting controlling DoD office) (date of determination) or higher DoD authority.

Distribution Statement F is also used when a document does not contain a distribution statement and no distribution statement can be determined.


☐ **DISTRIBUTION STATEMENT X.** Distribution authorized to U.S. Government Agencies and private individuals or enterprises eligible to obtain export-controlled technical data in accordance with DoDD 5230.25; (date of determination). DoD Controlling Office is (insert controlling DoD office).


Welcome!

The National Computer Security Center (NCSC) and the Computer Systems Laboratory (CSL) are pleased to welcome you to the Sixteenth Annual National Computer Security Conference. We believe that the Conference will stimulate a vital and dynamic exchange of information and foster an understanding of emerging technologies.

Our program this year covers a wide range of topics spanning the new draft Federal Criteria for Information Technology Security, research and development activities, techniques for building secure computer systems and networks, and ethics issues. It reflects the complex technical, economic, international, and social environment in which information system security must be developed, implemented, and practiced. Papers and panels to be presented address topics of particular concern today and for the future: the harmonization of U.S. criteria for information technology security with international criteria, present and future techniques for integrating commercial off-the-shelf products into secure systems, access control and other networking challenges, and the need for contingency planning that was highlighted so recently by the bombing of the World Trade Center.

We hope the conference presentations and these proceedings will provide you with insights and ideas that can be applied to your own efforts in information security. We recommend that you share ideas and information presented this week with your peers, your management, and your customers. Through sharing, we will help build the strong foundation of awareness, knowledge, and responsibility needed to enhance the security of our information systems and networks.


JAMES H. BURROWS
Director
Computer Systems Laboratory


PATRICK R. GALLAGHER, JR
Director
National Computer Security Center

**In Memory of
our Colleague and Friend
Howard L. Johnson**

With the passing of Howard in May 1993, the security community has lost a valuable advocate and stimulating author and debater.

Howard spent over 20 years in the security community. His aim in life was to stimulate the security community with new ideas and approaches which he considered critical to the development of sound security principles.

As President of Information Intelligence Services, he was an active participant in security workshops, presenting technical papers, teaching, and consulting.

His love of life extended beyond his sense of responsibility to his profession. He was a proud father, poet, and a very compassionate man with a desire to leave a legacy that will be remembered by his colleagues and friends.

We, who have known Howard, will miss his passion for excellence and his love of life.

Conference

Dr. Marshall Abrams
Rowland Albert
James P. Anderson
William Arbaugh
James Arnold
Alfred Arsenault
Victoria Ashby
David Balenson
James W. Birch
W. Earl Boebert
Dr. Martha Branstad
Dr. Blaine Burnham
Dr. John Campbell
Lisa Carnahan
Dr. Deborah Cooper
Karen Doty
Dr. Deborah Downs
David Ferraiolo
Ellen Flahavin
William Geer
Virgil Gibson
Dennis Gilbert
Irene Gilbert-Perry
Powell Glenn
James K. Goldston
Dr. Joshua Guttman
Dr. Grace Hammonds
Douglas Hardie
Ronda Henning
William Herndon
Dr. Harold Highland, FICS
Jack Holleran
Hilary H. Hosmer
Russell Housley
Howard Israel

The MITRE Corporation
National Security Agency
J.P. Anderson Company
Department of Defense
National Security Agency
National Security Agency
The MITRE Corporation
Trusted Information Systems, Inc.
I-NET, Inc.
Secure Computing Technology Corporation
Trusted Information Systems, Inc.
National Security Agency
National Security Agency
National Institute of Standards and Technology
Paramax Systems
CISEC
The AEROSPACE Corporation
National Institute of Standards and Technology
National Institute of Standards and Technology
Air Force Cryptologic Support Center
Grumann Data Systems
National Institute of Standards and Technology
National Institute of Standards and Technology
I-NET, Inc.
Martin Marietta
The MITRE Corporation
AGCS, Inc.
Paramax Systems
Harris Corporation
The MITRE Corporation
Computers & Security
National Security Agency
Data Security, Inc.
XEROX Information Systems
AT&T Bell Laboratories

Referees

Professor Sushil Jajodia

John Keenan

Dr. Steven Kent

Steven LaFountain

Paul A. Lambert

Dr. Carl Landwehr

Dr. Theodore M.P. Lee

Steven B. Lipner

Teresa Lunt

Frank Mayer

Barbara Mayer

Dr. Catherine Meadows

Sally Meglathery

William H. Murray

Dr. Peter Neumann

Nick Pantiuk

Donn Parker

Dr. Gopal Ramanathan

Kenneth Rowe

Professor Ravi Sandhu

Marvin Schaefer

Daniel Schnackenberg

Steven Skolochenko

Miles Smid

Dr. Stuart G. Stubblebine

Michael Thomas

Brian Tretick

Eugene Troy

Grant Wagner

Major Glenn Watt, USAF

Howard Weiss

Roy Wood

Thomas E. Zmudzinski

George Mason University

CISEC

Bolt, Barenek & Newmann

National Security Agency

Motorola GEG

Naval Research Laboratory

Trusted Information Systems, Inc.

The MITRE Corporation

SRI International

The AEROSPACE Corporation

Computer Sciences Corporation

Naval Research Laboratory

New York Stock Exchange

Deloitte & Touche

SRI International

Grumann Data Systems

SRI International

The MITRE Corporation

Department of Defense

George Mason University

CTA, Inc.

Boeing Aerospace Corporation

U.S. Postal Service

National Institute of Standards and Technology

USC Information Sciences Institute

National Security Agency

Booz•Allen & Hamilton

National Institute of Standards and Technology

National Security Agency

U.S. STRATCOM

SPARTA, Inc.

National Security Agency

Defense Information Systems Agency

Awards Ceremony

6:00 p.m. Tuesday, September 21
Convention Center, Room 317

A joint awards ceremony will be held at which the National Institute of Standards and Technology (NIST) and the National Computer Security Center (NCSC) will honor the vendors who have successfully developed products meeting the standards of the respective organizations.

The Computer Security Division at NIST provides validation services for vendors to use in testing devices for conformance to security standards defined in three Federal Information Processing Standards (FIPS): FIPS 46-1, *The Data Encryption Standard (DES)*; FIPS 113, *Computer Data Authentication*; and FIPS 171, *Key Management Using ANSI X9.17*.

Conformance to FIPS 46-1 is tested using the Monte Carlo test described in NBS Special Publication 500-20, *Validating the Correctness of Hardware Implementations of the NBS Data Encryption Standard* which requires performing eight million encryptions and four million decryptions.

Conformance to FIPS 113 and its American Standards Institute counterpart, ANSI X9.9, *Financial Institution Message Authentication (Wholesale)* is tested using an electronic bulletin board (EBB) test as specified in NBS Special Publication 500-156, *Message Authentication Code (MAC) Validation System: Requirements and Procedures*. The test consists of a series of challenges and responses in which the vendor is requested to either compute or verify a MAC using a specified randomly generated key.

Conformance to FIPS 171, which adopts ANSI X9.17, *Financial Institution Key Management (Wholesale)*, is also tested using an EBB as specified in a document entitled NIST *Key Management Validation System Point-to-Point (PTP) Requirements*.

The NCSC recognizes vendors who contribute to the availability of trusted products and thus expand the range of solutions from which customers may select to secure their data. The products are placed on the Evaluated Products List (EPL) following a successful evaluation against the *Trusted Computer Systems Evaluation Criteria* including its interpretations: *Trusted Database Interpretation*, *Trusted Network Interpretation*, and *Trusted Subsystem Interpretation*. Vendors who have completed the evaluation process will receive a formal certificate of completion from the Director, NCSC marking the addition to the EPL. In addition, vendors will receive honorable mention for being in the final stages of an evaluation as evidenced by transition into the Formal Evaluation phase or for placing a new release of a trusted product on the EPL by participation in the Ratings Maintenance Program. The success of the Trusted Product Evaluation Program is made possible by the commitment of the vendor community.

We congratulate all who have earned these awards.

16th National Computer Security Conference

Table of Contents

Refereed Papers

RESEARCH AND DEVELOPMENT, TRACK A

BApasswd: A New Proactive Password Checker	1
Chris Davies, Ravi Ganesan, Bell Atlantic	
Non-Repudiation in Open Telecooperation	16
Rüdiger Grimm, GMD Darmstadt	
NDU(C): A Mandatory Denial of Service Model	31
Edward G. Amoroso, AT&T Bell Laboratories	
Referential Integrity in Multilevel Secure Databases	39
Ravi S. Sandhu, Sushil Jajodia, George Mason University	
Query Acceleration in Multilevel Secure Database Systems	53
William Perrizo, Brajendra Panda, North Dakota State University	
Discretionary Access Control in Object-Oriented Databases: Issues and	63
Research Directions	
Roshan K. Thomas, Ravi S. Sandhu, George Mason University	
Regulating Processing Sequences via Object State	75
David L. Sherman, Daniel F. Sterne, Trusted Information Systems, Inc.	
Renewed Understanding of Access Control Policies	87
Marshall D. Abrams, The MITRE Corporation	
BSD IPC Model and Policy	97
Sandra G. Romero, Casey Schaufler, Nelson Bolyard, Silicon Graphics, Inc.	

SYSTEM IMPLEMENTATION, TRACK B

An Examination of Federal and Commercial Access Control Policy Needs	107
David F. Ferraiolo, Dennis M. Gilbert, Nickilyn Lynch, NIST	
An Open Security Architecture	117
Frederic B. Gluck, Datamedia Corporation	
Administration of Access Rights in a Multi-Vendor System--A Case History ...	129
Lee J. Becker, Defense Mapping Agency; Craig A. LaBarge, William S. Buonanni, Martin Marietta	

Refereed Papers (Cont'd)

Use of the Trusted Computer System Evaluation Criteria (TCSEC) for Complex, Evolving, Multipolicy Systems Howard L. Johnson, Information Intelligence Sciences, Inc.; Melvin L. De Vilbiss, Major (USA), National Security Agency	137
A Prototype Distributed Audit System Debra L. Banning, SPARTA, Inc.	146
Tandem Threat Scenarios: A Risk Assessment Approach Lisa M. Jaworski, Trusted Information Systems, Inc.	155
Toward a Comprehensive INFOSEC Certification Methodology Charles N. Payne, Judith N. Froscher, Carl E. Landwehr, Naval Research Laboratory	165
Modularity of Assembly-Language Implementations of Trusted Systems E. John Sebes, Terry C. Vickers-Benzel, Trusted Information Systems, Inc.	173
Security Considerations in the Design of Multi-level Secure (MLS) Database Applications Frank Kramer, Doug Nelson, Steve Heffern, James Studt, The Federated Software Group, Inc.	185
Trusted Data Distribution and Data Labeling Doug Nelson, Steve Heffern, Frank Kramer, Jim Studt, The Federated Software Group, Inc.	193
Composing Trusted Systems Using Evaluated Products Daniel Gambel, Joan Fowler, Grumman Data Systems	200
A Practical Application of Commercial-Off-The-Shelf Products to the Automated Information Systems Security of the NASA Johnson Space Center Control Center Complex James W. Coyne, Loral Space Information Systems	210
Choosing Among Standards for Lower Layer Security Protocols Wayne A. Jansen, Dale L. Walters, The National Institute of Standards and Technology	216
Message Handling Systems (X.400) Threats, Vulnerabilities, and Countermeasures Michelle J. Gosselin, The MITRE Corporation	226
Lockheed Testing Program of the Motorola Network Encryption System (NES) Joyce Capell, Kevin Gentile, Greg LaPlant, Lockheed Missiles & Space Company, Inc.	236
Certification and Accreditation Approach for the WWMCCS Guard Brian Tretick, Booz•Allen & Hamilton	245

Refereed Papers (Cont'd)

A Concept for Certification of an Army MLS Management Information System	253
Victoria P. Thompson, F. Stan Wentz, Reserve Component Automation System	
Certification and Accreditation Approach	260
Keith P. Frederick, Captain, USAF, Air Force Cryptologic Support Center (OL-FP)	
 MANAGEMENT & ADMINISTRATION, TRACK C	
Social Psychology and INFOSEC: Psycho-Social Factors in the Implementation of Information Security Policy	274
M. E. Kabay, Ph.D., National Computer Security Association	
Trusted Systems: Applying the Theory in a Commercial Firm	283
Ernest C. Charles, Donna A. Diodati, Walter J. Mozdierz, Aetna Life & Casualty	
How to Market the Information Systems Security Program	292
David Eakin, CISSP, Navy Ships Parts Control Center	
The OECD Guidelines for the Security of Information Systems: A Look to the Future	301
Christine Axsmith, Esq., ManTech Strategic Associates, Ltd.	
 CRITERIA & EVALUATION, TRACK D	
The Draft Federal Criteria and the ITSEC: Progress Towards Alignment	311
Julian Straw, Secure Information Systems, Ltd. (SISL)	
IT-Security: - a Quality Aspect! Quality Assurance in the ITSEC Evaluation Environment in Germany	324
K. Keus, W. Kurth, D. Loevenich, German Information Security Agency (GISA)	
 CLOSING PLENARY	
Seven Strategies for Information Technology Protection in the 1990s	334
Thomas R. Malarkey, Department of Defense	

Panel Summaries

RESEARCH AND DEVELOPMENT, TRACK A

Strategies for Integrating Evaluated Products	352
Dr. James G. Williams, Chair, The MITRE Corporation	
What is An Integrated System?	353
Milan S. Kuchta, Communications Security Establishment, Canada	
Integrating Specifications, Integrating Assurances	355
John McLean, Naval Research Laboratory	
Integrating Products into MLS Networks	358
W. Olin Sibert, Oxford Systems, Inc.	
The Role of Function in System Integration	359
Ruth Nelson, Information System Security Consultant	
Multipolicy System Composition	361
Hilary H. Hosmer, Data Security, Inc.	
Protection of Distributed Applications	364
Bill Shockley, Cyberscape Computer Services	
Object-Oriented Approach to the Interoperability of Trusted Database Management Systems	367
Bhavani Thuraisingham, The MITRE Corporation	
Long Term Prospects	370
Paul Boudra, National Security Agency	
Multilevel Information System Security Initiative (MISSI)	371
Gary Secrest, Chair, National Security Agency	
MISSI Overview	371
Clark Wagner, National Security Agency	
MOSAIC	372
Bill Bialick, National Security Agency	
Secure Network Server	372
Mark Roberts, National Security Agency	
Applique	372
Phil Quade, National Security Agency	
Network Security Manager	373
Robin Gerretson, National Security Agency	
MISSI Release 2	373
Lee Johnson, National Security Agency	

Panel Summaries (Cont'd)

Trusted Application Panel	374
Janet Cugini, Chair, NIST	
Trusted Database Systems: An Application for Trusted Operating Systems	375
John R. Campbell, National Security Agency	
Labeled Quadrees: Security and Geographical Information Systems	377
Mark E. Carson, IBM	
Mudumbai Ranganathan, University of Maryland	
Security Issues on Distributed System Applications	385
Chii-Ren Tsai, Citicorp International Communications, Inc.	
Security Services for Multimedia Conferencing	391
Stuart G. Stubblebine, USC Information Sciences Institute	
Best of New Security Paradigms II Workshop	396
Hilary H. Hosmer, Chair, Data Security, Inc.	
How Responsibility Modeling Leads to Security Requirements	398
Ros Strens and John Dobson, University of Newcastle upon Tyne	
Task-based Authorization: A Paradigm for Flexible and Adaptable Access Control in Distributed Applications	409
Roshan K. Thomas and Ravi S. Sandhu, George Mason University	
Identification and Authentication when Users have Multiple Accounts ..	416
W. R. Shockley, Cyberscape Computer Services	
Enterprise Security Solutions Panel	426
Paul Lambert, Chair, Motorola	
Business Issues of Information Security	427
Don Sortor, J. P. Morgan	
Securing the World's Largest Private Internet	427
Peter Browne, Motorola	
Enterprise-Wide Security with Token-Based Access Control	427
Bill Bosen, Enigma Logic, Inc.	
Secure Distributed Computing for Heterogeneous Operating System Environments	428
David Bauer, Bellcore	
 SYSTEM IMPLEMENTATION, TRACK B	
Debate of Critical Player Perspectives on MLS System Solution Acquisition Topics	429
Joel E. Sachs, Chair, Arca Systems, Inc	

Panel Summaries (Cont'd)

Network Security Management--The Harder Problem	432
Ronda Henning, Chair, Harris Corporation	
Viewpoint	433
Colonel Bill Thomas, USAF	
Viewpoint	434
W. Earl Boebert, Secure Computing Corporation	
Viewpoint	435
James Galvin, Trusted Information Systems	
Viewpoint	435
Mark J. Schertler, National Security Agency	
Viewpoint	436
Mike St. Johns, ARPA	
Application of INFOSEC Products on Wide-Area-Networks	438
Joyce Capell, Chair, Lockheed Missiles & Space Company, Inc.	
Product Developer	440
Ernie Borgoyne, Motorola Government Systems & Technology Group	
Security Technology Contributor	441
Blaine Burnham, National Security Agency	
Accreditation Support	442
Russ Mundy, Trusted Information Systems	
DSI System Integrator	443
Mark Whitney, BBN	
INFOSEC Design and Certification Initiatives: Update	444
Captain William Feteck, Chair, National Security Agency	
Viewpoint	445
Ruth Willard, National Security Agency	
Viewpoint	446
Brian M. Koehler, National Security Agency	
Security Issues for the Securities Industry	447
Sally Meglathery, Chair, New York Stock Exchange	
 MANAGEMENT AND ADMINISTRATION, TRACK C	
Virus Attacks and Counterattacks: Real-World Experiences	448
James P. Litchko, Chair, Trusted Information Systems, Inc.	
Terror at the World Trade Center	450
Sally Meglathery, Chair, New York Stock Exchange	

Panel Summaries (Cont'd)

Contingency Planning in the 90's	451
Irene Gilbert, Chair, NIST	
An Organizational Approach to Contingency Planning	452
Mark Wilson, NIST	
Business Continuity and Contingency Planning Versus Response, Resumption, Recovery, and Restoration from a Service Provider's Perspective	453
Martel Anse' Perry, Computer Technology Solutions	
Contingency Planning Guidance for Application Owners & Users	454
Sadie Pitcher, U.S. Department of Commerce	
The Evolution of Disaster Recovery Planning	455
Tom Terry, Bell Atlantic	
On a Better Understanding of Risk Management Techniques	456
Stuart W. Katzke, Chair, NIST	
The Risk Analysis Blindspot: The Threats in Security	457
Richard Baskerville, Binghamton University	
The Role of System Descriptions in Information System Risk Anslysis ...	458
Deborah J. Bodeau, The MITRE Corporation	
Methods and Issues in Risk Modeling for Computer Security	459
Ali Mosleh, University of Maryland	
Security Awareness, Training, and Professionalization: Status Report	460
Dennis Gilbert, Chair, NIST	
Needs for the Nineties: Growing Professionalization of Security Training and Security Trainers	461
Dorothea E. de Zafra, MPIA, U.S. Public Health Service	
(ISC) ²	463
Richard Koenig, International Information Systems Security Certification Consortium, Inc. (ISC) ²	
Awareness, Training, and Education in Information Technology Security: A Time for Focus and Consensus	465
Dr. William (Vic) Maconachy, Ph.D, National Security Agency	
Center for Information Systems Security (CISS)	466
Raymond Olszewski, Center for Information Systems Security	
National-Level INFOSEC Education, Training, and Awareness Initiatives	467
Joan M. Pohly, U.S. Air Force Cryptologic Support Center	
"How Much Security is Enough?" The Accreditor's Perspective	468
James P. Litchko, Chair, Trusted Information Systems, Inc.	

Panel Summaries (Cont'd)

Security and Auditability of Electronic Vote Tabulation Systems	470
Rebecca Mercuri, Chair, University of Pennsylvania	
An Integrity Model is Needed for Computerized Voting and Similar Systems	471
Roy G. Saltman, NIST	
Threats to Suffrage Security	474
Rebecca Mercuri, University of Pennsylvania	
Security Criteria for Electronic Voting	478
Peter G. Neumann, SRI International	
Security and Auditability of Electronic Vote Tabulation Systems: One Vendor's Perspective	483
Dr. Gary L. Greenhalgh, The Microvote Corporation	
Panel on Protection of Intellectual Property	490
Gerald S. Lang, Chair, Harrison Ave. Corp.	
Protection of Intellectual Property Embodied in Computer Software	491
Michael T. Platt, Dickinson, Wright, Moon, Van Dusen & Freeman	
Don't Copy That Floppy: Software Piracy is a Bigger Problem than You May Realize	492
Ilene Rosenthal, Software Publishers Association	
The Future of Intellectual Property Protection--A Smart Card Technology Solution	500
Burton G. Tregub, SPIDA, Inc.	
The Privacy Impact of Technology in the 90's	503
Wayne Madsen, Chair, Computer Sciences Corporation	
The OECD Guidelines for the Security of Information Systems: A Look to the Future (see page 301)	503
Christine Axsmith, Mantech	
Today's Technology--Tomorrow's Privacy	504
Gerard Montigny, Privacy Commission of Canada	
Privacy and Biometrics	504
John Hamlet, Deacon House	
Legal Liability of Disclosure of Private and Inaccurate Data	505
Julien Hecht, Miles and Stockbridge, Attorneys at Law	
Privacy Implications of Smart Cards	505
Juhani Saari, International Baseline Security OV, Finland	
Limits on the Use of Keystroke Monitoring for Security	505
Stewart Baker, National Security Agency	
Electronic Crime Prevention	506
Robert Lau, Chair, National Security Agency	

Panel Summaries (Cont'd)

CRITERIA & EVALUATION, TRACK D

Introduction to the Federal Criteria: User Overview and Update	507
Commander Debbie Campbell, NSA; Eugene Troy, NIST	
Federal Criteria for Information Technology Security: Protection Profile Development--A Tutorial	509
Janet Cugini, NIST; Major Mel DeVilbiss, National Security Agency	
Federal Criteria for Information Technology Security: Registry of Protection Profiles--A Tutorial on Initial Protection Profiles	511
Dave Ferraiolo, NIST; Lynne Ambuel, NSA	
Federal Criteria: Protection Profiles for Technology of the 90's Distributed Systems	513
Robert Dobry, Chair, National Security Agency	
Protection Profiles for Distributed Systems	513
Virgil Gligor, University of Maryland	
Protection Profiles for X-Window Implementations	515
Jeremy Epstein, TRW	
Protection Profiles for Secure Workstations	515
Ken Cutler, MIS Training Institute	
Vetting and Registration of Protection Profiles	516
Lynne Ambuel, Chair, National Security Agency	
International Vetting and Registration	516
Svein J. Knapskog, International Standards Organization	
The Risks and Liabilities of Vetting and Registering Protection Profiles .	517
Lawrence G. Martin, National Security Agency	
Evaluation Paradigms: An Update on the Trusted Product Evaluation Program and the Trusted Technology Assessment Program	518
Stephen Nardone, Chair, National Security Agency	
European National Evaluation Schemes	519
Ellen E. Flahavin, Chair, NIST	
The European Evaluation Process	521
Patricia Toth, Chair, NIST	

Panel Summaries (Cont'd)

International Harmonization I	522
Yvon Klein, Chair, SCSSI	
An International Comparison	522
Patricia Toth, NIST	
The Draft Federal Criteria and the ITSEC: Progress Towards Alignment	522
Julian Straw, Secure Information Systems, Ltd. (see page 311)	
IT Security:--a Quality Aspect! Quality Assurance in the ITSEC	522
Evaluation Environment in Germany	522
K. Keus, W. Kurth, D. Loevenich, German Information Security Agency (GISA) (see page 324)	
International Harmonization II	525
Eugene Troy, Chair, NIST	
Federal Criteria User Forum	528
Caralyn A. Wichers, Chair, National Security Agency	
Viewpoint	528
Charles Menk III, National Security Agency	
Viewpoint	529
W. Olin Sibert, Oxford Systems Inc.	
Viewpoint	529
Peter Callaway, IBM Corporation	
Viewpoint	530
Noelle McAuliffe, Trusted Information Systems	
Viewpoint	530
Marvin Schaefer, CTA Incorporated	
 TUTORIALS & PRESENTATIONS, TRACK E	
Tutorial Series on Trusted Systems	533
R. Kenneth Bauer, Joel Sachs, Dr. Eugene Schultz, Dr. Gary Smith, Dr. William Wilson, ARCA; Dr. Charles Abzug, LtCdr Alan Liddle, National Defense University	
Computer Virus Prevention	535
Aryeh Goretsky, McAfee Associates, Inc.	
Getting Your Work Published	536
Jack Holleran, Chair, National Security Agency	

Panel Summaries (Cont'd)

Information Systems Security Standards, The DISA Process	537
Bill Smith, Chair, CISSP, DISA	
Technical Architecture Framework for Information Management	537
John Keane, DISA/JIEO	
The Role of Standards in the DISSP Goal Security Architecture	537
Craig Sutherland, DISA/JIEO	
Some NSA Activities in the INFOSEC Standards Process	537
Dr. Dale Nunley, National Security Agency	
DISA's Information Technology Standards Management Program	537
Marilyn Kraus, DISA/JIEO	
Security Requirements for Cryptographic Modules	541
Lisa Carnahan, Chair, NIST	

Authors and Panelists Cross Index

Abrams, Marshall D.	87	Flahavin, Ellen E.	519
Abzug, Charles, Dr.	533	Fowler, Joan	200
Ambuel, Lynne	516	Frederick, Keith P. Captain, USAF	260
Amoroso, Edward G.	31	Froscher, Judith N.	165
Axsmith, Christine, Esq.	301, 503	Galvin, James	435
Baker, Stewart	505	Gambel, Daniel	200
Banning, Debra L.	146	Ganesan, Ravi	1
Baskerville, Richard	457	Gentile, Kevin	236
Bauer, David	428	Gerretson, Robin	373
Bauer, R. Kenneth	533	Gilbert, Dennis M.	107, 460
Becker, Lee J.	129	Gilbert, Irene	451
Bialick, Bill	372	Gligor, Virgil	513
Bodeau, Deborah J.	458	Gluck, Frederic B.	117
Boebert, W. Earl	434	Goretsky, Aryeh	535
Bolyard, Nelson	97	Gosselin, Michelle J.	226
Borgoyne, Ernie	440	Greenhalgh, Gary L., Dr.	483
Bosen, Bill	427	Grimm, Rüdiger	16
Boudra, Paul	370	Hamlet, John	504
Browne, Peter	427	Hecht, Julien	505
Buonanni, William S.	129	Heffern, Steve	185, 193
Burnham, Blaine	441	Henning, Ronda	432
Callaway, Peter	529	Holleran, Jack	536
Campbell, Debbie, CDR, USN ...	507	Hosmer, Hilary H.	361, 396
Campbell, John R.	375	Jajodia, Sushil	39
Capell, Joyce	236, 438	Jansen, Wayne A.	216
Carnahan, Lisa	541	Jaworski, Lisa M.	155
Carson, Mark E.	377	Johnson, Howard L.	137
Charles, Ernest C.	283	Johnson, Lee	373
Coyne, James W.	210	Kabay, M. E., Ph.D.,	274
Cugini, Janet	374, 509	Katzke, Stuart W.	456
Cutler, Ken	515	Keane, John	537
Davies, Chris	1	Keus, K.	324, 522
De Vilbiss, Melvin L. Major (USA)	137	Klein, Yvon	522
de Zafra, Dorothea E.	461	Knapkog, Svein J.	516
Diodati, Donna A.	283	Koehler, Brian M.	446
Dobry, Robert	513	Koenig, Richard	463
Dobson, John	398	Kramer, Frank	185, 193
Eakin, David	292	Kraus, Marilyn	537
Epstein, Jeremy	515	Kuchta, Milan S.	353
Ferraiolo, David F.	107, 511	Kurth, W.	324, 522
Fetech, William, Captain	444	LaBarge, Craig A.	129

Authors and Panelists Cross Index

Lambert, Paul	426	St. Johns, Mike	436
Landwehr, Carl E.	165	Saltman, Roy G.	471
Lang, Gerald S.	490	Sandhu, Ravi S.	39, 63, 409
LaPlant, Greg	236	Schaefer, Marvin	530
Lau, Robert	506	Schaufler, Casey	97
Liddle, Alan, LtCdr,	533	Schertler, Mark J.	435
Litchko, James P.	448, 468	Schultz, Eugene, Dr.	533
Loevenich, D.	324, 522	Sebes, E. John	173
Lynch, Nickilyn	107	Secrest, Gary	371
Maconachy, William (Vic), Ph.D	465	Sherman, David L.	75
Madsen, Wayne	503	Shockley, Bill	364
Malarkey, Thomas R.	334	Shockley, W. R	416
Martin, Lawrence G.	517	Sibert, W. Olin	358, 529
McAuliffe, Noelle	530	Smith, Bill	537
McLean, John	355	Smith, Gary, Dr.	533
Meglathery, Sally	447, 450	Sortor, Don	427
Menk, III, Charles	528	Sterne, Daniel F.	75
Mercuri, Rebecca	470, 474	Straw, Julian	311, 522
Montigny, Gerard	504	Strens, Ros	398
Mosleh, Ali	459	Stubblebine, Stuart G.	391
Mozdzierz, Walter J.	283	Studt, James	185, 193
Mundy, Russ	442	Sutherland, Craig	537
Nardone, Stephen	518	Terry, Tom	455
Nelson, Doug	185, 193	Thomas, Bill Colonel, USAF	433
Nelson, Ruth	359	Thomas, Roshan K.	63, 409
Neumann, Peter G.	478	Thompson, Victoria P.	253
Nunley, Dale, Dr.	537	Thuraisingham, Bhavani	367
Olszewski, Raymond	466	Toth, Patricia	521, 522
Panda, Brajendra	53	Tregub, Burton G.	500
Payne, Charles N.	165	Tretick, Brian	245
Perrizo, William	53	Troy, Eugene	525
Perry, Martel Anse'	453	Tsai, Chii-Ren	385
Pitcher, Sadie	454	Vickers-Benzel, Terry C.	173
Platt, Michael T.	491	Wagner, Clark	371
Pohly, Joan M.	467	Walters, Dale L.	216
Quade, Phil	372	Wentz, F. Stan	253
Ranganathan, Mudumbai	377	Whitney, Mark	443
Roberts, Mark	372	Wichers, Caralyn A.	528
Romero, Sandra G.	97	Willard, Ruth	445
Rosenthal, Ilene	492	Williams, James G. Dr.	352
Saari, Juhani	505	Wilson, Mark	452
Sachs, Joel E.	429, 533	Wilson, William, Dr.	533

BAsswd: A New Proactive Password Checker

Chris Davies

Christopher.I.Davies@Bell-Atl.com

Computer Security & Disaster Recovery Planning

Bell Atlantic

13101 Columbia Pike

Silver Spring, MD 20904

Ravi Ganesan

Ravi.Ganesan@Bell-Atl.com

Research & Strategic Planning

Bell Atlantic

11720 Beltsville Drive

Beltsville, MD 20705

Abstract

In our experience, poorly chosen passwords continue to be a major cause of security breaches. The increasing popularity of the UNIX operating system and the Kerberos authentication protocol in commercial environments accentuates this problem, as both are vulnerable to dictionary attacks which search for poor passwords. A proactive password checker is a component of a password changing program that attempts to validate the quality of a password chosen by the user, before the selection is finalized. In addition to checking for several attributes such as the size of the password and whether the password is derived from information about the user, the heart of any conventional proactive checker is a program that matches the password against a dictionary of passwords known to be bad. This dictionary of passwords can occupy tens of megabytes of space (in a distributed environment the dictionary may have to be replicated several times), and the time to search the dictionary can be high, especially if an attempt is made to filter out bad noisy passwords (which are of the form: common words plus one character noise, e.g. tiger2 or compQuter).

BAsswd is a new proactive password checker which drastically reduces the space and time requirements of the matching program. This is achieved by applying the theory of statistical inference on Markov chains to the "bad password recognition" problem. We assume that bad passwords are a language generated by a k th order Markov process, and then estimate the transition probabilities of this process from existing dictionaries of bad passwords. This table of transition probabilities, which takes up very little space, is then used in lieu of the dictionary itself. When given a password, BAsswd will use statistical tests to determine, with a high degree of confidence, whether that password could have been generated by the same Markov process, and if so, rejects the password. A key feature of BAsswd is that bad noisy passwords are automatically recognized as being unsuitable and need not be present in the initial training dictionary.

We present considerable empirical evidence to show that BAsswd successfully filters out bad passwords, while simultaneously ensuring that it does not become very burdensome for a legitimate user to choose a new password.

Keywords: Cryptography, Dictionary Attacks, Markov chains, Passwords, Proactive Password Checkers, Pronounceable Password Generators, Statistical inference.

1.0 Why proactive password checking?

Given the choice, most users choose passwords from a “likely password” password-space, κ_l , that is a small fraction of the entire password-space, κ , available to them. This smaller key¹ space is typically composed of words from natural languages, jargon, acronyms, dates and derivatives of these words. The small size of κ_l implies that attacks based on exhaustive search of the password-space become practical. For instance, in the UNIX operating system [11,18], user passwords are transformed using a one way function based on DES, and then stored in a password file that is usually publicly available, and is in all cases available to system administrators. As the one way function itself is not secret, an adversary can methodically apply this function² to all words in κ_l , and then compare the results to those in the password file. The Kerberos authentication protocol [17] is also vulnerable [3] to such a dictionary attack as, for reasons not relevant here, the protocol makes it possible for any adversary to request from the server, a ticket-granting-ticket encrypted with any user's password. The adversary can also obtain additional messages encrypted with user passwords by eavesdropping on the network. The adversary can decrypt the ticket or the messages using exhaustive key search over κ_l , stopping when the expected redundancy is discovered.

The size of the password-space that can be searched efficiently by an adversary, is much larger than is usually believed; the interested reader is referred to Karn and Feldmeier [15] for a discussion on the size that can be searched using current technology. Although their comments are directed towards UNIX password security, the results are widely applicable to most systems where the key space is artificially small, and where a chosen plaintext attack is feasible. Protecting a security system against such dictionary attacks, requires either altering the system itself (for instance Bellare and Merkle's Encrypted Key Exchange [2] approach to securing Kerberos), or enlarging the size of the likely password-space κ_l until it approaches the size of κ , which should naturally be chosen to be very huge. One method of achieving this would be to have the system select a random password from κ for the user. This is decidedly user unfriendly and may lead to problems such as passwords being written down. However, if such a policy can be enforced and policed, then it is, from a security perspective, an optimal approach³. A related approach is to have the system generate random, but pronounceable passwords [1,12]. We briefly discuss how one such scheme (not the one described in [1,12]) can be broken, but, our major objection to such schemes is our conjecture that a *user chosen* password is always more likely to be remembered and less likely to be written down or forgotten (note: we do not have any scientific evidence to support this conjecture).

Proactive password checkers are based on the philosophy that, with sufficient guidance from the system, users can select passwords from a fairly large key space, which are not likely to be “guessed” in the course of a dictionary attack. Such a program could interact with the user, explain the sort of passwords that are desirable, check for the appropriate size, the appropriate mix of lower case, upper case and special characters, check if the password is drawn from the user's name, login-name, etc., and finally check if the password belongs to a dictionary of passwords that are known to be bad. Note that a good proactive password checker will detect both words like *tiger* which is a bad password and words like *tiger2* or *compQuter* which we term bad noisy passwords (informally we define *bad noisy passwords* to be passwords obtained by adding one character of noise to a *bad password*).

1. We use the words key space and password space synonymously.

2. If databases of encrypted passwords are generated in advance, the “salting” introduced [7], multiplies the size of the password space that needs to be checked by a constant factor (4096)

3. This scheme will work particularly well in environments where security administrators can shoot users caught violating security policy. However, in environments, such as ours, where an unfriendly scheme would result in our *users* shooting the *security administrators*, we recommend proactive password checkers!

For a more comprehensive review of proactive password checkers we recommend the interested reader to Bishop's excellent survey[5]. Spafford's description of the OPUS project [21] is another useful reference, where the author eloquently argues the merits of proactive password checking.

The rest of this paper is structured as follows: In Section 2 we describe the motivation for the development of BApasswd. In Section 3 we review two related approaches and also briefly discuss some problems we identified with one pronounceable password scheme. In Section 4 we describe BApasswd. In Section 5 we empirically examine the performance of BApasswd and contrast it to some other schemes. In Section 6 we conclude.

2.0 Why BApasswd?

The single most important, and most difficult, function of a proactive password checker is verifying if the password chosen belongs to a dictionary of passwords known to be bad. The "obvious" way of performing this task, namely comparing the password to an actual dictionary suffers from three drawbacks:

1. **Space:** The size of a "good dictionary of bad passwords" can be several Megabytes (Spafford [21] reports a size of 25 MB for a dictionary compiled at Purdue). While this may be acceptable in a centralized environment, replicating¹ the dictionary in a distributed environment with thousands of workstations and servers is unacceptable. In addition to the size problems, such dictionaries need to be protected from unauthorized access to prevent the dictionary itself from being stolen and used in a dictionary attack against another site not protected by a proactive password checker.
2. **Time:** The time required to search a large dictionary may not in itself be very high. However, in order to capture bad noisy passwords (i.e. bad passwords plus one character of noise), it becomes necessary to incorporate further matching algorithms which may be time consuming. We note that a proactive password checker works in real time, while the user waits.
3. **Bad Noisy Passwords:** As noted above, a dictionary search does not easily capture bad noisy passwords, and unless significantly augmented, may well allow such passwords to be picked by users.

BApasswd is designed to address all three of these problems. It is designed to use an insignificant amount of data storage, be extremely fast² and successfully filter out bad noisy passwords. Before describing our design we very briefly review two related schemes.

3.0 Related work

We are aware of two other proactive password checkers which have been designed with the goal of saving on the storage space for the dictionary. Both are similar to BApasswd in that they follow the traditional pattern matching framework [9], which for our problem is:

1. Extract a set of characteristics, c , from given bad password dictionaries in an off-line mode. Key to saving space is that c should be much smaller than the dictionary itself.
2. In the on-line mode, use test, T , to determine if a given password has characteristics similar to c .

The differences in the schemes are in the characteristics, c , extracted, and consequently the test, T , used to make the determination in the second step. We now describe the two schemes.

Let $A = \{a, b, \dots, z, SPC, OTHER\}$ be the set of 28 characters which comprise the alphabet from which passwords are constructed. Nagle [19] describes an "Obvious Password Checker", in

1. An alternative to replicating the dictionary would be for the user to interact remotely with a centralized site where the dictionary is maintained. Unless a Kerberos-like system is used, securing the protocol for achieving this becomes a major problem.
2. The on-line portion of BApasswd runs in time and space constant in the size of the dictionary.

which c is a three dimensional boolean matrix, $M[i, j, k]$, where i, j and k correspond to the indices into the set A . In the training mode the bad password dictionary is scanned, and every sequence of three consecutive characters (henceforth called trigrams) that is observed, results in the corresponding bit in the boolean array being set. For instance, the password *abcd1* will cause $(M[a, b, c], M[b, c, d])$ and $M[c, d, OTHER]$ to be set to '1'. By scanning all the passwords in the dictionary, many such bits will be set. In the on-line phase all trigrams from the password are extracted, and the password is accepted as a good password, only if there are at least two trigrams which do not have their corresponding bits set in M . As we shall see later, this simple algorithm does an excellent job of screening most bad passwords. However, it does not do a very good job in keeping out bad noisy passwords. BApaswd is distantly related to this test (henceforth called the Nagle test), but does not have the vulnerability to bad noisy passwords. We present an empirical comparison in Section 5.

Spafford's OPUS [21] test is based on Bloom filters [6] which have found use previously in spelling checkers. Let $B[N]$ be a boolean array of size N . Let H_1, H_2, \dots, H_d be a set of d hash functions. Given a password, each hash function returns a number in the range $0 \dots N$. In the training phase each password in the dictionary is run through all d functions, and for each of the d numbers, n_i , generated, the bit in $B[n_i]$, is set. In the on-line testing phase, the password is run through the d functions, generating n_1, n_2, \dots, n_d . If any of $B[n_1], B[n_2], \dots, B[n_d]$, are not set, then the password is deemed to be suitable. If all corresponding bits in B are set, then, with a high probability, the password was present in the training dictionary. There is a small probability that words not in the initial dictionary, which may be good passwords, may be mistakenly identified (false positives) as being bad passwords. By increasing the size, N , in the array B , this probability can be made negligible. Also the choice of hash functions is extremely important.

The OPUS work is still in progress and consequently we cannot present an empirical comparison with BApaswd, as we are not aware of which hash functions and other parameters are recommended. Qualitatively we see the following differences:

1. Unlike BApaswd, OPUS (as described in [21]) will not be able to filter out noisy bad passwords. However, it is our understanding [22] that extensions to OPUS which deal with this situation are under development.
2. As observed earlier, OPUS requires $B[N]$ to be large enough to ensure that the number of false positives is low. Taking this into account, the author reports that $B[N]$ can be made between twelve to fifteen times smaller than the training dictionary it replaces. BApaswd on the other hand requires constant storage of about 175KB.
3. OPUS is guaranteed, by definition, to successfully recognize every bad password it "saw" during training. BApaswd is statistical in nature and there will be a small number of such passwords not recognized. This issue in BApaswd, is easily fixed, by augmenting the on-line search to carry out a regular dictionary search on those words from the training dictionary which it does not recognize as bad passwords in the on-line mode. This of course requires additional space.

Empirical comparisons are needed to contrast OPUS and BApaswd. We reiterate that we survey OPUS and the Nagle password checkers, and not any others, because our focus is on systems that eliminate the need for large on-line dictionaries.

An alternate method of solving the poor password problem is to use system generated passwords. As mentioned earlier we consider these to be extremely user-unfriendly. A slightly less unfriendly variant is to use 'pronounceable password schemes', such as those described in [1,12]. Our major objection to such schemes, for which we have no scientific evidence, is our belief that a machine chosen password is far more likely to be written down or forgotten. Given that a proactive password checker like BApaswd can allow the user to pick their own password, and yet ensure the password space is large, we see no reason to impose a machine generated password on the user. The size of the password space of a pronounceable password

checker can be deceptive. In one such scheme (not the scheme described in [1,12]), passwords are generated from 25 templates, where a template represents a pronounceable combination of characters (e.g. consonant-vowel-consonant may be a valid template for generating a three character password). Let the number of different passwords generated by each of the 25 templates be T_1, T_2, \dots, T_{25} . To generate a password the system indexes randomly into one of the templates and generates a password. Assume that this system was used on a UNIX server which has 100 users. Let us also assume that a hacker has a copy of the file (/etc/passwd) containing the encrypted passwords. If the hacker wants to try to guess a particular user's password, then she can expect to search

through half the total password space T , where $T = \sum_{i=1}^{25} T_i$. Since T would have been cho-

sen to be large, it appears that our hacker has been thwarted. However, the typical hacker is interested into breaking into the "system" and as a first step would like to compromise *any* user account. Now her task is easier: Let $T_{min} = \text{Min}(T_1, T_2, \dots, T_n)$. The hacker can successfully guess that one in every twenty-five users (i.e. about 4 of the 100 users), would have picked a password generated from T_{min} . Consequently the hacker will only have to exhaustively search T_{min} , which can be much smaller than T . In one such system we examined, it would have been fairly trivial for a hacker with limited computing resources to break the system.

4.0 The BApaswd design

BApaswd is a full fledged proactive password checker that can be used as a component of any password changing program. We first describe five major design requirements, and then concentrate on the technical details of how we meet two of these requirements.

4.1 Major design requirements

The five important requirements that led to the BApaswd design are:

1. The user should not be allowed to select a password that is based on user information commonly available on-line (e.g. user name), should be of an adequate length, should have the appropriate mix of upper case, lower case and special characters, etc. Other related goals are to enforce password aging and ensure that a history of passwords is maintained to discourage reuse. In meeting some of these goals BApaswd is very similar to Hoover's *npaswd* [14]. Also see Bishop [5] for a list of requirements, most of which, when complete, BApaswd will meet.
2. The user should not be able to select a password that is known to belong to a dictionary of bad passwords, or bad noisy passwords, which we defined as bad passwords to which one character of noise is added.
3. The proactive password checker should not require storing large dictionary files. We expect BApaswd to be installed on literally thousands of workstations, servers, mini-computers and mainframes in our environment, and requiring replication of a large dictionary on even a fraction of these computers is unacceptable.
4. The code and accompanying data files for the checker should be small, flexible and portable, as we expect to incorporate it into the password changing programs of several operating systems and security services (e.g. UNIX, ACF2). We have already incorporated BApaswd into *kpaswd*, the password changing program of Kerberos V [17].
5. BApaswd must be user friendly. Our environment is a typical commercial environment composed of users, who for the most part, are responsible corporate citizens, and will accept minor inconveniences as a price for improving security. However, we are not a top secret defense establishment, and any security sys-

tem that is difficult to use will not be tolerated. Consequently, BApaswd is designed to be a friendly, conversational, interactive system. Explaining why a given password was regarded as bad, and providing the criterion¹ for selecting a good password is important. Further, the entire transaction should not take more than one or two minutes. We expect that as users get used to the system, the time to select a password will become much smaller. Also, ensuring that good passwords are selected reduces (though not eliminates) the need to force the user to change passwords very often.

This paper does not address design issues related to requirements 1, 4 and 5 above. Rather, for the rest of the paper we focus exclusively on the way we meet requirements 2 and 3. We note, however, that our code is extremely compact (a few hundred lines of C).

4.2 The BApaswd approach to the bad password recognition problem

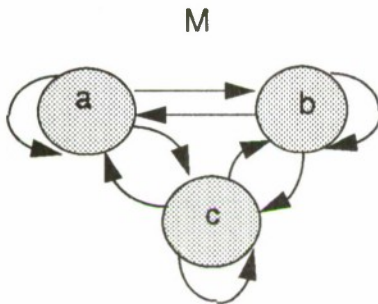
We first provide some background on Markov models, then explain our choice of parameters for the model and show how to extract the characteristics c in the off-line stage and finally present the test, T , used in the on-line password recognition stage.

4.2.1 Markov models: Background

As explained in Section 3, the traditional pattern matching framework, in our context is to:

1. Off-line: Extract a set of characteristics, c , from given bad password dictionaries.
2. On-line: Use test, T , to determine if a given password has characteristics similar to c .

In BApaswd, we assume that the bad password dictionary was generated by a k th order Markov model, and the characteristics, c , correspond to the transition probabilities of the model. Determining whether a given string was generated by a given Markov model, is a well studied problem in statistical inference on Markov chains, and, our test, T , is drawn from this literature. In earlier papers, co-authored by the second author of this paper, a guide [9] and empirical study [10] is provided to the problem of language recognition using Markov chains, with an emphasis on applications in cryptanalysis. This paper is self-contained and we refer the interested reader to [9,10], for more information. To the best of our knowledge, BApaswd represents the first time that this theory has been applied to the problem of proactive password checkers.



$M = \{3, \{a, b, c\}, T, 1\}$, where

$$T = \begin{bmatrix} 0.0 & 0.5 & 0.5 \\ 0.2 & 0.4 & 0.4 \\ 1.0 & 0.0 & 0.0 \end{bmatrix}, \text{ i.e. } T[a, a] = 0.0, T[a, b] = 0.5, \text{ etc.}$$

e.g. string probably from this language: *abbcacaba*.

e.g. string probably *not* from this language: *aaccccbaaa*.

Fig-1: An example Markov model. Likely strings can be generated by beginning in any state, and following high probability transitions. Observe that the string unlikely to have been generated by this model contains many zero transitions, e.g. *cc* and *aa*.

A Markov model M is a quadruple, $\{m, A, T, k\}$, where m is the number of states in the model, A is the state space, T is the matrix of transition probabilities and k is the order of the chain. In Fig-1 an example of such a model for a three character language is shown.

1. Care is taken to not over specify the constraints, which could result in artificially reducing the size of the keyspace of the passwords.

A key characteristic of a k th order Markov model, is that the probability of making transition $T[x, y]$, depends only on the previous k states that have been visited. In a 1st order model the probability of a transition ending in state y , depends only on the state from which the transition began (say x). i.e. $T[x, y] = \text{Prob}(y|x)$. In a 2nd order model, the probability of entering state y from state x , also depends on the state the process was in prior to entering x , say w . i.e. $T[x, y] = \text{Prob}(y|wx)$.

4.2.2 A Markov model of a bad password dictionary

As the example above illustrates, for our purposes, the state space very naturally corresponds to the alphabet of the natural language from which we expect passwords to be drawn. **B**apasswd uses a state space of size, $m = 28$, where $A = \{a, b, \dots, z, \text{SPC}, \text{OTHER}\}$. We do not differentiate between lower and upper case alphabets, and the remaining forty to fifty numbers, special characters and control characters are mapped into the OTHER category which is then treated like any other character.

We experimented with both a first order model as well as a second order model. In this paper we report on the second order model which gives better overall performance. Observe that the size of the transition matrix that has to be stored increases with increasing order: for a 1st order model the matrix occupies about 5-6 KB, while for a 2nd order model it will occupy about 175K.

Having specified m , A and k , it only remains to be seen how the probabilities in the transition probability matrix T are estimated. The first step is to select a fairly large file of known bad passwords. As discussed in the next section we experimented with several such dictionaries, and found that a medium sized dictionary (about 1MB) proved adequate. The dictionary, henceforth called D , is described in the next Section. We are interested in estimating the transition probabilities $T[i, j, k]$, which is the probability of a transition from the j th state to the k th state, given that the process reached the j th state from the i th state. The steps in calculating T are:

1. From D , we first calculated the frequency matrix f , where $f[i, j, k]$ is the number of occurrences of the trigram consisting of the i th, j th and k th characters. For instance, the password, *parsnips*, yields the trigrams *par*, *ars*, *rsn*, *sni*, *nip* and *ips*.
2. For each bigram ij calculate $f(i, j, \infty)$, as the number of trigrams beginning with ij . So $f(a, b, \infty)$ would be the number of trigrams of the form *aba*, *abb*, *abc*, ..., *abSPC*, *abOTHER*.
3. We could then calculate T as:

$T[i, j, k] = \frac{f(i, j, k)}{f(i, j, \infty)}$. This method of calculating transition probabilities is known[4] to be a maximum likelihood estimate

of the transition probabilities.

When we experimented with bigrams (1st order model) this proved adequate. However, when we shifted to a 2nd order, trigram model (to obtain greater accuracy) we found that the performance of the system was seriously effected because the trigram transition probability matrix contained too many zeroes and we had to use an alternate method. The problem of zeroes in the transition matrix, is well understood in the statistical literature, and in [9] we had reported some methods of dealing with this situation. However, for this work we relied on a different method of adjusting our transition probability matrix to deal with zeroes. Namely, we used the well known Good-Turing [13] method of adjusting the frequencies. In this method, after computing the frequencies in Step 2 above, the following steps are performed:

1. A *trigram* is any three consecutive characters. Similarly, a *bigram* is any two consecutive characters and a *unigram* is a single character.

4. Calculate values for array R , where $R[i]$ contains the number of times the frequency i occurs in f . For instance if f contains 500 zero elements, then $R[0] = 500$. It is recommended that the distribution of R be 'smoothed', however, we did not find the necessity to do so.
5. As per Katz's recommendation [16], if $f[i, j, k] = 1$, then perform $f[i, j, k] \leftarrow 0$.
6. Adjust the frequency matrix f , using the Good-Turing method [13] as follows:

$$f[i, j, k] \leftarrow \frac{(f[i, j, k] + 1) \times R[f[i, j, k] + 1]}{R[f[i, j, k]]}$$

Note that as per Katz's recommendation[16], this 'adjustment' is only performed when $f[i, j, k] \leq 5$.

7. Calculate the T matrix from the adjusted f matrix as described in Step 2.

Finally, we note that Church and Gale [7] describe yet another method of adjusting the frequencies. They first define a maximum likelihood estimate for the probability of a given trigram as $T[i, j, k] = \frac{f(i, j, k)}{Total}$ where $Total$ is the sum of all frequencies in f .

Note that this is different from the maximum likelihood estimate described in Step 2 above, which is the probability of a trigram occurring, given that the first two characters have been observed. They show that their enhanced method works better than the maximum likelihood estimate they defined. While possible, it is not obvious from their work that the enhanced method they describe will outperform the maximum likelihood estimate described above in Step 2. From a practical perspective the method we used performs adequately.

4.2.3 Tests for bad passwords

Having completely parameterized our Markov model, $M = \{28, \{a, b, \dots, z, SPC, OTHER\}, T, 1\}$, we have completed the first step of extracting the characteristics C , from the dictionary. We now turn our attention to showing how it can be determined if a given password, P , has characteristics similar to our C .

By modeling the dictionary as a Markov model, we have reduced the "Is this a bad password?" question to "Was this string (the password) generated by this Markov model (the model of the dictionary)?" This reduction allows us to draw from the wealth of literature on statistical inference on Markov chains. This theory is also very useful in an unrelated application (language recognition in cryptanalysis) which motivated one of us to co-author a guide[9] and an empirical study [10] of the discipline. We refer the interested readers to these papers, and to their references, for further reading. In this paper we restrict ourselves to a description of one of the many tests which BApaswd's design permits use of. All these tests, use the transition probability matrix, T , and the candidate password P , as their sole inputs. While these tests have been used before in different cryptologic applications, to the best of our knowledge, the design of BApaswd is the first time that they have been applied to the bad password recognition problem.

The test we use here is a log-likelihood function and is a standard statistical test¹ for determining whether a given string belongs to a particular Markov chain. Let the password P , be

depicted by $p_1 p_2 \dots p_l$, where l is the length of the password. Given a particular transition probability matrix, T , and a password P , the log likelihood function llf , is given by:

$$llf = \sum_{i=1}^{l-2} \ln(T[p_i p_{i+1} p_{i+2}])$$

For instance, for the password *unknown2*, llf is given by:

$$llf = \ln(T[u, n, k]) + \ln(T[n, k, w]) + \ln(T[k, w, o]) + \ln(T[w, o, n]) + \ln(T[o, n, OTHER])$$

Observe that the OTHER character is treated like any other character, the difference being that it is actually an equivalence class for any character which is not present in A..Z and SPC. Also note that since the transition probabilities are by definition less than one, and since we are summing the natural logs of the transition probabilities, llf will always be negative or zero.

In order to transform llf to the final test we actually use, we carry out standard statistical techniques of scaling, centering and normalizing, giving the final test, we call BAP , as:

$$BAP = \frac{\frac{llf}{l-2} - \mu}{\sigma}$$

where $l-1$ is the number of trigrams and, μ and σ , are the estimated mean and standard deviation of $\frac{llf}{l-2}$. The estimated mean and standard deviation are calculated by computing the value $\frac{llf}{l-2}$, for every password in the bad password dictionary, D (from which T , the transition probability matrix was calculated) and then calculating the mean and standard deviation of the resulting values using standard formulas. Note that the method for scaling, centering and normalizing that we use can be found in any statistics textbook. In [10] other techniques are discussed.

Finally, we note that due to the centering and normalizing, BAP has, by definition, a mean of zero and a standard deviation of one. It is now possible to set a threshold (we chose 2.6 standard deviations, which corresponds to about 99% of the area under the normal curve), and accept as a good password any password that has a value of less than -2.6. Passwords close to the mean, zero, are viewed as being drawn from the bad password dictionary, and are hence unacceptable. Due to space considerations, we shall not describe any other tests. We would like to note that we have experimented with another test based on unigram positional frequencies, and were not impressed with the results.

4.2.4 Time/Space Efficiency of Test

The test described above happens in real time and hence must work real fast, and preferably should not take much space. *BAPasswd* takes as input the transition probability file and a configuration file containing the mean, μ , the standard deviation, σ , and the threshold. All this data is computed in an off-line phase. The on-line test computes llf and then BAP which require minimal computation. Observe that the natural log function need not be computed on-line since instead of storing the transition probabilities, $T[i, j, k]$, it is possible to store $\ln(T[i, j, k])$. The space taken by *BAPasswd* is mainly for storing the transition probability file, which, for the 2nd order model, is about 175KB, which we consider practically negligible.

1. In the cryptologic literature, Sinkov [20] uses this test in the context of comparing two candidate solutions during the cryptanalysis of Vigenere ciphers [8]. However, his application did not require him to perform the standard statistical procedures of linear scaling, centering and normalizing which we do.

5.0 BApasswd: An empirical evaluation

In this section we first illustrate BApasswd's performance in keeping out bad passwords and bad noisy passwords, next we show how it performs when presented with good passwords and finally we compare it with the Nagle test we described in Section 3. Our tests were conducted on the password files BP1-BP6 (six files of bad passwords), NBP1-NBP3 (three files of bad noisy passwords), GP1-GP3 (three files of good passwords) and on UP1 (a file of machine generated pronounceable passwords). Descriptions of these files are summarized in Fig-2. We first trained (i.e. calculated τ , μ and σ) BApasswd on BP5 and then tested it on all these files. We depict our results both using histograms and a summary table of the percentage of passwords accepted. Each histogram has a vertical line at the value -2.6, and passwords to the left of this line are accepted as being good passwords, and those to the right are rejected.

5.1 How well does BApasswd keep out bad passwords?

In Fig-3 six histograms illustrate the performance of BApasswd on files containing 'bad' passwords. As can be seen, BApasswd works extremely well in recognizing bad passwords. The only file requiring explanation is BP6 which does have 12% of the passwords classified as 'good'. This happens because many of the ostensibly bad passwords in BP6, are indeed quite good, and are unlikely to be present in any hacker's dictionary. This file also contains a German technical dictionary, which though it contains many English terms, has several words that can be considered good passwords. Finally, it should be noted that the histograms have different scales as the files range in sizes from BP2 which has 280 jargon words to BP6 which has 961,947 words.

5.2 BApasswd and bad "noisy" passwords

We chose NBP1-NBP3 as our baseline for bad noisy passwords. As shown in Fig-5, BApasswd very successfully keeps out bad "noisy" passwords. As shown in Section 5.4, this is a significant advantage of BApasswd over the Nagle test. The OPUS test, unless augmented to recognize noise, will by definition be unable to detect noisy passwords.

5.3 Does BApasswd keep *in* good passwords?

BApasswd is statistical in nature and some good passwords will be mistakenly classified as bad. If the percentage of such passwords is large, then BApasswd will become very user unfriendly, and will be practically useless. Fortunately, our experiments show that users can select acceptable passwords using BApasswd. These good passwords may well look random in nature and the question of comparing it with systems that generate random user passwords arises. The key difference, which makes all the difference to usability, is that the good random passwords that are obtained using BApasswd are *chosen by the user*, and, presumably, have some semantics that aid memory. Note that the other portions of BApasswd, not described here, will guide the user into inserting some special characters in the password, which results in increasing randomness. To measure this quantitatively we conducted experiments on GP1 (keyspace of 95 random characters), GP2 (keyspace=A.Z + five special characters) and GP3 (keyspace = A.Z) all three of which contain "good" random passwords. As can be seen in Fig-4 below, BApasswd correctly classifies a high portion of all three password files correctly.

In addition, we also conducted some usability tests by asking users to select passwords using our system. Our methodology for these trials were somewhat ad hoc and consequently we choose not to report detailed quantitative results. Our general observations were that:

- Users rapidly adapt to and "get the hang of" selecting passwords that will be accepted by BApasswd. Our unscientific sampling on a small set of users showed that it did not take more than three tries to select a good password.

File	# Passwords	Description
BP1	586	A file of bad passwords available with the Crack[7] package
BP2	280	Also from Crack, but contains many jargon words
BP3	18780	The standard UNIX dictionary distributed with SunOS 4.1.2
BP4	80698	Includes BP3, but also has first & last names, and slang words
BP5	86536	All passwords from BP1 - BP4 with duplicates removed
BP6	961947	BP4 + a German technical dictionary + all words in 3 years of netnews (including several misspellings) + 2 Webster's and one Collins dictionary. Special characters and duplicates removed.
GP1	100000	Random 8 character passwords generated from the chars. A - Z
GP2	100000	Random 8 character passwords generated from A - Z + 5 special chars
GP3	119916	Random 8 character passwords generated from all printable characters (95 characters)
NBP1	18780	UNIX Dictionary, except that one special character is stuffed into every password at a random location
NBP2	18780	Same as NBP1, except the random characters aren't stuffed into the first or the last positions
NBP3	80698	BP4 with a special character stuffed in a random location
UP1	69630	Produced from a program that generates random pronounceable passwords

Fig. 2: Description of password files used in experiments.

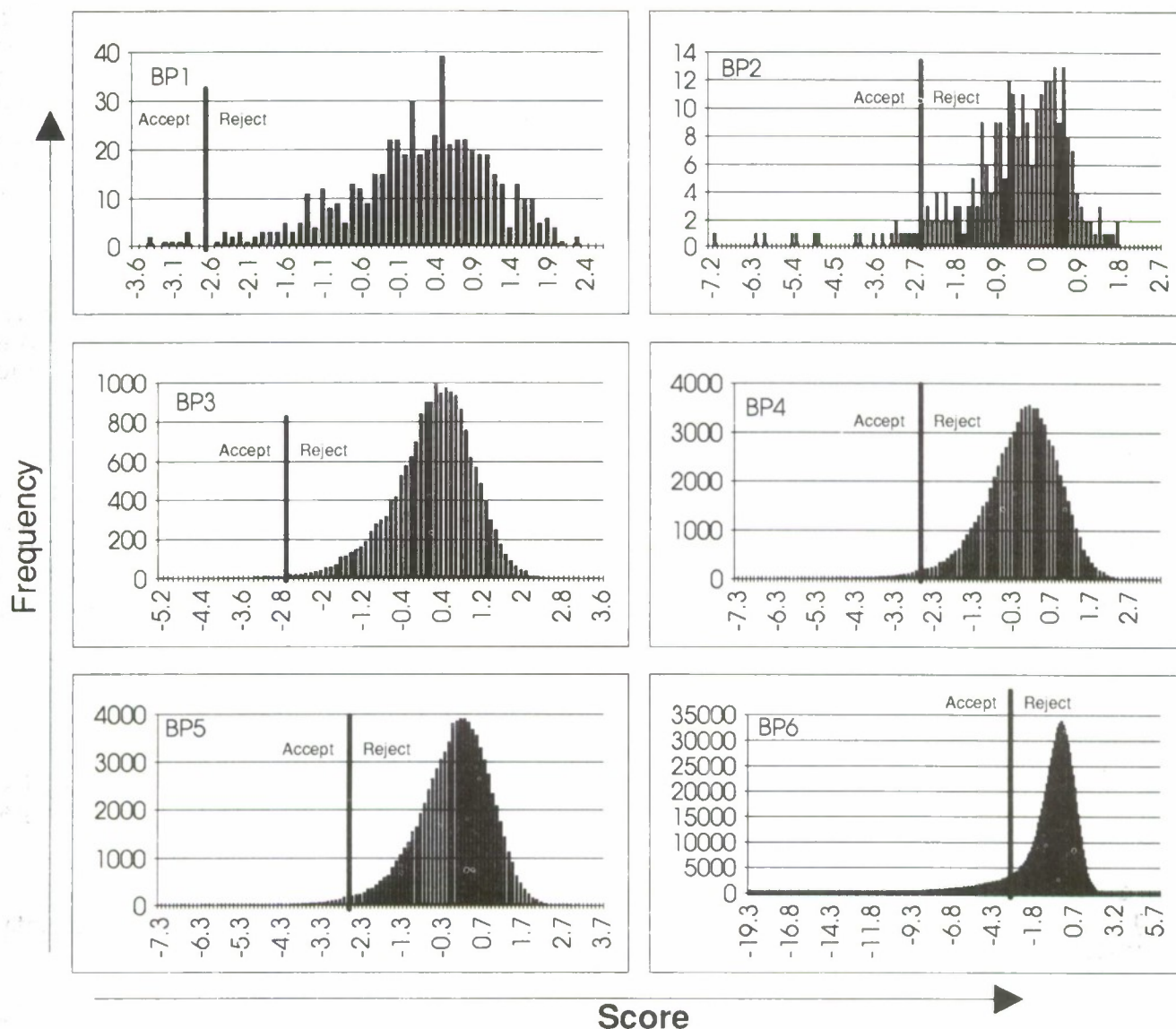


Fig 3: Histograms graphing BApsswd scores and the frequency of passwords in files BP1 - BP6. Our criterion of 2.6 Standard Deviations (99%), implies that all passwords to the left of -2.6 are classified as acceptable.

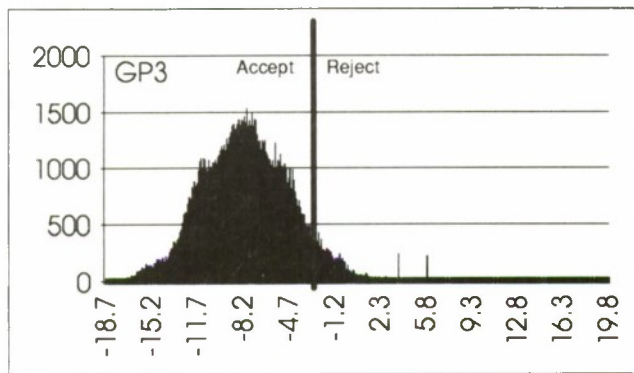
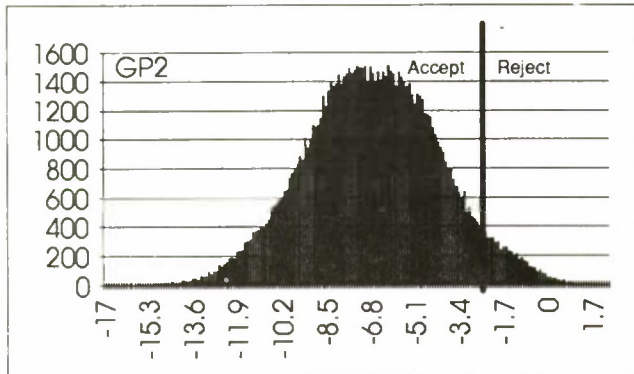
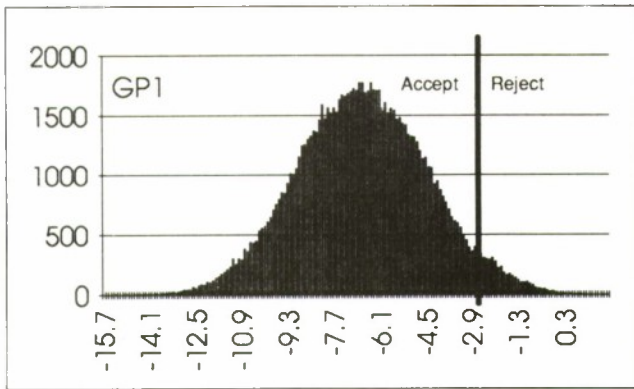


Fig. 4: Although it filters out bad passwords, these histograms show that BApaswd is successful in keeping in good passwords,

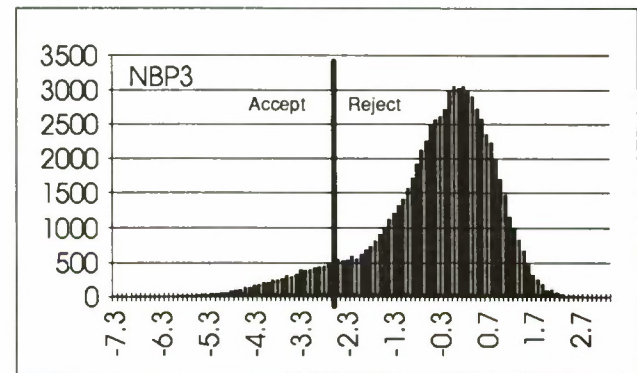
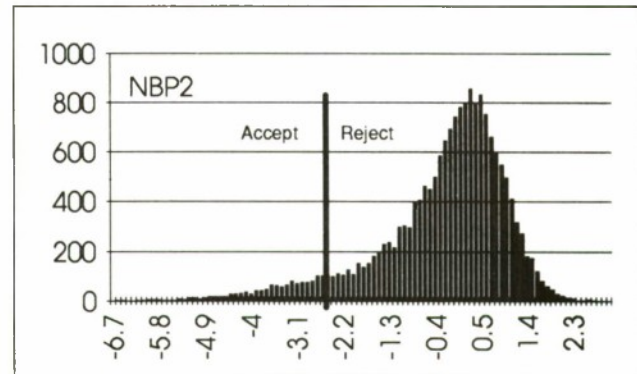
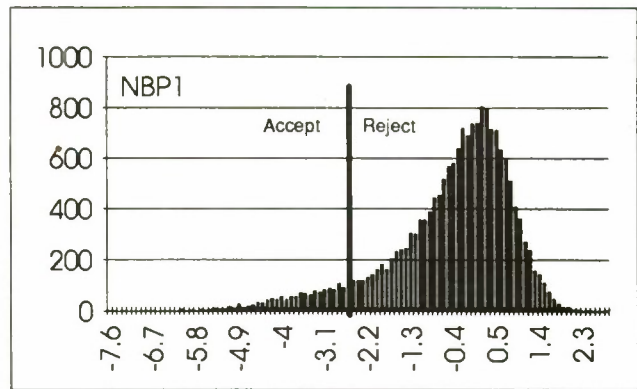


Fig. 5: The Histograms illustrate that BApaswd successfully filters out bad noisy passwords.

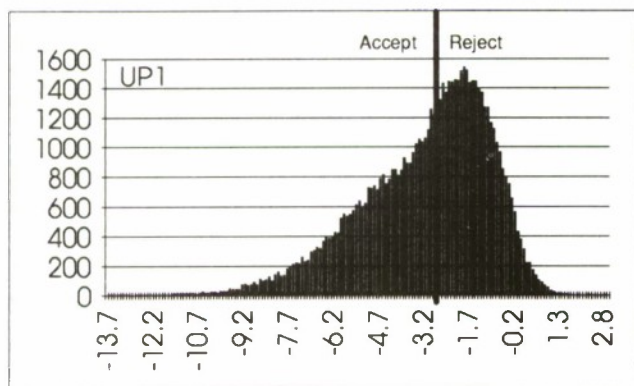


Fig. 6: Roughly 50% of passwords from a program that generated "random pronounceable passwords" were classified as unacceptable.

File	Number Accepted	Total Passwords	Percent Accepted
BP1	15	586	2.56%
BP2	19	280	6.79%
BP3	119	18780	0.63%
BP4	1418	80698	1.76%
BP5	1454	86536	1.68%
BP6	118223	961947	12.29%
GP1	96257	100000	96.26%
GP2	94403	100000	94.40%
GP3	111758	119916	93.20%
NBP1	1530	18780	8.15%
NBP2	1274	18780	6.78%
NBP3	6970	80698	8.64%
UP1	36119	69630	51.87%

Fig. 7: Summary Performance Evaluation of BApaswd. Shows percentages recognized as 'acceptable'. Note that some of the passwords in the 'bad' and 'bad noisy' password files are indeed acceptable.

- Some of the passwords our users selected looked fairly random, and we were concerned that they would not be able to remember them. However, on questioning we determined that while the passwords looked random to us, to the user they had a firm semantic link to some tangible memory. For instance the password *vill84mth* did not look easily memorizable, or even pronounceable, until the user explained that he had graduated from Villanova University in 1984 with a Math major!
- The conversational, interactive nature of BApaswd is critical. A user must be informed why her password was not accepted.

Our (scientific) experiments with GP1-GP3 and our (admittedly unscientific) usability tests with users, convinced us that usability would not be a major issue, and that users will be able to easily select passwords using BApaswd. Our users were comparing BApaswd to their current environment where they can pick *any* password. In environments where machine generated passwords are used, BApaswd could represent a major improvement in user friendliness, and could result in a decrease in the instances of passwords being written down or forgotten, which have their associated security and administrative costs respectively.

5.4 BApaswd and 'random pronounceable passwords'

We experimented briefly with passwords generated by the 'random pronounceable password generator' which we showed how to break in Section 3. As expected, several of these supposedly good passwords were classified as bad. This is expected given the way BApaswd works, but we do not see this as a problem for two reasons:

1. The passwords picked by users of BApaswd are very likely to contain special characters and would consequently not be pronounceable. However, these passwords will be *picked by the users*, and, in our opinion, will be more easily memorized than a machine generated pronounceable password.
2. We are not convinced that pronounceable passwords have an adequately large key-space. We have already shown how the scheme we tested can be broken, and we also point out that Gasser [12] clearly warns that passwords generated by his system may contain English words that need to be filtered out. If we extend this to include English words plus one character noise, then it is possible that a considerable number of passwords generated by his scheme are actually 'bad passwords', and consequently, it is not surprising that BApaswd recognizes them as such.

If the number of such bad passwords is significant, then perhaps the random pronounceable password generator should use a proactive password checker like BApaswd to ensure that no bad passwords are inadvertently generated.

5.5 Comparing BApaswd and the Nagle test

As mentioned in Section 3, the Nagle test performs remarkably well, except that it is flawed in the manner in which it handles bad noisy passwords. BApaswd is similar to the Nagle test in that both use trigrams as the base unit of information. BApaswd uses more information (the frequency of trigrams as opposed to whether a trigram is present or not) and uses tests based on statistical inference on Markov chains. We implemented the Nagle test and ran experiments comparing performance. these results are given in Fig-8 below:

Fig-8: A comparison of the Nagle Test and BApasswd

Password File	% classified as good by BApasswd	% classified as good by the Nagle Test	Comments
BP4	1.65	0.0	By definition, the Nagle test correctly classifies all passwords it was trained on, BApasswd is statistical in nature, and hence a small percent are mis-classified.
BP6	12.27	9.1	Similar comments as above, the 12% for BApasswd is partly because BP6, contains a small but significant fraction of words, that may be good passwords.
GP2	94.2	95.0	Both tests accurately classify random (good) passwords.
NBP1	7.91	52.6	Here is where BApasswd handily outperforms the Nagle test. These passwords which are dictionary words plus one special character, should be recognized as bad passwords.
NBP2	6.65	43.2	See comments for NBP1

Note that the “% classified as good by BApasswd” differs slightly from Fig-7 as for this experiment we trained both the Nagle test and BApasswd on BP4, whereas for Fig-7, BApasswd was trained on BP5.

6.0 Conclusion

By modeling bad passwords as a language generated by a Markov process, BApasswd:

- filters out bad passwords.
- filters out bad noisy passwords.
- does not have to use a large dictionary (saves space).
- is extremely fast and well suited for real-time situations.
- is compact - the complete code for both on-line and off-line modes is a couple of hundred lines of C.

Equally important, BApasswd achieves this without making it too difficult for a user to choose a good password. For these reasons we believe that BApasswd achieves the correct balance between security and user friendliness. The fact that the on-line code is extremely compact means it can be integrated into any password changing program very easily.

7.0 Acknowledgments

We would like to thank Dave Feldmeier, Bellcore for providing us with the BP4 and BP6 dictionaries of bad passwords, Gene Spafford, Purdue University, for information on OPUS. Rich Graveman, Bellcore, Peter Matthews, UMBC, Bernie Cavanaugh, Paul Haven, Al Leslie, Ray Pyle, Byron Stump, Kelvin Tucker and Aaron Waller, Bell Atlantic for encouragement, advice and/or for being ‘alpha-testers’ of BApasswd. We would also like to thank Ravi Sandhu, George Mason University for encouraging us to submit this paper to the National Computer Security Conference. Finally, we are deeply indebted to Steve Abney and Karen Kukich, Bellcore, for facilitating the migration of BApasswd to the more accurate 2nd order model, by encouraging us to use the Good-Turing adjustments to overcome the problems with zero element transition probabilities.

8.0 References

- [1]“Automated Password Generator (Draft)”, *Federal Information Processing Standards Publication*, National Institute of Standards and Technology, September 1992.

- [2] Bellare, S. and M. Merritt. "Encrypted key Exchange", *IEEE Computer Society Symposium on Security and Privacy*, May 1992, Oakland, CA.
- [3] Bellare, S. and M. Merritt, "Limitations of the Kerberos Authentication Protocol", *USENIX - Winter 91*, Dallas, Texas.
- [4] Bhat, U. N., *Elements of Applied Stochastic Processes*, John Wiley, New York, 1984.
- [5] Bishop, M., "Proactive Password Checking", *4th Workshop on Computer Security Incident Handling*, August 1992.
- [6] Bloom, B. H., "Space/Time trade-offs in Hash Coding with Allowable Errors", *Communications of the ACM*, 13(7), July 1970.
- [7] Church, K. W. and W. A. Gale, "A Comparison of the Enhanced Good-Turing and Deleted Estimation Methods for Estimating Probabilities of English Bigrams", *Computers, Speech and Language*, vol 5, 1991.
- [8] Denning, D. E. R., *Cryptography and Data Security*, Addison-Wesley, Reading, MA, 1983.
- [9] Ganesan, R. and A. Sherman, "Statistical Techniques for Language Recognition: An Introduction and Guide for Cryptanalysis", *University of Maryland TR CS -93-2*, 1993. (submitted to Cryptologia)
- [10] Ganesan, R. and A. Sherman, "Statistical Techniques for Language Recognition: An Empirical Study Using Real and Simulated English", *University of Maryland TR CS -93-3*, 1993.
- [11] Garfinkel, S and G. Spafford. *Practical UNIX Security*, O'Reilly and Associates, Sebastopol, CA - 1991.
- [12] Gasser, M., "A Random Word Generator for Pronounceable Passwords", *National Technical Information Service (NTIS) AS A 017676*.
- [13] Good, I.J., "The Population Frequencies of Species and the Estimation of Population Parameters", *Biometrika*, Vol 40. 1953.
- [14] Hoover, C., *npasswd version 1.7*. Available from ftp.cc.utexas.edu.
- [15] Karn, P. R. and D. C. Feldmeier, "UNIX password security - Ten years later", *Advances in Cryptology - CRYPTO '89*, G. Brassard (Ed.) *Lecture Notes in Computer Science*, Springer-Verlag, 1990.
- [16] Katz, S.M., "Estimation of Probabilities from Sparse data for the Language Model of the IBM Speech Recognition System", *IEEE Transactions on Acoustics, Speech and Signal Processing*, Vol. ASSP-32, 1985.
- [17] Kohl, J. C. Neuman and J. Steiner, "The Kerberos Network Authentication Service", *MIT Project Athena* (October 8, 1990) Version 5.3.
- [18] Morris, R. and K. Thompson. "Password security: A Case History", *Communications of the ACM*, 22(11), November 1979.
- [19] Nagle, J.B., "An Obvious Password Detector", *comp.sources.unix* 16(60) Nov. 1988. USENET message 1175@fig.bbn.com.
- [20] Sinkov, A., *Elementary Cryptanalysis: A Mathematical Approach*, The Mathematical Association of America, New Mathematical Library No. 22, Washington, D.C., 1966.
- [21] Spafford, E.H., "OPUS: Preventing Weak Password Choices", *Purdue Technical Report CSD-TR 92-028*, June 1991.
- [22] Spafford, E. H., *Personal Communication*, Feb. 1993.

NON-REPUDIATION IN OPEN TELECOOPERATION

Rüdiger Grimm

GMD, Dolivostr. 15, D-64293 Darmstadt, Germany

grimm@darmstadt.gmd.de

16th National Computer Security Conference, Sep 1993, Baltimore, MD

Abstract

This paper sketches a security model about non-repudiation of communicative actions. Non-repudiation is considered not only with regard to isolated actions, but with regard to the communication of mutual commitments such as the negotiation of a contract. The model defines a “balance” between *states of obligation* of all participants of a telecooperation and the *proofs* of these states. Proofs are based on asymmetrically computed digital signatures. A simple obligation logic is used in order to express obligation states. The *global* balance is supported by *local* user agents.

This paper gives a general overview over the “Balance Model” and presents a semi-formal method to describe states of obligation, the change of these states, and the requirements for proofs of these changes. As an example, the model is applied to a simple cooperation between two economic partners, a provider and a user of an application service, e.g. an information service or the remote use of an IT-system. In an obvious way this example can be interpreted as the bilateral negotiation of a contract.

Keywords

Obligation, commitment, proof of action, responsibility, non-repudiation, data integrity, digital signature, separation of duty, security in open systems.

1 Introduction

Non-repudiation of a single promise is usually achieved by a digital signature of the promise attached to the data which contain the promise. Non-repudiation of the receipt of a message can be achieved by a digitally signed report of delivery. The technical basis for a non-repudiable proof of a single communication act across communication networks is the concept of a digital signature due to [DIHE 76]. The signature scheme is called *asymmetric* because it allows a person to sign a text with the help of a personal private key which must not be disclosed to an untrusted party. A personal private key is protected within a private environment of its owner, typically a smartcard. The digital signature is the image of a one-way function which is applied to the signed text. The one-way function is parameterized by the signer’s private key. For verification of the signature, in contrast, a public key is used which uniquely identifies the signing person. This way, the digital signature proves both the

integrity of the signed text and the authenticity of the originator of the text.¹ Therefore, the verification of a digital signature is more than a simple data consistency check. Data integrity and origin can be verified by any neutral third party even outside a trusted domain. This satisfies one requirement of a legal proof.

Non-repudiation is addressed in the OSI security model [OSI 84]. The security model of the X.400 message handling recommendations [X400 88] defines security service elements which are designed to encounter non-repudiation of message origin or non-repudiation of message receipt. However, these elements refer to isolated messages. Cooperative actions are driven by more complex obligation structures. Promises are given *under the condition* that others perform certain well-defined actions before. Cooperating partners are "glued together" by obligation states of mutual conditional promises. Consider, for example, a reservation agent and his client. The client would promise: "if you reserve me a seat then I will pay for it," while the service agent commits himself to the promise: "if you pay me then I will reserve you a seat." Now, how do these two persons achieve the goal of cooperation which consists of both, payment *and* reservation?

The "Balance Model" describes a "balance principle" which helps cooperating persons to protect this type of cooperative goals. We consider persons to cooperate with the support of locally implemented cooperation user agents which use an underlying telecommunication system.

One way, of course, would be to enforce the cooperation rules by automatic mechanisms internal to the cooperation system. That is, the system wouldn't allow for a wrong behaviour of a partner. This requires global control over the whole system. However, in an open environment such as the word-wide economic market this is not feasible. In an open environment one must reckon with flawed remote user agents: a service provider might, for example, receive a payment but not perform the service.

Therefore, global enforcement by system mechanisms is replaced by local support of the personal responsibilities of the cooperating persons. Locally available system mechanisms observe the receipt of proofs, and as a sign of good behaviour they produce proofs duly expected by the partners. Even if a remote partner is able to break the rules the local partner is secure in that he can prove the remaining obligations of his partners. This is the *balance principle* of a reliable telecooperation system which can be implemented locally: *states of obligations* of all participants of a telecooperation and the *proofs* of these states have to be balanced. The change of state of obligation must be compensated by a proof of this change. The local user agent observes this balance. A person can *decide securely* about the progress of a cooperation in that he examines the obligation states of his partners. If he is able to prove them, he can resume the cooperation safely. Otherwise, he interrupts the cooperation and demands his rights on the basis of the proofs received until this point. He can also encounter unjustified demands of others at any point. Proofs are based on digital signatures. States of obligations are described by logical expressions introduced later in this paper.

¹For details see W. Diffie and M. Hellman 1976 in [DIHE 76] where the idea of asymmetric encryption algorithms is developed. For successful realizations of asymmetric digital signature schemes see [RSA 78] and [ELGA 85]. One concept of certification of public keys is detailed in [X500 88]. There are procedures available which support digital signatures. For example, see the specifications of the "Privacy Enhancement for Internet Electronic Mail (PEM)" by the Internet Research and Engineering Task Forces [PEM 93].

2 The Balance Model

2.1 Cooperative Goals and Obligations

All participants of a cooperation agree on an explicitly specified syntactic goal. The *cooperation principle* determines that *either every or no partner achieves the cooperative goal*. An example of a cooperative goal is a unique contract. Even if there is a different understanding of the semantic content of a contract, the syntactic wording of a contract must be clear and the same everywhere.

The aim is to protect the cooperation principle with respect to the common syntactic goal of a cooperation regardless of possible semantic conflicts about this goal. A goal of a cooperation can be a set of single activity goals. For example, if the receipt of money is the activity goal of a selling person, and the receipt of a good is the activity goal of a buying person, then the common syntactic goal of the purchase cooperation is the aggregated set of these two goals. In such a case there is a danger that one partner achieves his goal while the other partner does not achieve his goal. Another kind of a cooperative goal can be one single event, towards which all participants of a cooperation move. For example, the completion of an order form by two employees according to the four-eyes control principle is the goal of a separation-of-duty cooperation within one organisation. In this case there is a danger that one partner misuses his power in order to obstruct the goal or to enforce the goal against the rules in force.

The basic idea is to embed the single actions of the partners which are relevant for the goal into so called *obligation structures*. The single actions of the partners are related to one another in a way that a step of one partner commits the other partner to perform the next step. This way the partners approach the goal stepwise. In the special case of an aggregated cooperative goal the different activity goals are tied together in a way that the achievement of one activity goal by *one* partner *obliges* him to help the *other* partner to achieve *his* goal as well.

2.2 The Example of the Bilateral Offer-Order Cooperation

This example of a cooperation between a service provider *p* and a service user *u* will be used throughout the paper. The purpose of this simple example is to demonstrate the idea of the balance principle. More complex cooperations would require a more complex analysis, however based on the same principle of balance.

The offer-order cooperation includes two types of actors, a service provider *p* and a service user *u*. They exchange messages with the intention to create, change and resolve states of mutual obligations. The service is an abstraction from any service which can be realized by an IT system, for instance an information service, a directory service, a remote system use, the reservation of a ticket, etc. The goal of the service user is to receive a service result from the service provider. Depending on the service, the result is a piece of information or a system reply or a ticket confirmation, etc. The goal of the service provider is to be paid by the service user. The cooperative goal is the aggregation of the two activity goals.

There are four basic types of messages: an *offer*, an *order*, a *result*, and a *cheque*. Each message type has an associated request, acknowledge, and refuse message type, e.g., *offer_{please}*, *cheque_{ack}*, or *result_{refuse}*. The “request” message type is used in order to call on a partner to send a message of the requested type. The “acknowledge” message type is used in order to express explicitly the receipt of a message regardless if this message is accepted as correct or

not. A “refuse” message type is used in order to express the opinion that the referred message is incorrect. It will become clear later in this paper how persons determine the correctness of messages. The “refuse” message type can be combined with a “request” message type in order to repeat the questionable step of cooperation.

The product which the service provider sells is expressed by a message of the type *result*. The result is related to a message of the type *order* which expresses the will of the service user to receive the *result* and pay for it. The payment is expressed by a message of the type *cheque*. The relationship between possible results and required payments for a result is expressed by a message of the type *offer*. The goal of the cooperation is expressed by the statement “*u* receives *result* and *p* receives *cheque*”.

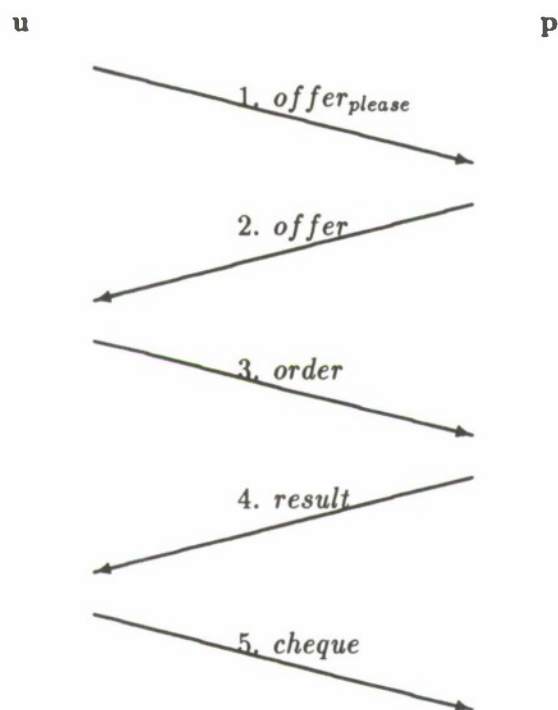


Fig. 1: The simple bilateral “offer-order cooperation” without explicit acknowledgements or refusals

The last two steps represent the goal of the cooperation. The way to the goal is directed by personal obligations. The cooperation is initiated by the service user who asks the service provider for an offer. This request does not commit anyone. The service provider either ignores this request or responds with an offer. An offer, however, creates a *state of a conditional obligation* for the service provider:

*If p sends an offer, and if u replies with an order,
then p is obliged to send a result.*

Until this point, the service user is free of obligations. However, an order creates a *state of a conditional obligation* for the service user:

*If u sends an order, and if p replies with a result,
then u is obliged to send a cheque.*

These two expressions which are called *obligation expressions*, describe the obligation states of the partners of this cooperation. Note, how the single steps are tied together. *All* actions are interrelated. Note in particular that the last action of the service provider serves three purposes at once: First, the result fulfills his own obligation. Second, it is the activity goal of his partner. Third, it fulfills the last condition of his partner who by this condition is now unconditionally obliged to make *his* last step. The last action of the service user serves two purposes: First, the cheque fulfills his own obligation. Second, it is the activity goal of his partner. Those last two actions make the cooperative goal be achieved. Note that the tie is not by automated actions of protocol machines, but by obligations of persons. Why this must be so, will be explained in the following subsection.

2.3 Obligation Structures and their Logical Expressions

In a cooperation a partner passes discrete local states. The transition of one state to another happens by the occurrence of a local event. An event is either the receipt or the transmission of a message at the external message interface of a partner. In particular, every partner of a cooperation is associated with a *state of his personal obligation*. An obligation state is described by a *logical expression of obligation*. An obligation expression is of the following form:

If events $(\tau_1, \tau_2, \dots, \tau_v)$ of predefined types have occurred, then the respective person is obliged to proceed with an event τ_{v+1} of a predefined type.

An obligation expression is *true*, if one of its suppositions $(\tau_1, \tau_2, \dots, \tau_v)$ has not been fulfilled, *or* if the concluding event τ_{v+1} is fulfilled. An obligation expression is *false*, if *all* of its suppositions $(\tau_1, \tau_2, \dots, \tau_v)$ have been fulfilled, *and* if the subsequent event τ_{v+1} is not (yet) fulfilled. It is the *personal responsibility* of every partner in a cooperation to keep his personal obligation expression true during the whole cooperation.

However, it is not quite clear what it means, that an event is “fulfilled”. It is not the format alone. It is important to note that the semantic *content* of an event is not fully determined syntactically by its *type*. For example, a contract can have an incorrect content despite its correct format. Also the reverse case happens: a contract is semantically correct while it contains formal defects. An event is *true*, or *fulfilled*, if and only if it has a correct type and a correct content, i.e. if it is both syntactically and semantically correct. Unfortunately, for automata it is not easy to handle semantic correctness. On the other hand, the treatment of syntactic correctness is straight forward.

As to *syntax* specification, every state E_i out of a sequence of states (E_1 “before” E_2 “before” ... E_{v+1}) at which events can take place is associated with a set T_i of specified event types:

$$T_i = \{\tau_{i1}, \tau_{i2}, \dots, \tau_{ie_i}\} \quad (e_i \in N)$$

Recall that an event is the receipt or the transmission of a message at the external message interface of a partner. Every $\tau_{ij} \in T_i$ ($i \in \{1, \dots, v+1\}, j \in \{1, \dots, e_i\}$) represents one syntactic alternative of the state E_i , i.e. at E_i exactly one event m_i will happen with $Type(m_i) \in T_i$. At every state E_i a protocol instance can check automatically if an occurring event is an expected event as far as its type is concerned.

As noticed before, an automatic check of *semantic* correctness is not easy if feasible at all. In general, this is not feasible and therefore human persons perform this task. However, in order to understand the position of the persons in the model, for a short moment we assume the existence of an automatic semantic checker: under this assumption a protocol could be defined which *automatically* fulfills a conditional obligation. This would lead to the following protocol machine: First, for every state E_i ($i = 1, \dots, v+1$) the automaton selects one event type $\tau_{i,j_i} \in T_i$ as the "correct type" at this state. Then the events are interrelated by the following expression of *temporal logic*:

$$(\tau_{1,j_1} \wedge \tau_{2,j_2} \wedge \dots \wedge \tau_{v,j_v}) \Rightarrow F(\tau_{v+1,j_{v+1}})$$

"F" represents the "following"-operator of temporal logic. The meaning of this temporal-logical expression is this:

If at all states E_i ($i = 1, \dots, v$) correct events of type τ_{i,j_i} have occurred, then at state E_{v+1} the respective protocol instance selects a correct event of type $\tau_{v+1,j_{v+1}}$ and resumes the cooperation with it.

The type selection covers the syntax. However, the temporal expression assumes semantic correctness as well. Despite the fact that semantic correctness cannot be specified generally, there is an even more important argument against this protocol realization from the *security* point of view. In an open environment, one cannot *rely* upon a partner instance to follow this temporal logic rule. The reason is that semantics go with personal interest, and personal interest is a major source for security attacks.

Therefore, *persons* are introduced into the model. A person owns personal competence which includes a semantic understanding of the cooperation and a personal interest in its goal. Persons are responsible for their doing. In the stead of an *automatic* subsequent event there is a *personal obligation* to make the correct subsequent event happen. Every partner in a cooperation is associated with an obligation state which is described by an obligation expression of the form:

$$(\tau_{1,j_1} \wedge \tau_{2,j_2} \wedge \dots \wedge \tau_{v,j_v}) \Rightarrow O(\tau_{v+1,j_{v+1}})$$

"O" represents the "obligation"-operator of deontic logic. The meaning of this expression of obligation logic is analogous to the respective expression of temporal logic, whereby the "following"-operator is replaced by the "obligation"-operator:

If at all states E_i ($i = 1, \dots, v$) correct events of type τ_{i,j_i} have occurred, then at state E_{v+1} the respective *person is obliged* to select a correct event of type $\tau_{v+1,j_{v+1}}$ and to resume the cooperation with it.

The partners evaluate this logical expression by their personal competence. The logical evaluation covers both, the syntactic and semantic aspect of the events, in a natural way. There is neither a restriction nor a demand of an automatic support of this evaluation on any side. In particular, a participant does not take it as a matter of course, that his partner will act according to his obligation. Instead of an automatic enforcement, he expects proofs of evidence which enable him to enforce the obligation outside of the technical system, if necessary. This will be outlined later in this section (see subsections 2.6 and 2.7).

At this point it might be helpful to look for a moment at an embedding of the model of obligations presented so far into a more general model of telecooperation. Semantics and cooperating persons are essential elements of the Balance Model. Therefore, the embedding can be described in the terminology of J. Dobson and J. McDermid [DOBS 89]. They have stressed the specific problem that security models should relate to semantic meaning rather than to data or to predefined information types. They notice that “meaning in the context of an enterprise is a social construction”. Consequently, they introduce human *individuals* into their security model who use *interpretation functions* in order to manipulate and evaluate data. The interpretation functions work on a background of a social context and are not (or at least not fully) specified. In the Balance Model of obligations presented here, it is by those interpretation functions that persons evaluate the obligation expressions. From the point of view of a formal model, these functions are there but they are underspecified. However, the model supposes that persons do evaluate obligation expressions and that in a social context of cooperation the principles of evaluation are agreed. This includes the possibility of conflicts. The conflicts can be communicated and eventually solved on the basis of syntactic proofs of the obligation states.

2.4 Obligation States in the Offer-Order Cooperation

The example of a cooperation between a service provider p and a service user u was explained in subsection 2.2 (cf. fig. 1, p.4). In this cooperation either partner passes a sequence of five obligation states:

	event types T_i for p	event types T'_i for u	comment
step 1 Initiation	T_1 at state E_1 contains: τ_{11} : p receives <i>offer_{please}</i>	T'_1 at state E'_1 contains: τ'_{11} : u sends <i>offer_{please}</i>	
step 2 Service Offer	T_2 at E_2 contains: τ_{21} : p sends <i>offer</i>	T'_2 at E'_2 contains: τ'_{21} : u receives <i>offer</i>	
step 3 Service Order	T_3 at E_3 contains: τ_{31} : p receives <i>order</i> τ_{32} : p receives <i>offer_{refuse}</i> τ_{33} : p receives <i>offer_{please}</i>	T'_3 at E'_3 contains: τ'_{31} : u sends <i>order</i> τ'_{32} : u sends <i>offer_{refuse}</i> τ'_{33} : u sends <i>offer_{please}</i>	resume at E_2
step 4 Service Result	T_4 at E_4 contains: τ_{41} : p sends <i>result</i> τ_{42} : p sends <i>order_{refuse}</i> τ_{43} : p sends <i>order_{please}</i> τ_{44} : p sends <i>offer</i>	T'_4 at E'_4 contains: τ'_{41} : u receives <i>result</i> τ'_{42} : u receives <i>order_{refuse}</i> τ'_{43} : u receives <i>order_{please}</i> τ'_{44} : u receives <i>offer</i>	resume at E_3 resume at E_3
step 5 Payment	T_5 at E_5 contains: τ_{51} : p receives <i>cheque</i> τ_{52} : p receives <i>result_{refuse}</i> τ_{53} : p receives <i>result_{please}</i> τ_{54} : p receives <i>order</i>	T'_5 at E'_5 contains: τ'_{51} : u sends <i>cheque</i> τ'_{52} : u sends <i>result_{refuse}</i> τ'_{53} : u sends <i>result_{please}</i> τ'_{54} : u sends <i>order</i>	resume at E_4 resume at E_4

Fig. 2: States and event types in an offer-order cooperation.

Generally, this cooperation is completed by a positive acknowledgement of cheque receipt.

Corresponding events τ_{i,k_i} at state E_i of p and τ'_{i,k_i} at state E'_i of u are interrelated by the communication infrastructure. In case of a secure communication channel, corresponding states of p and u can be identified. This aspect of a communication infrastructure is explained in subsection 2.5 below.

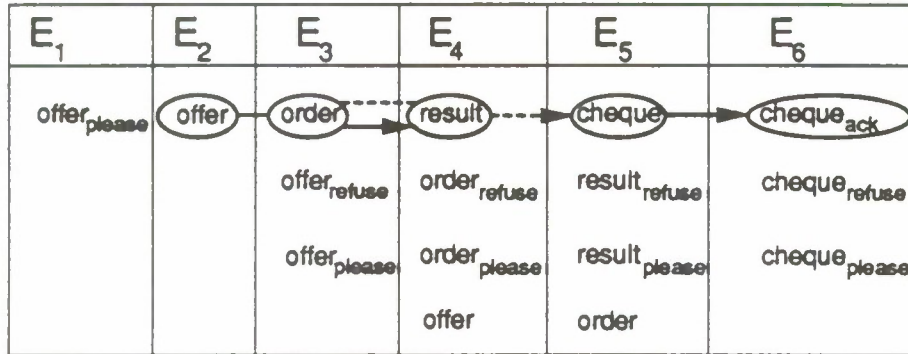
Without further restriction, every sequence of event types $(\tau_{i,k_i})_{i=1,\dots,6}$ is legal, for instance the sequence $(\tau_{11}, \tau_{21}, \tau_{31}, \tau_{42}, exit)$ which corresponds to a refusal of a user order by the service provider. Depending on the semantic content of this cooperation instance, *order_refuse* either expresses a legal refusal of an unacceptable order or an illegal denial of service. However, the property of an event of being legal or illegal is derived from the obligation structure in the following way:

The obligation states put an additional structure on the cooperation. The service provider p in the offer-order cooperation is conditionally obliged to send a result to the service user u , under the two conditions that he had sent an offer before and that he has received an order which matches the offer. u , on the other hand, is conditionally obliged to send a cheque to p , under the two conditions that he had sent an order before and that he has received a result which satisfies the order. There is another conditional obligation of the service provider, namely to acknowledge an accepted payment positively. Other obligations of acknowledgements could be introduced in the cooperation. For the sake of simplicity, however, they are not expressed here.

Let $s(\tau : \tau)$ denote the event that a person of type s receives a message m of type τ from a person of type r . And let $s(-\tau : \tau)$ denote the event that a person of type s sends a message m of type τ to a person of type r . Let $OE(s)$ denote an obligation expression of a person of type s . Then the obligation states of p and u are formally described by the following obligation expressions:

$$\begin{aligned} (OE(p)) \quad [p(-offer : u) \wedge p(order : u)] &\Rightarrow O(p(-result : u)) \\ &\quad p(cheque : u) \Rightarrow O(p(-cheque_{ack} : u)) \\ (OE(u)) \quad [u(-order : p) \wedge u(result : p)] &\Rightarrow O(u(-cheque : p)) \end{aligned}$$

With this additional structure the single activity goals of the two partners are tied together to one aggregated goal of the cooperation. In that u receives the result, he achieves his goal and he fulfills his last condition by which he is now unconditionally obliged to make p achieve his goal as well.



—— obligation structure of P ——

----- obligation structure of U -----

Fig. 3: Sequence of obligation states with events and obligation structures. Without obligation structure: Every event path from left to right is equally possible. With obligation structure: Preferred event paths. After following a preferred path until the last but one event τ_{i_v, j_v} , the next event *must* be the last event τ_{i_v+1, j_v+1} of this preferred path.

2.5 A Semi-formal Description of a Reliable Telecooperation

There are *six* aspects to be described.

The *first* aspect refers to the definition of the *subject types* and *message types* of a cooperation. In the offer-order cooperation the subject types are *p* and *u*, and the message types are *offer*, *order*, *result*, and *cheque* together with the associated request, acknowledge, and refuse message types.

The *second* aspect is a description of the *communication infrastructure*. Logical expressions describe the motion of messages. For example, the expression

$$s(m : r) \Rightarrow r(-m : s)$$

states that if a subject *s* has received the message *m* from the subject *r*, then *r* has indeed sent this message *m* to *s*. This is the non-repudiation of origin. The converse statement of motion,

$$s(-m : r) \Rightarrow r(m : s)$$

which expresses the non-repudiation of receipt, is only true, if the sender *s* is in possession of a proof-of-delivery by the network provider or by another neutral observer. Otherwise, the statement should include the supposition that *s* has received a personal receipt acknowledgement by the message recipient:

$$s(-m : r) \wedge s(m_{ack} : r) \Rightarrow r(m : s).$$

The delivery of a sent message and the origin of a received message, respectively, can be derived from the expressions of message motion.

In the example of the offer-order cooperation, the motion of messages between *p* and *u* within a secure communication infrastructure which provides proofs-of-delivery, is described by the following logical expressions (C1) and (C2). For simplicity, from an expression $s(-m : r)$ or $s(m : r)$ the indexed recipient or originator *r*, respectively, is dropped, if recipient and originator, respectively, are clear. In a bilateral cooperation they are clear. Therefore, $p(-m)$ stands for $p(-m : u)$, etc.

$$(C1) \quad p(-m) \Leftrightarrow u(m),$$

$$(C2) \quad u(-m) \Leftrightarrow p(m).$$

The implications " $p(-m) \Rightarrow u(m)$ " and " $u(-m) \Rightarrow p(m)$ " (i.e. non-repudiation of delivery without cooperation of the recipient) hold only if a secure communication infrastructure supplies the sender with a proof-of-delivery. However, if a secure communication infrastructure is *not* available, the expressions of message motion must contain personal acknowledgements in order to prove the receipt of a message. (C1) and (C2), for example, would then expand to

$$(C'1) \quad p(-m) \wedge p(m_{ack}) \Rightarrow u(m),$$

$$p(-m) \Leftarrow u(m),$$

$$(C'2) \quad u(-m) \wedge u(m_{ack}) \Rightarrow p(m),$$

$$u(-m) \Leftarrow p(m).$$

The *third* aspect is a description of the *obligation structure* defined by the obligation states of all partners. Obligation states are formulated and interpreted with the help of an obligation logic by so-called obligation expressions. The obligation structure of the offer-order

cooperation is described by the obligation expressions $OE(p)$ and $OE(u)$ (see subsection 2.4 above).

The *fourth* aspect is the definition of the common *goal of cooperation*. It is described by a logical expression of events. The formal goal of the offer-order cooperation, for example, is described by

$$(G) \quad u(result) \wedge p(chegue).$$

This means that u demands the service and p wants to earn money. u wants the service result and p wants the cheque. They cooperate in that they agree to achieve both as a cooperative goal.

Note that it is the purpose of the obligation structure to support the cooperative goal. When designing a cooperation system, obligations are structured in a way that the partners are *obliged* to help one another mutually in achieving the cooperative goal.

The *fifth* aspect is the *equilibrium* of obligation states and proofs about these states. This equilibrium, or balance as it is called, is realized by the cooperation protocol. Every partner observes the balance and reacts accordingly, if any partner tries to change the obligation state without supplying an appropriate proof. Also, unacceptable events would violate the balance. The observation and reaction is supported by the local system. The following subsections 2.6 and 2.7 are dedicated to this fifth aspect, the balance principle.

The *sixth* aspect is, of course, the *cooperation protocol* itself.

2.6 The Security Aspect of the Balance Model

Note that the messages are related to one another not only by syntactic rules, but also by semantic correctness. While the syntax of the events underlies objective rules, semantics can be subject to *conflicts*. The judgement of semantic correctness depends on the pragmatic background of knowledge and interest of a person. However, on the basis of non-repudiable syntactic proofs, conflicting persons can cooperate securely. The security aspect is described by the identification of the security requirements and the security measures which are designed to encounter the security threats.

Security Requirement:

Every partner of a cooperation is personally responsible for his personal obligation expression. He must keep it true throughout the cooperation. A participant is protected against false obligation expressions of his partners. A participant is protected against unjustified accusations that his personal obligation expression is false.

Security Measure:

A participant of a cooperation is protected by means of syntactic proofs of all messages which are intended to change an obligation state ("balance principle"). Proofs are based on asymmetric digital signatures. They prove the origin of received messages and the delivery of sent messages. They also prove the integrity of messages.

Security Threats:

The violation of the cooperation principle by an incorrect system or by incorrect behaviour of a participant is a security threat against a reliable cooperation. There are two types of

security attacks:

1. A proof is denied or manipulated.
2. An obligation expression becomes false.

A participant of a cooperation identifies a security attack of the first kind by a (purely syntactic) type check of the expected proof. A participant of a cooperation identifies a security attack of the second kind by a local (syntactic and semantic) evaluation of the obligation expression. The question if there is really a violation of the obligation state or if there is only a wrong understanding on one side can be subject to a personal conflict between cooperation partners. Both, violation and misunderstanding, would lead to an evaluation of the semantic aspects of the cooperation.

As a first approach, either attack leads to an interruption or termination of the cooperation. The conflict is solved outside of the cooperation in question. It is beyond the scope of this model to introduce elegant methods which support an automatic solution of a conflict. This is a problem of group communication technology. However, a telecooperation system which follows the balance principle supports the solution of a conflict, in that it supplies proofs of the current obligation states of all partners which are commonly accepted until the point of interruption. The proofs are also valid in front of neutral third parties like legal courts. The problem in question will be discussed between the partners, and in front of a third party if necessary. As to semantic conflicts (attacks of the second kind), the discussion includes the semantic meaning of the message contents, i.e. the information messages are intended to carry.

2.7 Proofs of Events

It is the intention of a participant of a cooperation who cannot trust his partners in cases of conflict to prove unfulfilled obligations of his partners or his own fulfilled obligations, respectively. For this purpose he collects messages which allow him to derive an obligation state to be proved. For any proof, three pieces are used: first, the initial obligation states; second, proved information about events, so called “elements of proof”; and third, the expressions of the message motion within the communication infrastructure. With the combination of these three pieces a participant can prove any current state of obligation.

An event is the delivery or the submission of a message at a communication interface of an actor. An “element of proof” (of an event) is a received message or a proof-of-delivery of a sent message. Now, about *which* events should a participant collect proofs? Fortunately, this can be formally derived from the obligation expressions of all partners.

Every participant is responsible that his own obligation expression remains true. Therefore, everyone collects messages which prove the *truth of his own* obligation expression and, if existent, messages which prove that an obligation expression *of a partner is false*. The proof of the truth of his own expression serves the defence in case of an unjustified accusation. A proof of a false obligation expression of a partner, if existent, serves the legal enforcement of a justified claim.

A proof of the truth of an obligation expression consists of a proof that one of its suppositions is false, *or* if this does not exist, of a proof that the conclusion is true. For example, the service provider p in the offer-order cooperation proves the truth of his obligation expression $(OE(p)) : [p(-offer) \wedge p(order)] \Rightarrow O(p(-result))$ in that he proves that $p(-offer)$ or $p(order)$ has not happened, *or* if both have happened, that $p(-result)$ has happened as well.

A proof that an obligation expression is false consists of proofs about *all* suppositions that they are true, *and* of a proof that the conclusion is not (yet) true. For example, the service user u in the offer-order cooperation proves that p 's obligation expression ($OE(p)$): $[p(-offer) \wedge p(order)] \Rightarrow O(p(-result))$ is false in that he proves that both, $p(-offer)$ and $p(order)$ have happened correctly, *and* that $p(-result)$ has not happened yet or that it was incorrect. This proof serves the purpose to force p to fulfill his obligation, i.e. to provide the ordered result. A proof about a false event expression refers to an event which has not happened or which has happened incorrectly. In order to be able to recognize unsent or lost messages, time intervals are defined.

The replay threat might lead to false proofs. An old message retransmitted by an eavesdropper or by one of the legitimate parties might look correct and would be accepted to "prove" an event. Examples are used cheques, out-of-date orders, or overbooked reservations. While some kinds of replay threats are of semantic nature such as an overbooked reservation, others like cheques and order forms can be protected by "time stamps" or "universal identifiers". It is the responsibility of a local actor within a cooperation to maintain and check identifiers and time intervals. For example, as an enterprise security policy, a service provider can map unique identifiers to all of its offers and accept only those orders which refer to one of its registered identifiers. The integrity protection of messages by a digital signature includes time stamps and identifiers.

3 The Balance in the Offer-Order Cooperation

The bilateral offer-order cooperation between a service provider p and a service user u is explained in subsection 2.2 (cf. fig. 1, p.4). In steps 1 through 5 the message types *offer_{please}*, *offer*, *order*, *result*, and *cheque* are exchanged. The corresponding obligation states $OE(p)$ and $OE(u)$ are presented in subsection 2.4. They are essentially expressed by

$$\begin{aligned} (OE(p)) \quad [p(-offer) \wedge p(order)] &\Rightarrow O(p(-result)) \\ (OE(u)) \quad [u(-order) \wedge u(result)] &\Rightarrow O(u(-cheque)) \end{aligned}$$

Note that step 1 is the receipt of *offer_{please}* and does not change an obligation state. The table in fig. 4 below compares the obligation states with elements of proofs about these states at every single step of the offer-order cooperation. In the notation of obligation states, fulfilled suppositions are left away in order to express the fact that they do not longer play the role of a condition for the concluding obligation. Then, a conditional obligation with all conditions fulfilled is expressed as an unconditional obligation which is indeed equivalent.

	Obligation States		Proof Elements	
	p	u	p	u
initial state and state after step 1	$[p(-offer) \wedge p(order)] \Rightarrow O(p(-result))$	$[u(-order) \wedge u(result)] \Rightarrow O(u(-cheque))$	-	-
after step 2	$p(order) \Rightarrow O(p(-result))$	$[u(-order) \wedge u(result)] \Rightarrow O(u(-cheque))$	-	$u(offer)$
after step 3	$O(p(-result))$	$u(result) \Rightarrow O(u(-cheque))$	$p(order)$	$u(offer) \wedge u(-order)$
after step 4	-	$O(u(-cheque))$	$p(order) \wedge p(-result), p(-result)$	$u(offer) \wedge u(-order)$
after step 5	-	-	$p(order) \wedge p(-result), p(-result)$	$u(offer) \wedge u(-order), u(-cheque)$

Fig. 4: Obligation states and proof elements in an offer-order cooperation.

Every participant collects only *those* elements of proof which he needs in order to prove the fulfilled suppositions of his partner or his own fulfilled obligations. For a proper book-keeping he will also make entries about other events such as the resolution of a partner's obligation or the fulfillment of a supposition of his own obligation. However, they play no role for the balance between obligations and proofs.

p collects $p(order)$ and $p(-result)$ because from these elements he can derive that the suppositions of the obligation $O(u(-cheque))$ of u with regard to him are fulfilled. p collects $p(-result)$ for the additional reason that it serves to prove the fact that his own obligation $O(p(-result))$ with regard to u is resolved. u collects $u(offer)$ and $u(-order)$ because from these elements he can derive that the suppositions of the obligation $O(p(-result))$ of p with regard to him are fulfilled. u collects $u(-cheque)$ for the additional reason that it serves to prove the fact that his own obligation $O(u(-cheque))$ with regard to p is resolved.

An obligation of a partner passes different states during a cooperation. Initially it is "conditional", then one supposition after the next is fulfilled. Immediately before fulfillment of the obligation its state is "unconditional". Eventually, after fulfillment of the obligation it is "neutral". Proofs increase accordingly. They are stored until after solution of possible conflicts. As one can see from the table in fig. 4 above, proofs and obligation states are balanced between the partners, such that any unfulfilled obligation can be proved by the respective other partner at every step of the cooperation.

From the point of view of u , the messages $u(offer)$ and $u(-order)$ are not yet proofs of an obligation state of p . These elements of proof must yet be combined with the *expressions of motion* (C1) and (C2). This is also true for the elements of proof $p(order)$ and $p(-result)$ from the point of view of p .

One example of a proof is demonstrated now. How does u prove after step 2 that the new current obligation state of p is now " $p(order) \Rightarrow O(p(-result))$ "? As described in subsection 2.7 above, he uses his elements of proof, the expressions of message motion, and the initial obligation state of p . The element of proof $u(offer)$ and the expression of motion (C1): $u(offer) \Rightarrow p(-offer)$ together imply $p(-offer)$. This is precisely the supposition

in the initial obligation state of p from which the new state is concluded: $p(-offer)$ and $(OE(p)) : [p(-offer) \wedge p(order)] \Rightarrow O(p(-result))$ imply " $p(order) \Rightarrow O(p(-result))$ " which was to be proved.

All other proofs of new obligation states are analogous.

4 Conclusion

The cooperation principle associates every cooperation with a cooperative goal and states that either all or none of the partners of a cooperation must achieve the common goal of the cooperation. The Balance Model sketched in this paper describes the balance principle which leads to a *local* method to protect the cooperation principle in open cooperation environments *globally*.

However, the model presented so far is semi-formal only. There is a need to formalize the balance principle, e.g., by extended finite state machines that represent the local user agents. Obligation states would be defined by local obligation attributes. States change by the transmission or receipt of messages. Globally balanced states would be defined formally and theorems about secure progress of cooperation could be proved. This formalism would also help to design and analyse more complex cooperations between groups of persons. So far, only the area of formalisation is marked off: instead of system mechanisms that enforce the cooperation globally, there are local user agents that keep the global balance between actions and proofs.

Environments of application are open societies of autonomous agents, e.g. open economic markets. The balance principle is particularly useful across the boundaries of security domains and might thus help to extend the Clark-Wilson security model [CLWI 87]. Basic telecooperation activities include the negotiation of contracts, the purchase of information, and the reliable and acknowledged transfer of important documents. However, the model is also applicable to more complex cooperations between more than two partners with several states of obligations which refer to different subsets of participants. The principle of balance does always apply in the same way. The model also refines the separation-of-duty cooperation within a closed environment.

Application research is required in order to specify cooperation scenarios such as teleshopping, telebanking and teleadministration. Also, inter-organizational cooperative work is subject to telecooperation. For example, remote computer maintenance and distributed software development are important applications.

Acknowledgements

This paper is based on discussions in the institute of telecooperation technology of the GMD in Darmstadt. Substantial input came from H.J. Burkhardt and E. Raubold. Work on this paper is supported by the project REMO under the umbrella of the German Federal Ministry of Research and Technology (BMFT), which is about a reference model of secure systems. Cooperative partners of REMO are the University of Karlsruhe, GMD, IABG, Siemens, and Teles. I have also learnt from the new ideas of D. Chaum [CHAU 85] about secure communication.

References

- [CHAU 85] D. Chaum: *Security without Identification: Card Computers to Make Big Brother Obsolete*. Communication of the ACM 28(10), 1030-1044, 1985.
- [CLWI 87] D.D. Clark, D.R. Wilson: *A Comparison of Commercial and Military Security Policies*. Proceedings of the 1987 IEEE Symposium on Security and Privacy, Oakland, California. Computer Society Press of the IEEE, Washington D.C., 184-194, 1987.
- [DIHE 76] W. Diffie, M.E. Hellman: *New Directions in Cryptography*. IEEE Transactions on Information Theory, Vol.IT-22, 644-654, 1976.
- [DOBS 89] J.E. Dobson, J.A. McDermid: *Security Models and Enterprise Models*. IFIP 1989. In: Database Security II: Status and Prospects, C.E. Landwehr (Editor), 1-39, Elsevier Science Publishers B.V. (North-Holland), IFIP 1989.
- [ELGA 85] T. ElGamal: *A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms*. IEEE Transactions on Information Theory, Vol.IT-31, 469-472, 1985.
- [PEM 93] *PEM: Privacy Enhancement for Internet Electronic Mail*. Privacy and Security Research Group (PSRG) of the Internet Research Task Force (IRTF) and PEM Working Group of the Internet Engineering Task Force (IETF).
J. Linn: *PEM Part I: Message Encryption and Authentication Procedures*. Proposed Internet Standard Protocol RFC 1421, Feb 1993, 42 pages. Obsoletes RFC 1113.
- [RSA 78] R. Rivest, A. Shamir, L. Adleman: *A Method for Obtaining Digital Signatures and Public Key Cryptosystems*. (RSA.) Communications of the ACM, Volume 21, Number 2, 120-126, Feb 1978.
- [X400 88] CCITT: *Blue Book Volume VIII – Fascicle VIII.7, Data Communication Networks. Message Handling Systems, Recommendations X.400-X.420 (1988)*. Geneva, 1989. In particular: X.411: *Message Transfer System: Abstract Service Definition and Procedures*. (see also: ISO 10021 and ISO 10021-4)
- [X500 88] CCITT: *Blue Book Volume VIII – Fascicle VIII.8, Data Communication Networks: Directory, Recommendations X.500-X.521 (1988)*. Geneva, 1989. In particular: X.509: *The Directory – Authentication Framework*. (see also: ISO 9594 and ISO 9594-8)
- [OSI 84] ISO 7498: *Information Processing Systems – Open Systems Interconnection – Basic Reference Model*. ISO 7498-1984(E). First Edition – 1984-10-15. (Technical Corrigendum 1, ISO/IEC JTC 1, ISO 7498: 1984/Cor.1: 1988(E).) In particular: Part2: *Security Architecture* (ISO 7498-2).

NDU(C): A Mandatory Denial of Service Model

Edward G. Amoroso
Secure Systems Department, AT&T Bell Laboratories
Whippany, N.J. 07981 - USA
Tel: (201) 386 - 6398
ega@neptune.att.com

Abstract

It is argued that security models for denial of service should focus on malicious attack, rather than on the correct provision of service by a computing base in the absence of intruders. A mandatory denial of service model is introduced that focuses on the mitigation of malicious attack. The NDU(C) (no deny up within C) model ensures that low priority subjects never deny services within a distinguished set C of all system services to high priority subjects. NDU(C) is introduced in the context of Millen's resource allocation model (RAM) and several policy instantiations of the model are presented. The model and policy instantiations are assessed with respect to familiar level-oriented model criticisms.

Keywords: Criticality, denial of service, NDU(C), priority, resource allocation model (RAM), security model, security policy.

Introduction

An apparent trend in the development of security models for denial of service is toward expressions of the form: "if request, then grant with respect to some temporal constraint." For example, Millen recently proposed a resource allocation model that allows expression of the finite waiting time (FWT) rule ("if request, then eventually grant") and the maximum waiting time (MWT) rule ("if request then grant before maximum waiting time expires") [Mi92]. A problem with such rules is that they do not allow for justified service denial by an agent with suitable authorization (e.g., an administrator or a higher priority user). Another problem with such rules in the context of computer security is that violations may be caused by circumstances that are unrelated to malicious attack. For instance, natural disasters that cause damage to resources during a pending service request are more often viewed as survivability or availability issues than security issues. Similarly, design errors that cause requests to be delayed are generally viewed as software and system engineering issues more than security issues. If one chooses to characterize these issues as within the purview of security, then one must include other issues such as user-interface design (requests might be delayed if the interface is hard to use), performance (bottlenecks cause delayed service), and many other areas of computer science and system engineering.

In this paper, we describe a denial of service model based on earlier work [Am90] that focuses specifically on preventing users from initiating an action that will cause an authorized request from a higher priority user to be denied or delayed beyond a required target duration. The model is motivated by the level-oriented rules and approach in the Bell-LaPadula model [BL75] and the mandatory integrity portion of the Biba model [Bi77]. While these types of models seem to have fallen into disfavor among researchers (see [Mc87]), they continue to guide the development of many practical secure systems. For

example, some system designers have chosen to employ access controls in conjunction with the Bell-LaPadula model in such a way as to support disclosure protection via disclosure levels and integrity protection via a reverse interpretation of disclosure levels (i.e., high integrity subjects and objects are placed at the lowest disclosure level). Thus, we employ a level-oriented approach and we include an assessment of how the traditional criticisms of the Bell-LaPadula and Biba models apply to our model.

The model is referred to as NDU(C) (pronounced "no deny up within C") and is based on the premise that subjects can be associated with a priority from a lattice of levels (we will generally refer to levels as high and low priorities). A further premise is that objects can be partitioned into critical and non-critical objects. Generalization of this notion to multiple levels of criticality requires only a simple modification to the model. In developing the model, we observed that in traditional system operation, subjects of higher priority should have the option to pre-empt requests by subjects of lower priority if sufficient justification exists. Furthermore, NDU(C) stipulates that operations requested on higher criticality objects may require more urgent attention than operations requested on lower criticality objects. A final premise worth mentioning with respect to the model is that it addresses denial of service attacks by users of the system during operation, rather than during system design and development.

The NDU(C) model stipulates specifically that low priority subjects should never have the ability to cause a service request for an object within a set C of critical objects that is made by a higher priority subject, to be denied. This is illustrated in the diagram in Figure 1 in which circles depict subjects, squares depict objects, lines depict priority boundaries (low priorities at the bottom), dotted arrows depict requests, solid arrows depict denial attempts (unsuccessful denials are shown with a slash), and C is depicted by the dashed box which has no associated priority:

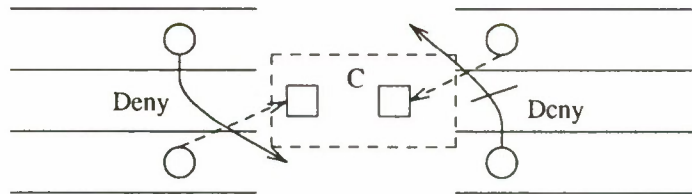


Figure 1. Illustration of NDU(C)

Compliance with NDU(C) requires not only a determination of what constitutes a service denial, but also of what constitutes a critical object. Actually, NDU(C) can be generalized to a global no deny up rule when all objects are viewed as critical. A further insight worth mentioning is that NDU(C) might be viewed as a restricted type of reverse non-interference. That is, whereas high sensitivity subjects should not (among other things) affect service requests made by low sensitivity subjects in systems that meet non-interference, low priority subjects should not affect service requests made by high priority subjects in systems that meet NDU(C).

The purpose of this paper is to introduce the NDU(C) model in the context of Millen's resource allocation model (RAM) so that it can be combined and compared with other rules such as FWT and MWT that are expressible in the context of Millen's RAM. Three different policy interpretations of the model are presented that address different types of protections. The first prevents single users from having resource requirements that could cause denial of service, the second ensures that the system allocate resources to users in a manner that avoids denial of service (even if users request resources in

such a manner that would cause such problems), and the third ensures that no group of N low users can work in collusion to cause a denial of service attack. The paper also outlines how the model deals with many of the traditional level-oriented security model criticisms that have been reported in recent years. Specifically, criticisms relating to the bi-direction of information flow caused by remote reads, the requirement for trusted process support, and the problems induced by the System Z scenario [McL87] are addressed.

Millen's RAM

Since the NDU(C) model will be expressed using Millen's resource allocation model (RAM), we briefly describe the RAM here. Readers familiar with Millen's work might skip to Section 3. The RAM allows one to specify denial of service models and policies in terms of the detailed resource allocations that comprise the provision of service to users in computing systems. It is based on the notion that subjects have certain space and time requirements for resources in order to proceed in a desired task. Service denials occur when the space and time allocations for some process do not meet its requirements. Millen shows that policies such as finite waiting time (FWT) and maximum waiting time (MWT) can be easily specified in the context of his RAM. He does not, however, include subject privileges and object criticalities as part of his proposed framework (we add these in Section 3).

The RAM consists specifically of a collection of rules that characterize a family of computing systems that is well-suited to meeting denial of service constraints. That is, the rules are designed to introduce concepts to this family that will greatly assist in the specification and analysis of denial of service models and policies. The RAM and its associated rules are presented below.

A set P of active processes and a set R of passive resource types are assumed. Some fixed constraint c denotes the collective maximum number of units of all resource types available on the system being examined. An allocation vector A_p denotes the number of units of each resource that are allocated to process p in some state. In this way, an allocation vector can be viewed as a snapshot of the resources allocated to a process at some instant. A special type of resource known as the CPU resource is used to model whether a process is running or asleep. Specifically, whenever $A_p(\text{CPU}) = 1$, we say that $\text{running}(p)$ is true and whenever $A_p(\text{CPU}) = 0$, we say that $\text{asleep}(p)$ is true.

A space requirements vector sQ_p denotes the number of units of each resource that process p requires to proceed in its desired task in some state. It is assumed that processes can identify the set of resources necessary to complete a task before they initiate that task. A function $T(p)$ denotes the last time the clock for process p was updated to reflect a real clock. A time requirements vector tQ_p denotes the amount of time that process p requires for each resource to complete its present task. Just as with space requirements, it is assumed that a process can determine its time requirements for a particular task. Additional details on these notions can be found in Millen's original exposition [Mi92].

The eight rules that comprise Millen's RAM are listed below. Each rule is intended to constrain the family of systems that are consistent with the model. Ticked variables (e.g., $\text{running}(p)'$) are intended to denote the value of a variable after a single state transition. Rule R1 stipulates that the sum of allocated resource units to all processes in P must be less than the system constraint c . Millen refers to models and policies that violate this rule as infeasible.

$$(R1) \sum_{p \in P} A_p \leq c$$

Rule R2 states that running processes must have zero space requirements. Millen reasons that if a process does not have all of the resources it desires in some state, then it makes little sense for that process to proceed. Starvation occurs when a process has non-zero space requirements that are never met (or are met late).

$$(R2) \text{ if running}(p) \text{ then } {}^S Q_p = 0$$

Rule R3 states that resource allocations are not changed for running processes. This is a powerful assumption because it implies that running processes will not be preempted during their operation as a result of some resource reallocation (other than reallocation of the CPU resource). The construction "running(p) and running(p)'" is intended to stipulate that in some state a process p is running and in the next state it remains running.

$$(R3) \text{ if running}(p) \text{ and running}(p)' \text{ then } A_p' = A_p$$

Rule R4 states that process clocks are never updated when CPU allocation is not changed. The units of time are assumed to be positive integers that always increase when time is updated.

$$(R4) \text{ if } A_p(\text{CPU})' = A_p(\text{CPU}) \text{ then } T(p)' = T(p)$$

Rule R5 states that clocks are updated when CPU allocation changes and this update always reflects an increase in time. Notice that each process has its own clock and no provision is made to ensure that different clocks are synchronized to each other or to some real time clock.

$$(R5) \text{ if } A_p(\text{CPU})' \neq A_p(\text{CPU}) \text{ then } T(p)' > T(p)$$

Rule R6 states that space requirements are adjusted for sleeping processes. In other words, when a process is asleep, it must determine the resources that will be required in order to make progress in some task. Once all of these resources are obtained, the process wakes up. This notion of meeting space requirements before initiation of a task will provide a framework for expressing the NDU(C) denial of service model.

$$(R6) \text{ if asleep}(p) \text{ then } {}^S Q_p' = {}^S Q_p + A_p - A_p'$$

Rule R7 states that time requirements are not adjusted for sleeping processes. Instead, time requirements are adjusted when a process is actively utilizing a resource.

$$(R7) \text{ if asleep}(p) \text{ then } {}^T Q_p' = {}^T Q_p$$

Rule R8 states that transitions that put processes to sleep reallocate only CPU resources. Space allocation changes must occur only after a process is asleep.

$$(R8) \text{ if running}(p) \text{ and asleep}(p)' \text{ then } A_p' = A_p - \text{CPU}$$

Millen uses the above RAM as a means for specifying certain policies. For example, the finite waiting time (FWT) policy can be expressed in the context of the RAM. We use the *leads-to* operator of temporal logic (i.e., $A \text{ leads_to } B$ means henceforth A implies eventually B) to specify intervals that

may result from multiple transitions.

$$FWT: \forall p, S: \exists S': S'(\text{running}(p)) \text{ and } S \text{ leads_to } S'$$

In the above expression, $S(x)$ means x is true in state S and $S \text{ leads_to } S'$ means $\forall S, S': S((T(p) = n))$ and $S'((T(p) = m))$ and $m > n$. FWT states that users will eventually receive requested resources (i.e., they will eventually receive the CPU to make progress). Maximum waiting time (MWT) can be expressed similarly.

$$MWT: \exists B: \forall p, S: \exists S': S'(\text{running}(p)) \text{ and } S \text{ leads_to}(b) S'$$

In this expression, $S \text{ leads_to}(b) S'$ means that $\forall S, S': S((T(p)) = n)$ and $S'((T(p) = m))$ and $m - n \leq b$. MWT differs from FWT in that an explicit time limit is imposed on how long users must wait to make progress in their task.

Specifying NDU(C) Using Millen's RAM

As suggested above, we would like to introduce process privileges and object criticality as a means for expressing the NDU(C) mandatory denial of service model. These will be assigned in a manner analogous to the assignment of clearances and classifications for disclosure. Thus, new functions on processes and resources are introduced as follows:

$$\begin{aligned} \pi: P &\rightarrow N \\ \chi: R &\rightarrow \text{boolean} \end{aligned}$$

The value $\pi(p)$ is intended to denote the natural-valued privilege of process p . If $\pi(p_1)$ is greater than $\pi(p_2)$, then we say that p_1 has a greater privilege than p_2 . Thus, an ordering is imposed on the set of privileges. Similarly, the value $\chi(r)$ is intended to denote the criticality of a resource. If the value $\chi(r)$ is true, then we say that r is a critical resource. If the value $\chi(r)$ is false, then we say that r is a non-critical resource.

Tranquility rules must now be added to ensure that privileges and criticalities are not changed in inappropriate manners. We choose to specify strong tranquility for greater assurance, but weak tranquility could suffice. Rule R9 states that a process privilege is always the same from one state to another. Therefore, process privileges would have to be established in an initial state.

$$(R9) \quad \forall p: \pi(p) = \pi(p)'$$

Rule R10 states that resource criticalities also do not change and would have to be established in an initial state.

$$(R10) \quad \forall r: \chi(r) = \chi(r)'$$

As we will show, the use of privileges and criticalities supports our goal of relaxing policies such as MWT and FWT so that processes with higher privilege can deny critical resources to processes with a lower privilege under an appropriate set of circumstances. Specifically, if higher privilege processes require the use of critical resources that are needed by lower privilege processes, then under such

prioritized schemes, the allocation of these critical resources would not be made to the lower privilege processes. This should follow one's intuition regarding denial of service on real systems. If a system administrator, for example, chooses to deny service to some normal user, then one presumes that this decision is made for the overall good of the system. To classify such an occurrence as a denial of service is misleading.

Enforcing a denial of service requirement with respect to only a set of critical resources is analogous to the enforcement of other policies to an isolated set of resources. For example, on a secure system, denial of service requirements might only apply to the resources associated with a TCB. By restricting the domain of applicability for denial of service requirements, we increase their implementability.

Given these concepts, we can express NDU(C) in terms of the space requirements for the various processes on the system. The basic idea is that processes at some priority level should not interfere with the space requirements of higher priority processes. This is expressed in the context of Millen's RAM with our proposed priority and criticality enhancements. As a shorthand concept to assist in the presentation of NDU(C), we define the set of processes that have priority greater than the priority of some process p (denoted $p.\uparrow$) as follows:

$$p' \in p.\uparrow \text{ iff } \pi(p) < \pi(p')$$

We choose to characterize the model as three separate policy instantiations, any one of which might be selected for a particular implementation. These three instantiations are presented below:

Single User Requirements (SUR) Policy: The first policy instantiation specifies that single users are prevented from requiring space in a way that would allow them to solely deny service to high priority users.

$$SUR: \forall p: [\chi[c] - \chi[{}^S Q_p] \geq \sum_{p' \in p.\uparrow} \chi[{}^S Q_{p'}]]$$

In the expression, $\chi[{}^S Q_p]$ denotes the space requirements of process p for all critical resources and $\chi[c]$ denotes the total amount of critical resource on the system. This policy could be enforced by system restrictions on user space requirements or by user agreements to advertise space requirements that respect the policy.

Single User Allocation (SUA) Policy: The second policy instantiation specifies that a single user cannot be allocated enough resource to ever solely deny service to a higher priority user, regardless of what the single user establishes as space requirements.

$$SUA: \forall p: [\chi[c] - \chi[A_p] \geq \sum_{p' \in p.\uparrow} \chi[{}^S Q_{p'}]]$$

This policy would be primarily enforced by the system ensuring that allocation is performed commensurate with the desired condition. Note that the SUR policy implies the SUA policy, but that the reverse is not true.

Multi-User Allocation (MUA) Policy: The third policy instantiation specifies that no N different users can be allocated services in a way that could deny service to any user with priority higher than the maximum priority of the N users.

$$MUA: \forall p_1, p_2, \dots, p_n: \pi(p_1) \leq \pi(p_2) \leq \dots \leq \pi(p_n): [\chi[c] - \sum_{p_i} \chi[A_{p_i}] \geq \sum_{p' \in p_i, \uparrow} \chi[S_{Q_{p'}}]]$$

An important issue highlighted in each policy instantiation of the NDU(C) model is the emphasis on avoiding low privilege interference in high privilege requests to critical resources. This does not ensure that high privilege requests will be granted (as in the FWT and MWT policies). It merely precludes one type of occurrence that could cause such requests to not be granted. If a given system must exhibit certain availability attributes for critical resources, then a different model must be selected (e.g., FWT or MWT). We view this strict attention to the avoidance of low level interference as a desirable aspect of the NDU(C) model because it formally represents our contention that the security community should be emphasizing security issues over survivability, fault tolerance, reliability, availability, and other system engineering concerns. Previous work in the area of denial of service has not made this distinction.

Concluding Remarks

Since NDU(C) is a level-oriented model in which mandatory decisions are made based on a comparison of different leveled attributes, traditional criticisms of such approaches must be examined. For example, McLean [Mc87] suggests that traditional level-oriented policies such as the Bell-LaPadula model must include provision for secure transitions as well as secure states. That is, one cannot simply reason inductively on a set of states in order to demonstrate that a system is secure. Specifically, he suggests that one cannot simply rely on demonstration of the *-property and the ss-property in each state of a system's behaviors because it is possible that a process could be downgraded or upgraded as needed to ensure that any access is always granted. Bell [Be88] retorts that tranquility deals acceptably with this problem since it ensures that security attributes such as clearances and classifications either cannot change (strong tranquility) or can only change subject to an explicit set of rules (weak tranquility). Since NDU(C) assumes tranquility rules (R9 and R10), we conclude that upgrading and downgrading as in System Z will not cause violations.

Another problem related to bi-directional information flow stems from the fact that when a read request is made by a higher trusted process to a less trusted process, the actual request constitutes a write down, which is not allowed in the Bell-LaPadula model. This problem is particularly evident in distributed systems and similar problems exist in the Biba mandatory integrity policy. NDU(C), however, does not exhibit this problem since it is not information flow-oriented. That is, bi-directional information flow is not a problem in an NDU(C)-compliant system because denial operations are the focus, rather than direction of information flow.

A third problem with level-oriented models that we will mention is that they generally work around trusted processes, rather than within them. That is, device drivers and resource handlers that must be kept secure only benefit from level oriented models in that they are protected from less trusted processes. Within the set of trusted processes, most level-oriented models are not much help. Unfortunately, the NDU(C) exhibits this drawback because denial of service between trusted processes (or between any set of processes with the same privilege) is not dealt with in the model. If one requires that trusted processes avoid denial of service threats, then the FWT or MWT policies might be more suitable.

A final problem worth mentioning is related to user agreements. The notion of user agreements in denial of service models essentially states that users must make reasonable requests for services in order to be granted a request. For example, if a user generates an infeasible space requirement, then it will be

impossible for a system to grant that request. While the NDU(C) model avoids the traditional user agreement problem (NDU(C) does not stipulate that requests be granted eventually or within a bounded interval), it does allow high privilege users to maintain constantly high space or time requirements that would ensure the starvation of lower privilege processes. As a result, the NDU(C) model does introduce potential system vulnerabilities in this area.

One suggested research direction for Millen's RAM involves an investigation of the implications of relaxing Rule R3. This rule prevents a malicious intruder from stealing resources allocated to a running process. This precludes a great many denial of service attacks. By relaxing the rule, one can investigate the conditions under which attacks on executing processes can occur.

References

- [Am90] E.G. Amoroso, "A policy model for denial of service", *Proc. Computer Security Foundations Workshop III*, Franconia, N.H., 1990.
- [BK91] E.M. Bacic and M. Kuchta, "Considerations in the preparation of a set of availability criteria", *Proc. Third Annual Canadian Computer Security Symp.*, Ottawa, Canada, 1991.
- [Be88] D. Bell, "Concerning 'modeling' of computer security", *Proc. IEEE Symp. Security and Privacy*, 1988.
- [BL75] D. Bell and L. LaPadula, "Secure Computer Systems: Unified Exposition and Multics Interpretation," ESD-TR-75-306, Mitre, 1975.
- [Bi77] K. Biba, "Integrity considerations for secure computer systems," Mitre TR-3153, 1977.
- [Gl83] V.D. Gligor, "A note on the denial of service problem", *Proc. IEEE Symp. Research in Security and Privacy*, Oakland, Ca., 1983.
- [Mc87] D. McCullough, "Specifications for multi-level security and a hook-up property", *Proc. IEEE Symp. Security and Privacy*, 1987.
- [Mc87] J. McLean, "Reasoning about security models", *Proc. IEEE Symp. Security and Privacy*, 1987.
- [Mi92] J.K. Millen, "A resource allocation model for denial of service", *Proc. IEEE Symp. Research in Security and Privacy*, Oakland, Ca., 1988.
- [YG88] Y.S. Yu and V.D. Gligor, "A formal specification and verification method for the prevention of denial of service", *Proc. IEEE Symp. Research in Security and Privacy*, Oakland, Ca., 1988.

REFERENTIAL INTEGRITY IN MULTILEVEL SECURE DATABASES

Ravi S. Sandhu and Sushil Jajodia¹

Center for Secure Information Systems

&

Department of Information and Software Systems Engineering

George Mason University, Fairfax, VA 22030-4444

Abstract This paper studies referential integrity in multilevel relations with element-level labeling. Our principal contribution is resolution of an impasse left by previous work in this area. We show that the previous work leaves us with a choice of either accepting referential ambiguity, or severely curtailing the modeling power of multilevel relations. We then show how to escape this impasse by eliminating entity polyinstantiation, while retaining element polyinstantiation (as an option). We also discuss how entity polyinstantiation can be securely eliminated.

Keywords: multilevel secure databases, referential integrity, polyinstantiation

1 INTRODUCTION

Referential integrity is an important component of the classical relational data model [4]. It is concerned with references from one relation to another. The principal motivation for referential integrity is to prevent dangling references across relations, such as when an employee is assigned to a non-existent department. Consideration of referential integrity in multilevel relations leads to the realization that it can result in signaling channels for leakage of secret information [3, 6, 7]. A multilevel secure relational model must cope with the possibility of these channels.

The central point of this paper is that prior work on referential integrity has left us with a choice of two undesirable alternatives. We either have referential ambiguity, which results in confusion about the meaning of data in relations; or we have serious limitations on the expressive power of multilevel relations, such as the inability to classify a relationship between unclassified entities.

Our principal contribution in this paper is to show how this unacceptable impasse can be resolved by building upon the distinction between entity and element polyinstantiation. We argue that entity polyinstantiation is so contrary to referential integrity that it must be eliminated. We also demonstrate how entity polyinstantiation can be easily prevented, by means of the usual integrity constraints in Database Management Systems. On the other hand element polyinstantiation can be tolerated if it is required for purpose of cover stories, or some similar reason. In other words, element polyinstantiation can be available as an option as needed; whereas entity polyinstantiation should be eliminated in the data model. (Note that element polyinstantiation can be securely prevented using the technique of [20], if it is not needed in a particular application.)

The paper is organized as follows. Section 2 defines a model for multilevel relations with element level labeling. In this section only individual relations are considered. Section 3 discusses the semantics of polyinstantiation, including the important distinction between entity and element polyinstantiation. Some of the more subtle aspects of the definitions of section 2 are also discussed. Section 4 reviews prior work on referential integrity in multilevel relations, which leaves us in the impasse mentioned above. Section 5 describes how to resolve this impasse by eliminating entity polyinstantiation. Section 6 concludes the paper.

¹This work was partially supported by the U.S. Air Force, Rome Laboratory under the contract # F30602-92-C-0002. We are indebted to Joe Giordano for his support and encouragement which made this work possible.

2 MULTILEVEL RELATIONAL MODEL

In this section, we give the basic definitions and assumptions used with multilevel relations. Our initial focus is on individual relations considered in isolation. Consideration of referential integrity, which involves two relations, is deferred until sections 4 and 5. The definitions and properties for multilevel relations given here are conceptually simpler, and different in important ways, as compared to previous work on element-level labeling [6, 11, 12, 13, 15, 16, 17, 19, 20]. The most significant difference is the requirement that there can be at most one tuple in each access class for a given entity. This gives us the simplicity of tuple-level labeling, combined with the flexibility of element-level labeling. There are also some other subtle differences in the precise formulation of various properties.

The reader is assumed to be familiar with basic concepts of relational database theory. In analogy to the usual definition of a relation, a *multilevel relation* consists of the following two parts.

Definition 1 [RELATION SCHEME] A state-invariant multilevel relation scheme which is denoted by $R(A_1, C_1, A_2, C_2, \dots, A_n, C_n, TC)$, where each A_i is a *data attribute*² over domain D_i , each C_i is a *classification attribute* for A_i , and TC is the *tuple-class* attribute. The domain of C_i is specified by a range $[L_i, H_i]$, $H_i \geq L_i$, which defines a sub-lattice of access classes ranging from L_i up to H_i . \square

Definition 2 [RELATION INSTANCES] A collection of state-dependent *relation instances*, each of which is denoted by $R_c(A_1, C_1, A_2, C_2, \dots, A_n, C_n, TC)$; one for each access class c in the given lattice. Each instance is a set of distinct tuples of the form $(a_1, c_1, a_2, c_2, \dots, a_n, c_n, tc)$ where each $a_i \in D_i$ and $c_i \in [L_i, H_i]$, or $a_i = \text{null}$ and $c_i \leq H_i$; and $tc \geq \text{lub}\{c_i : i = 1 \dots n\}$.³ Note that c_i must be defined even if a_i is null, i.e., a classification attribute cannot be null. \square

We assume that there is a user-specified *apparent primary key* AK consisting of a subset of the data attributes A_i . In general AK will consist of multiple attributes. We also assume that the relation scheme is itself unclassified (or, more generally, classified at the greatest lower bound of L_i , $i = 1 \dots n$). A tuple whose tuple class is c is said to be a c tuple. (Similarly, a subject whose clearance is c is said to be a c subject.)

We now list four integrity requirements which we feel must be satisfied by all multilevel relations. We call these the *core integrity properties*. We use the notation $t[A_i]$ to mean the value corresponding to the attribute A_i in tuple t , and similarly for $t[C_i]$ and $t[TC]$.

Property 1 [Entity Integrity] Let AK be the apparent primary key of R . A multilevel relation R satisfies entity integrity if and only if for all instances R_c and $t \in R_c$

1. $A_i \in AK \Rightarrow t[A_i] \neq \text{null}$,
2. $A_i, A_j \in AK \Rightarrow t[C_i] = t[C_j]$ (i.e., AK is uniformly classified), and
3. $A_i \notin AK \Rightarrow t[C_i] \geq t[C_{AK}]$ (where C_{AK} is defined to be the classification of the apparent primary key). \square

²In many cases it is useful to have an A_i represent a collection of *uniformly classified* data attributes. This extension requires straightforward modifications to our statements in this paper, which are all formulated in terms of the A_i 's being individual data attributes.

³Note that in previous work [6, 11, 12, 13, 15, 16, 17, 19, 20] it has generally been required that $tc = \text{lub}\{c_i : i = 1 \dots n\}$. The main reason for relaxing this requirement to $tc \geq \text{lub}\{c_i : i = 1 \dots n\}$ is to allow a c -subject to specify the classification of individual attributes in a c -tuple. For example, let M_1 and M_2 be incomparable labels whose least upper bound is S and greatest lower bound is U . We should have some means of allowing a S -subject to instantiate a S tuple whose individual classification attributes are at, say, U , M_1 , and M_2 . Careful consideration of the update semantics in such situations, leads to the conclusion that a S -subject should be able to instantiate a S tuple, even if the least upper bound of the individual classification attributes turns out to be less than S .

The first requirement is exactly the definition of entity integrity from the standard relational model, and ensures that no tuple in R_c has a null value for any attribute in AK . The second requirement says that all attributes in AK have the same classification in a tuple. This will ensure that AK is either entirely visible, or entirely null at a specific access class c . The final requirement states that in any tuple the class of the non- AK attributes must dominate C_{AK} . This rules out the possibility of associating non-null attributes with a null primary key. Property 1 is identical to the entity integrity property of SeaView [17].

Notation. In order to simplify our notation, we will henceforth use A_1 as synonymous to AK , i.e., A_1 and AK both denote the apparent primary key.

The next property is concerned with consistency between relation instances at different access classes. It requires that at every access class c , exactly those tuples whose access class is dominated by c are visible.

Property 2 [Inter-Instance Integrity] A multilevel relation R satisfies the inter-instance integrity property if and only if for all $c' \leq c$ we have $R_{c'} = \{t \in R_c \mid t[TC] \leq c'\}$. \square

Thus, for example, a TS-subject will see the entire relation given in figure 1, while a C-subject will see the filtered instance given in figure 2. Let us denote the relation between $R_{c'}$ and R_c described in property 2 by $R_{c'} = \sigma(R_c, c')$, where σ is called the *filter function*. It is evident that $\sigma(R_c, c) = R_c$, and $\sigma(\sigma(R_c, c'), c'') = \sigma(R_c, c'')$ for $c \geq c' \geq c''$; as one would expect from the intuitive notion of filtering.

The formulation of filtering given here is simpler than the definition given in [11, 13, 15] (and subsequently adopted by SeaView [17]). The main difference is that the null-subsumption property of [11, 13, 15] is no longer being required (principally because the null-integrity property of [11, 13, 15] has been dropped). In the formulation given here null values require no special treatment from a security viewpoint.

An important consequence of the inter-instance integrity property is that it allows instances such as shown in figure 3. Note that there is a C tuple whose key class is U, but the key value (and class) do not occur in any U tuple. U subjects will see an empty relation in this case, as indicated in figure 4. We will see in section 5 that this phenomenon has significant, and beneficial, implications for referential integrity. Contrast figure 3 with the instance shown in figure 5 (with the Unclassified view shown in figure 6). With our definition of inter-instance integrity both figures 3 and 5 are valid Confidential instances of SOD, but they are semantically different.⁴ We will return to consideration of this issue in section 5.

Next, we have the following polyinstantiation integrity constraint which prohibits polyinstantiation within a single access class.

Property 3 [Polyinstantiation Integrity (PI)] A multilevel relation R is said to satisfy polyinstantiation integrity (PI) if and only if for every R_c we have for all A_i that $A_1, C_1, C_i \rightarrow A_i$. \square

This property stipulates that the user-specified apparent key A_1 , in conjunction with the classification attributes C_1 and C_i , functionally determines the value of the attribute A_i . In other words the real primary key of the relation is $A_1, C_1, C_2, \dots, C_n$. This formulation of PI was first proposed in [11].⁵ The effect of polyinstantiation integrity is to rule out instances such in figure 7, where there are two values labeled U for the Objective attribute of the Enterprise.

⁴Note that with prior definitions of inter-instance integrity [11], which include null-subsumption, the closest one can get to these instances is to have the C instance of figure 3 with corresponding U instance of figure 6.

⁵It should be noted that the SeaView definition of polyinstantiation integrity [16, 17] requires property 3, but in addition requires a multi-valued dependency property which has the undesirable consequence of introducing spurious tuples in the multilevel relation [11].

SHIP		OBJ		DEST		TC
Enterprise	U	Exploration	U	Talos	U	U
Enterprise	U	Mining	C	Sirius	C	C
Enterprise	U	Spying	S	Rigel	S	S
Enterprise	U	Coup	TS	Orion	TS	TS

Figure 1: A multilevel relation SOD

SHIP		OBJ		DEST		TC
Enterprise	U	Exploration	U	Talos	U	U
Enterprise	U	Mining	C	Sirius	C	C

Figure 2: Confidential view of figure 1

SHIP		OBJ		DEST		TC
Enterprise	U	Mining	C	Sirius	C	C

Figure 3: Another Confidential Instance of SOD

SHIP	OBJ	DEST	TC

Figure 4: Unclassified view of figure 3

SHIP		OBJ		DEST		TC
Enterprise	U	null	U	null	U	U
Enterprise	U	Mining	C	Sirius	C	C

Figure 5: Yet Another Confidential Instance of SOD

SHIP		OBJ		DEST		TC
Enterprise	U	null	U	null	U	U

Figure 6: Unclassified view of figure 5

Starship		Objective		Destination		TC
Enterprise	U	Exploration	U	Talos	U	U
Enterprise	U	Spying	U	Rigel	S	S

Figure 7: Violation of Polyinstantiation Integrity

Finally, we introduce the fourth integrity property, which was first identified in [19]. The intuitive idea is that every entity in a relation can have at most one tuple for every access class.⁶ The requirement is formally as follows.

Property 4 [PI-tuple-class] R satisfies tuple-class polyinstantiation integrity if and only if for every instance R_c , $(\forall A_i \notin A_1)[A_1, C_1, TC \rightarrow A_i]$. \square

To appreciate the motivation for PI-tuple-class consider the instance SOD_U given in figure 8. Let Starship be the apparent key of this relation. Eight instances of SOD_S are shown in figure 9. All these instances of SOD_S are consistent with SOD_U of figure 8 with respect to the inter-instance integrity property. In other words, if tuples with $TC = S$ are removed from any of the eight SOD_S instances we are left with the single tuple of the SOD_U instance. Moreover, all eight instances of SOD_S satisfy the entity integrity and polyinstantiation integrity properties. Thus any of these eight instances are acceptable under properties 1, 2 and 3.

It is clear that instances 2, 3 and 4 of figure 9 have a much simpler interpretation than instances 5, 6, 7 and 8. The PI-tuple-class property formalizes this intuitive distinction by requiring that there be at most one tuple for the Enterprise at each access class. Instances 2, 3 and 4 have exactly one S tuple for the $\langle \text{Enterprise}, U \rangle$, in addition to the single U tuple. The U tuple is then easily interpreted to denote a cover story with respect to the S tuple. Instances 5, 6, 7 and 8 are in violation of PI-tuple-class because they all have two or more tuples with tuple class S which have the same apparent key and key class (i.e., $\langle \text{Enterprise}, U \rangle$).

Polyinstantiation integrity (or PI) and PI-tuple-class are independent properties. Instances 5, 6, 7 and 8 of figure 9 illustrate relation instances which satisfy PI but not PI-tuple-class. The instance of SOD_S given in figure 10 shows how PI-tuple-class can be satisfied while PI is violated.

We regard properties 3 and 4 as the formal definition of the informal notion of A_1 as the user-specified apparent primary key. Note that for single level relations C_1 and C_i will be equal to the same constant value in all tuples. In this case property 3 amounts to saying $A_1 \rightarrow A_i$, which is precisely the definition of primary key in standard relational theory. Similarly, property 4 also reduces to $A_1 \rightarrow A_i$ for single-level relations.

3 SEMANTICS OF POLYINSTANTIATION

In the previous section we have given a formal model (albeit without referential integrity) for multi-level relations with element-level labeling. In this section we consider the semantic interpretation of polyinstantiation in these relations. The essential points can be illustrated in context of the instance of figure 11. This instance is permitted by the integrity properties of section 2. It exhibits two distinct forms of polyinstantiation which we call *entity polyinstantiation* and *element polyinstantiation*.

Entity polyinstantiation arises when there are two tuples with the same value of the apparent primary key, but with different values of the key class. This is illustrated in figure 11 where the third tuple has the same apparent key value (i.e., Enterprise) as the first (or second) tuple, but the key class in the third tuple (i.e., S) is different from the key class in the first (or second) tuple (i.e., U). The interpretation is that in this case there are two Starships, the $\langle \text{Enterprise}, U \rangle$ and the $\langle \text{Enterprise}, S \rangle$. In other words the two S-tuples pertain to two distinct real world entities. In contrast, the top two tuples in figure 11 refer to the same starship $\langle \text{Enterprise}, U \rangle$; the S-tuple gives the classified values for the Objective and Destination attributes, whereas the U-tuple gives the unclassified cover story for both attributes. The S-tuple for $\langle \text{Enterprise}, S \rangle$ pertains to a

⁶ The formulation of this property in [19] disclosed some problems with this intuitive idea, which have been carefully avoided in the present paper. We also note that the behavior of multilevel relations in LDV [10] essentially requires this property, although the precise formalization and detailed semantics are somewhat different.

Starship		Objective		Destination		TC
Enterprise	U	Exploration	U	Talos	U	U

Figure 8: An instance SOD_U

No.	Starship		Objective		Destination		TC
1	Enterprise	U	Exploration	U	Talos	U	U
2	Enterprise	U	Exploration	U	Talos	U	U
	Enterprise	U	Spying	S	Talos	U	S
3	Enterprise	U	Exploration	U	Talos	U	U
	Enterprise	U	Exploration	U	Rigel	S	S
4	Enterprise	U	Exploration	U	Talos	U	U
	Enterprise	U	Spying	S	Rigel	S	S
5	Enterprise	U	Exploration	U	Talos	U	U
	Enterprise	U	Exploration	U	Rigel	S	S
	Enterprise	U	Spying	S	Rigel	S	S
6	Enterprise	U	Exploration	U	Talos	U	U
	Enterprise	U	Spying	S	Talos	U	S
	Enterprise	U	Spying	S	Rigel	S	S
7	Enterprise	U	Exploration	U	Talos	U	U
	Enterprise	U	Spying	S	Talos	U	S
	Enterprise	U	Exploration	U	Rigel	S	S
8	Enterprise	U	Exploration	U	Talos	U	U
	Enterprise	U	Spying	S	Talos	U	S
	Enterprise	U	Exploration	U	Rigel	S	S
	Enterprise	U	Spying	S	Rigel	S	S

Figure 9: Eight instances of SOD_S

Starship		Objective		Destination		TC
Enterprise	U	Exploration	U	Talos	U	U
Enterprise	U	Spying	U	Rigel	S	S

Figure 10: An instance of SOD_S satisfying PI-tuple-class but not PI

Starship		Objective		Destination		TC
Enterprise	U	Exploration	U	Talos	U	U
Enterprise	U	Spying	S	Rigel	S	S
Enterprise	S	Attack	S	Sirius	S	S

Figure 11: Entity and Element Polyinstantiation

Starship		Objective		Destination		TC
Enterprise	U	null	U	null	U	U
Enterprise	U	Spying	S	Rigel	S	S

Figure 12: An S instance of SOD

Starship		Objective		Destination		TC
Enterprise	U	null	U	null	U	U

Figure 13: The U view of figure 12

Starship		Objective		Destination		TC
Enterprise	U	Spying	S	Rigel	S	S

Figure 14: Another S instance of SOD

Starship		Objective		Destination		TC

Figure 15: The U view of figure 14

completely different Starship whose existence is not known at the unclassified level. In short, entity polyinstantiation is interpreted by asserting that a real-world entity is identified in the database by the apparent key and key class.

Element polyinstantiation, on the other hand, arises when there are two tuples with the same value of the apparent primary key, and with the same value of the key class. This is illustrated in figure 11 by the first two tuples. The interpretation, in this case, is that both tuples refer to the same Starship in the real world, viz., the $\langle \text{Enterprise}, U \rangle$. The U-tuple gives the unclassified values for the Objective and Destination attributes, whereas the S-tuple gives the classified values for these attributes. In short, element polyinstantiation is interpreted by asserting that the same real-world entity has different values for its attributes at different access classes.

Figures 12 through 15 further illustrate a subtle aspect of the inter-instance property, briefly alluded to in the previous section. Figure 12 shows element polyinstantiation for a single Starship called Enterprise, whose key class is U. Even though the values of the Objective and Destination attributes in the U tuple are null, we will consider this to be element polyinstantiation because non-null values have been given in the S tuple. The corresponding U instance is shown in figure 13. Now consider the S instance of SOD shown in figure 14. This instance is allowed by the integrity properties of the previous section. The corresponding U instance is shown in figure 15. Note that even though the S tuple of figure 14 has a component labeled U, the U instance is completely empty.

What interpretation are we to give to the fact that the Starship name is labeled U in figure 14? We will understand such a situation to mean that the Enterprise may become visible at the U level, even though currently it is not. The implication is that if a U tuple for the $\langle \text{Enterprise}, U \rangle$ does come about in SOD, it is going to refer to exactly the same real-world starship that the existing S tuple refers to.

We will see, in section 5, that this interpretation turns out—rather unexpectedly—to be important for certain aspects of referential integrity. It should be kept in mind that, if the semantics of the application dictate that the instance of figure 14 is not allowed we can prevent its occurrence by the usual integrity constraints in relational systems. The point is that our data model does not

inherently rule out this instance, as is done by previous data models [6, 11, 12, 13, 15, 16, 17, 19, 20] in this area.

4 PRIOR WORK ON REFERENTIAL INTEGRITY

In this section we review previous work on referential integrity and point out its weaknesses. The notion of a foreign key relates two relations: a *referencing* relation, say R , and a *referenced* relation, say Q . A foreign key FK of R is declared to be one or more attributes of R which collectively reference the primary key PK of Q . The number of attributes in FK and PK , as well as their domains (such as number or character string), must be identical for a valid declaration of a foreign key.

The first requirement for foreign keys is as follows.

Property 5 [Foreign Key Integrity] Let FK be a foreign key of the referencing relation R . A multilevel relation R satisfies foreign key integrity if and only if for all instances R_c and $t \in R_c$

1. Either $(\forall A_i \in FK)[t[A_i] = \text{null}]$ or $(\forall A_i \in FK)[t[A_i] \neq \text{null}]$.
2. $A_i, A_j \in FK \Rightarrow t[C_i] = t[C_j]$ (i.e., FK is uniformly classified). □

The first part of this property arises from standard relations. The motivations for the second part of this property are similar to those for the uniform classification of apparent primary keys in the entity integrity property.

The foreign key property by itself is not sufficient. In standard relations, the referential integrity property precludes the possibility of dangling references from R to Q . In other words a non-null foreign key must have a matching tuple in the referenced relation. To avoid signaling channels that arise due to upward (or sideways) references, SeaView originally proposed the following formulation of referential integrity for multilevel relations [6].

Property 6 [Referential Integrity (SeaView I)] Let FK be a foreign key of the referencing relation R . Let Q be the referenced relation, with apparent primary key AK . R and Q satisfy referential integrity if and only if for all instances R_c and Q_c occurring together, and for all $t \in R_c$ such that $t[FK] \neq \text{null}$, there exists $q \in Q_c$ such that $t[FK] = q[FK] \wedge t[CFK] \geq q[CAK]$. □

Unfortunately, the above formulation results in referential ambiguity. The problem of referential ambiguity was first noted by Gajnak [9]. It is illustrated in figures 16(a), where SOD is as before, and CAPTAIN is the apparent primary key of the CS relation. In this example SHIP is a foreign key from CS to SOD. In the CS relation, at the U level Kirk has not been assigned to any starship, while at the S level Kirk's assignment is to the Enterprise. However, due to entity polyinstantiation, there are two starships called Enterprise in SOD. It is therefore ambiguous as to which one Kirk is assigned to (or perhaps he is captain of both).

Gajnak's observations led SeaView researchers to modify the above referential integrity property to require equality of the key classifications [16, 17], as follows.

Property 7 [Referential Integrity (SeaView II)] Let FK be a foreign key of the referencing relation R . Let Q be the referenced relation, with apparent primary key AK . R and Q satisfy referential integrity if and only if for all instances R_c and Q_c occurring together, and for all $t \in R_c$ such that $t[FK] \neq \text{null}$, there exists $q \in Q_c$ such that $t[FK] = q[FK] \wedge t[CFK] = q[CAK]$. □

This formulation takes care of referential ambiguity, but has the unfortunate consequence of curtailing the modeling power of multilevel relations. For example, the instance of figure 16(b) is

SHIP		OBJ		DEST	TC
Enterprise	U	Exploration	U	Talos	U
Enterprise	S	Spying	S	Rigel	S

CAPTAIN		SHIP		TC
Kirk	U	null	U	U
Kirk	U	Enterprise	S	S

(a)

SHIP		OBJ		DEST	TC
Enterprise	U	Exploration	U	Talos	U
Enterprise	U	Spying	S	Rigel	S

CAPTAIN		SHIP		TC
Kirk	U	null	U	U
Kirk	U	Enterprise	S	S

(b)

Figure 16: Foreign key references from CS to SOD

not valid anymore. However, there is nothing semantically incorrect with these relations. We are simply trying to keep the assignment of Kirk to the Enterprise secret, whereas the existence of the Enterprise is unclassified. If we store information about starships and about assignment of captains in two different relations, the SeaView II rule will allow us to keep the assignment of Kirk secret only if it is to a secret starship. We cannot classify the assignment of Kirk to an unclassified starship!

5 PROPOSED SEMANTICS OF REFERENTIAL INTEGRITY

Prior work on referential integrity in multilevel relations leaves us in an impasse. We either have referential ambiguity or substantial loss of modeling power. Since neither of these is a viable alternative, we must find some means of getting around this impasse.

The problem of referential ambiguity arises due to entity polyinstantiation. Therefore our proposal is to retain the original SeaView referential integrity property (i.e., property 6) which allows downward references,⁷ and disallow entity polyinstantiation. Let us see how entity polyinstantiation can be securely prevented.⁸ We distinguish two kinds of relations for this purpose, as follows.

- *Atomic Relations*: In these relations the apparent primary key *AK* does not contain a foreign key as a proper subset of the attributes of *AK*.
- *Composite Relations*: In these relations the apparent primary key *AK* does contain a foreign key as a proper subset of the attributes of *AK*.

These two cases are respectively discussed in the following two subsections.

⁷We will see later in this section that property 6 needs to be slightly modified to work correctly.

⁸Note that element polyinstantiation can also be securely prevented using the technique of [20]. Our proposal is to eliminate entity polyinstantiation as part of the data model, but keep element polyinstantiation as a possible option.

5.1 Prevention of Entity Polyinstantiation in Atomic Relations

The basic technique for preventing entity polyinstantiation in atomic relations is to partition the domain of the primary key among the various security classes possible for the primary key [14].⁹ For our SOD example, we can introduce a new attribute, called *Starship#*. Whenever a new tuple is inserted, we enforce the requirement that all unclassified *Starships* are numbered between 1 and 1,000, all confidential *Starships* are numbered between 1,001 and 2,000, and so on.

In a SQL-like data definition language, the modified SOD schema could be created as follows:

```
CREATE TABLE SOD
( Starship#    SMALL INTEGER NOT NULL [U:TS]
  Starship     CHAR(15) NOT NULL [U:TS]
  Objective    CHAR(15) {U, TS},
  Destination  CHAR(20) [U:TS],
  Primary Key (Starship# ),
  CHECK (Subject Access class = 'U'  AND Starship# BETWEEN 1      AND 1000),
  CHECK (Subject Access class = 'C'  AND Starship# BETWEEN 1001 AND 2000),
  CHECK (Subject Access class = 'S'  AND Starship# BETWEEN 2001 AND 3000),
  CHECK (Subject Access class = 'TS' AND Starship# BETWEEN 3001 AND 4000) );
```

The notation [L:H] specifies a range of security classes with lower bound L and upper bound H. The notation {X,Y,Z} enumerates the allowed values for the security class as one of X, Y or Z. Here, the domain of the security class of the apparent primary key *Starship#* has been specified as a range with a lower bound of U and an upper bound of TS. However, the domain of the *Starship#* has been partitioned across these security classes.

It should be noted that confidentiality does not depend on correct partitioning of the key space by the integrity enforcement mechanism of the Database Management System (DBMS). If this mechanism fails, or is deliberately malicious due to Trojan Horse infection, the integrity properties will fail but there will be no leakage of information. To fully substantiate this statement, we would need to give a kernelized implementation of the DBMS, i.e., an implementation which does not use subjects exempted from the mandatory controls of the underlying multilevel secure operating system. Description of such an implementation is outside the scope of this paper.

5.2 Prevention of Entity Polyinstantiation in Composite Relations

Consider the relations shown in figure 17. SOD is the familiar relation, with apparent primary key SHIP. Let CAPTAIN be the apparent primary key of the relation CR. Now consider the relation CSH, some of whose instances are illustrated in figure 18. The apparent primary key of CSH consists of the attributes CAPTAIN and SHIP. By the entity integrity property (property 1) both attributes must be uniformly classified. Hence only one classification is shown for these two attributes. Suppose CAPTAIN is a foreign key from CSH to CR, and SHIP is a foreign key from CSH to SOD. For the rest of this discussion, assume that SOD and CR are as shown in figure 17.

A valid instance of CSH is shown in figure 18(a). The top two tuples in figure 18(a) correspond to the same entity, viz., <Kirk, Enterprise, U>, and indicate the occurrence of element polyinstantiation. The interpretation is that Kirk is assigned to the Enterprise for 15 hours at the U level, and for 10 hours at the S level. The bottom three tuples of figure 18(a) correspond to three distinct entities, all of which are secret. These three entities represent the assignment of Kirk to Voyager, and the assignments of Spock to the Enterprise and to the Voyager. These entities are labeled S

⁹This is analogous to the manner in which static resource allocation across security classes eliminates covert channels which arise due to dynamic resource allocation in multilevel Operating Systems.

SHIP		OBJ		DEST		TC
Enterprise	U	Exploration	U	Talos	U	U
Voyager	S	Spying	S	Rigel	S	S

CAPTAIN		RANK		TC
Kirk	U	Admiral	U	U
Spock	S	General	S	S

Figure 17: Relations SOD and CR

CAPTAIN	SHIP		HOURS/WEEK		TC
Kirk	Enterprise	U	15	U	U
Kirk	Enterprise	U	10	S	S
Kirk	Voyager	S	30	S	S
Spock	Enterprise	S	20	S	S
Spock	Voyager	S	15	S	S

(a)

CAPTAIN	SHIP		HOURS/WEEK		TC
Kirk	Enterprise	U	10	S	S

(b)

CAPTAIN	SHIP		HOURS/WEEK		TC
Kirk	Enterprise	U	15	U	U
Kirk	Enterprise	U	10	S	S

(c)

CAPTAIN	SHIP		HOURS/WEEK		TC
Kirk	Enterprise	S	10	S	S

(d)

CAPTAIN	SHIP		HOURS/WEEK		TC
Kirk	Enterprise	U	15	U	U
Kirk	Enterprise	S	10	S	S

(e)

Figure 18: Foreign key references from CSH to SOD and CR

because each one of them references a secret entity in SOD or CR or both. Since only downward references are allowed, by property 7 these foreign keys must be labeled S.

Figures 18(b) and (c) illustrate the phenomenon of entities which are not currently visible at a lower level, but may become visible in the future. This situation was encountered in context of the inter-instance property in section 2, and was also discussed in the latter part of section 3. We now see that this phenomenon is useful in relations which relate existing entities. The single S tuple in figure 18(b) assigns Kirk to the Enterprise with a secret load of 10 hours/week. It is possible that later Kirk is assigned to the Enterprise with a unclassified load of 15 hours/week, as shown in figure 18(c). Note that in going from figure 18(b) to (c), from a S subject's point of view, we are not instantiating another entity but merely making an unclassified entity visible at the unclassified level. From a U subject's point of view, we are instantiating another entity at the U level, but this entity may or may not have previously instantiated at a higher level.

Figures 18(d) and (e) illustrate the incorrect approach to handling the situation of figures 18(b) and (c). In this case the S tuple in figure 18(d) is for the entity $\langle \text{Kirk}, \text{Enterprise}, S \rangle$. This opens up the possibility of entity polyinstantiation as shown in figure 18(e). References from some other relation to $\langle \text{Kirk}, \text{Enterprise} \rangle$ in CSH will therefore be ambiguous. In such cases we must make sure that we do not over classify the apparent primary key of CSH.

5.3 Referential Integrity Property

Based on our discussion we recommend going back to the original formulation of the SeaView referential integrity property (i.e., property 6). We need to change this property slightly to avoid references to entities that are potentially visible at level c , but are currently only instantiated at levels above c . This requires the additional condition, $t[C_{FK}] \geq q[TC]$, relative to property 6, giving us the following definition.

Property 8 [Referential Integrity] Let FK be a foreign key of the referencing relation R . Let Q be the referenced relation, with apparent primary key AK . R and Q satisfy referential integrity if and only if for all instances R_c and Q_c occurring together, and for all $t \in R_c$ such that $t[FK] \neq \text{null}$, there exists $q \in Q_c$ such that $t[FK] = q[FK] \wedge t[C_{FK}] \geq q[C_{AK}] \wedge t[C_{FK}] \geq q[TC]$. \square

With this definition, and with elimination of entity polyinstantiation, we will have eliminated referential ambiguity while retaining the expressive power to allow classification of relationships among unclassified entities. Elimination of entity polyinstantiation can be formally expressed as follows.

Property 9 [No Entity Polyinstantiation] A multilevel relation R is said to satisfy the “no entity polyinstantiation” property if and only if for every R_c we have $A_1 \rightarrow C_1$. \square

6 CONCLUSION

In this paper we have shown that previous work on referential integrity leaves us with a choice of either accepting referential ambiguity or severely curtailing the modeling power of multilevel relations. We have shown how to escape this impasse by eliminating entity polyinstantiation, while retaining element polyinstantiation (as an option).

In future work, one should define a formal update semantics for relations which satisfy the core integrity properties of section 2, and the referential integrity and “no entity polyinstantiation” properties of section 5. Completeness and soundness of the semantics should be proved. It is also important to develop correct decomposition and recovery algorithms for a kernelized architecture (i.e., an architecture in which no subject is exempted from the simple-security or star-properties) which give these semantics.

References

- [1] "Multilevel Data Management Security," Committee on Multilevel Data Management Security, Air Force Studies Board, National Research Council, Washington, DC (1983).
- [2] Bell, D.E. and LaPadula, L.J. "Secure Computer Systems: Unified Exposition and Multics Interpretation." MTR-2997, MITRE (1975).
- [3] Burns, R.K. "Referential Secrecy." *IEEE Symposium on Security and Privacy*, Oakland, California, May 1990, 133-142.
- [4] Date, C.J. *An Introduction to Database Systems*. Volume II, Addison-Wesley, (1983).
- [5] Denning, D.E., Lunt, T.F., Schell, R.R., Heckman, M., and Shockley, W.R. "A Multilevel Relational Data Model." *Proc. IEEE Symposium on Security and Privacy*, 220-234 (1987).
- [6] Denning, D.E., Lunt, T.F., Schell, R.R., Shockley, W.R. and Heckman, M. "The SeaView Security Model." *Proc. IEEE Symposium on Security and Privacy*, 218-233 (1988).
- [7] Doshi, V.M. and Jajodia, S. "Referential Integrity in Multilevel Secure Database Management Systems." *Proceedings of the IFIP TC 11 8th International Conference on Information Security*, (1992).
- [8] Doshi, V.M. and Jajodia, S. "Enforcing Entity and Referential Integrity in Multilevel Databases." *Proc. 15th NIST-NCSC National Computer Security Conference*, Baltimore, MD, October 1992, pages 134-143.
- [9] Gajnak, G.E. "Some Results from the Entity-Relationship Multilevel Secure DBMS Project." *Aerospace Computer Security Applications Conference*, 66-71 (1988).
- [10] J. Thomas Haigh, Richard C. O'Brien, and Daniel J. Thomsen, "The LDV Secure Relational DBMS Model." *Database Security IV: Status and Prospects*, S. Jajodia and C. E. Landwehr (editors), North-Holland, 1991, pages 265-279.
- [11] Jajodia, S. and Sandhu, R.S. "Polyinstantiation Integrity in Multilevel Relations." *Proc. IEEE Symposium on Security and Privacy*, Oakland, California, May 1990, pages 104-115.
- [12] Jajodia, S. and Sandhu, R.S. "Polyinstantiation Integrity in Multilevel Relations Revisited." *Database Security IV: Status and Prospects*, Jajodia, S. and Landwehr, C. (editors), North-Holland, pages 297-307, 1991.
- [13] Jajodia, S. , Sandhu, R.S., and Sibley E. "Update Semantics of Multilevel Relations." *Proc. 6th Annual Computer Security Applications Conf.*, Tucson, AZ, December 1990, pages 103-112.
- [14] Jajodia, S. and Sandhu, R.S. "Enforcing Primary Key Requirements in Multilevel Relations," *Proc. 4th RADC Workshop on Multilevel Database Security*, Rhode Island, April 1991.
- [15] Jajodia, S. and Sandhu, R.S. "A Novel Decomposition of Multilevel Relations Into Single-Level Relations." *Proc. IEEE Symposium on Security and Privacy*, Oakland, California, May 1991.
- [16] Lunt, T.F. et al. *Secure Distributed Data Views*. Volume 1-4, SRI Project 1143, SRI International (1988-89).
- [17] Lunt, T.F., Denning, D.E., Schell, R.R., Heckman, M. and Shockley, W.R. "The SeaView Security Model." *IEEE Transactions on Software Engineering*, 16(6):593-607 (1990).

- [18] Lunt, T.F. and Hsieh, D. "Update Semantics for a Multilevel Relational Database." *Database Security IV: Status and Prospects*, Jajodia, S. and Landwehr, C. (editors), North-Holland, pages 281-296, 1991.
- [19] Sandhu, R.S., Jajodia, S. and Lunt, T. "A New Polyinstantiation Integrity Constraint for Multilevel Relations." *Proc. IEEE Workshop on Computer Security Foundations*, Franconia, New Hampshire, June 1990, pages 159-165.
- [20] Sandhu, R.S. and Jajodia, S. "Honest Databases That Can Keep Secrets." *14th NIST-NCSC National Computer Security Conference*, Washington, D.C., October 1991, pages 267-282.
- [21] Sandhu, R.S. and Jajodia, S. "Polyinstantiation for Cover Stories." *Proc. European Symposium on Research in Computer Security*, Toulouse, France, November 1992, pages 307-328. Published as *Lecture Notes in Computer Science, Vol 648, Computer Security—ESORICS92* (Deswarte, Y., Eizenberg, G., and Quisquater, J.-J., editors), Springer-Verlag, 1992.

QUERY ACCELERATION IN MULTILEVEL SECURE DATABASE SYSTEMS

William Perrizo Brajendra Panda
Department of Computer Science
North Dakota State University
Fargo, ND 58105

Abstract

During the past few years research in multilevel secure database systems has received a great deal of attention. While transaction management and query processing in these systems have become the crux of the research, no significant work has been done in the area of query acceleration. In this paper, we describe a fast, space-efficient, and secure technique for accelerating queries that take place among the various base relations in a kernelized multilevel secure database system. Our model follows the SeaView model which uses element level classification of data. Our approach achieves a significant performance improvement when the multilevel query involves selection on one or more attributes of the multilevel relations. This is a common case for large relations. Moreover, this method generates no spurious tuples during the process, which has been a concern in the SeaView model as pointed out by several researchers in the past.

1. Introduction

As the number of database users and the size of databases increase, the security of the databases becomes more and more important. Although existing database systems provide some form of data security, so-called discretionary access controls, they do so by controlling modes of access privileges of users to data. But these do not provide adequate mechanisms for preventing unauthorized disclosure of information. The systems, for example, used in the Department of Defense contain data having different access classes and users with different clearance levels. These kind of systems require the enforcement of mandatory (or nondiscretionary) access control mechanisms [4] so that classified data can be available to cleared users only. Besides, in order to guarantee complete security, the system must protect sensitive information from disclosure through indirect means, such as covert signalling channels [10]. Covert channels are communication channels that allow malicious subjects to transfer information to low users.

The Air Force Summer Study of 1982 [1] proposed various designs for Multilevel Secure Relational Database Management Systems (MLS/RDBMS). Among these, the three most interesting architectures are 1) the Distributed/Replicated architecture, 2) the Kernelized architecture, and 3) the Integrity Lock architecture. The first architecture uses a separate DBMS to manage data at or below each security level; a database at a security class contains all information at its class and below, and therefore, lower level data are replicated in all databases containing higher level data. In this architecture, all reads are local whereas all writes except for data at system-high must be propagated to higher containers.

In the second architecture, the multilevel database is partitioned into single-level databases which are then stored separately. In this case all writes are local but all reads that involve lower level data must read across containers. This makes query acceleration an important factor in the kernelized architecture, which is the subject of our research in this paper. The third architecture is based on the integrity lock technology, and is also called the spray paint DBMS architecture in which data are separated purely by software means. In the rest of this paper we assume kernelized architecture unless otherwise mentioned.

The SeaView model [12], developed as a joint effort by SRI International and Gemini Computers, is a research prototype based on the Kernelized approach that uses element level classification of data. In this

model the multilevel relations are partitioned into single-level base relations that are stored separately. As pointed out by several researchers, the above approach results in poor performance because of the fact that multilevel query involves taking repeated joins of base relations, which is expensive. Also the materialization algorithm used by SeaView to recover multilevel relations gives rise to spurious tuples when elements have been polyinstantiated.¹ In this work we propose a method that will reduce the query response time and eliminate the generation of spurious tuples.

The rest of this paper is organized as follows. Section 2 gives an overview of the security model. The SeaView model and its decomposition and recovery algorithms are discussed in section 3. Our algorithm is presented in section 4. Section 5 gives the performance analysis of our method when compared to a join without any acceleration.

2. The Security Model

One widely accepted model for enforcing mandatory security policies was developed by Bell and LaPadula. It is known as the Bell-Lapadula model [2]. The model is defined in terms of subjects and objects. A subject represents an active entity in the system (e.g., a process), whereas an object represents passive data (e.g., a relation, a record, a field etc.). Every object has a security classification, or access class, which consists of a hierarchical sensitivity level (e.g., TOP-SECRET, SECRET, CONFIDENTIAL, UNCLASSIFIED etc.) and a set of nonhierarchical categories (e.g., FAR EAST, NEAR EAST). Every subject has a clearance level. In order for a subject to be granted access to an object, the Bell-LaPadula model imposes the following restrictions:

The Simple Security Property: A subject can be given a read access to an object only if the subject's clearance level dominates the object's classification level.

*The *-Property:* A subject can be given a write access to an object only if the subject's access set's sensitivity level is dominated by the object's classification level.

The above two properties ensure that the information will only flow monotonically upward, never downward. But in spite of these restrictions, a system may not be fully secure unless it guards against covert signalling channels. Covert channels provide indirect means by which a malicious subject can signal information to a low level subject. For example, a high subject using a read or write lock on a data item at his own level can signal bits of information (e.g., locked = 1, and unlocked = 0) over a period of time to a low subject who also wants to write the same data item.

Besides these, to meet the DoD requirements [13], it must be possible to demonstrate the trustworthiness of the DBMS. To do so the concept of a trusted computing base (TCB) was developed. All security-critical functions are segregated from the rest of the system and kept in the TCB. The TCB must mediate each reference to an object by a subject, allowing or denying the access. It must be tamperproof; it can not be bypassed; and it must be small and simple enough to be verified correct and secure, with respect to the policies it enforces.

3. Multilevel Relations in the SeaView Model

The SeaView model implements a multilevel relation as a view over a set of single-level base relations. A multilevel relation, R , is represented by the schema $R(A_1, C_1, \dots, A_n, C_n)$, where C_i is the classification of the attribute A_i . The domain of C_i is the range of classifications for data that can be associated with attribute A_i and the domain range(A_i) = $[L_i, H_i]$ which is the sublattice of the lattice of access classes. The

¹Polyinstantiation means, there exist multiple tuples at different security levels with the same primary key value.

decomposition formula for automatically decomposing a multilevel relation into single level base relations and the recovery algorithm for materializing the multilevel relation from its base relations have been given in [5] and a modified version is given in [11].

The decomposition formula generates single-level base relations as follows: Let A_1 be the apparent primary key² of the multilevel relation. For each class, x , in $[L_1, H_1]$ one Primary Key Relation, $R_{1,x}(A_1)$ is created, and for every non-primary key attribute, A_i , an Attribute Relation, $R_{i,x,y}(A_1, A_i)$ is created, for each class x, y such that $x \in [L_1, H_1]$, $y \in [L_i, H_i]$, and $x \leq y$. Figure 1 illustrates a multilevel relation MISSILE and the corresponding single-level base relations constructed using SeaView's decomposition method are given in Figure 2. TC represents the Tuple Class of the record, which is the least upper bound of the attribute classifications in the tuple.

MISSILE	name	range	speed	TC
	MT1 U	350 U	750 C	C
	NT5 U	450 U	750 U	U
	NT5 U	480 C	1000 C	C
	DNT U	400 U	750 U	U
	DNT U	450 C	750 U	C
	KR1 U	500 U	800 U	U
	FD7 C	450 C	900 C	C
	KR1 C	400 C	null C	C

Figure 1: A Multilevel Relation MISSILE

MIS	name	range	speed	TC
	NT5 U	450 U	1000 C	C
	NT5 U	480 C	750 U	C

Figure 3: Spurious Tuples in recovering MISSILE relation

MISSILE _{name,u}	name	MISSILE _{range,u,u}	name	range	MISSILE _{speed,u,u}	name	speed
	MT1		MT1	350		NT5	750
	NT5		KR1	500		DNT	750
	DNT		NT5	450		KR1	800
	KR1		DNT	400			
		MISSILE _{range,u,c}	name	range	MISSILE _{speed,u,c}	name	speed
			DNT	450		NT5	1000
			NT5	480		MT1	750
MISSILE _{name,c}	name	MISSILE _{range,c,c}	name	range	MISSILE _{speed,c,c}	name	speed
	FD7		FD7	450		FD7	900
	KR1		KR1	400			

Figure 2: The base relations for MISSILE relation

The recovery of a multilevel relation from single-level base relations is as follows: First, for each primary key class, x , and for each non-primary key attribute, A_i , a relation, $P_{1,x}$, is computed as the union over all multilevel relations, $R_{i,x,y}$, where $x \leq y$. Each tuple in $P_{1,x}$ is of the form (a_1, x, a_i, y) . Let $P_{1,x}$ represent the derived relation (derived from $R_{1,x}$) as $(A_1, C_1 = x)$. Denoting P_i as union of all $P_{i,c}$ where $c \in [L_i, H_i]$, the multilevel relation R is obtained by taking the right outer join of the relations P_i , for $i = 1 \dots n$, where the joining attributes are A_1, C_1 .

²The full primary key consists of the apparent key, its classification, and all classifications for all remaining attributes.

However, as noted above, the SeaView recovery process is not fast enough and it also gives rise to spurious tuples ([8], [9]). Figure 3 shows the spurious tuples generated during the recovery of the MISSILE relation from its base relations. The reason for generation of spurious tuples is as follows: Since some elements of a particular tuple might have been polyinstantiated by users at different higher access classes, different non-key values with the same primary key value appear in different attribute relations. Hence the recovery process creates tuples with all possible matches thus resulting in spurious tuples. Although the user relies on the information that is at the highest visible level, still it is not a time efficient process to find out and suppress the unwanted records. Again, if the multilevel query involves selections either on the attribute values or on classifications, it is required that all the tuples of the participating relations must be read. This results in the increased query response time. The algorithm we propose here can eliminate these problems.

4. Query Acceleration in Multilevel Relations Using DVA

The Domain Vector Accelerator (DVA) concept developed by Perrizo et al. [14], is a space and time efficient method for accelerating joins between relations in traditional Relational Database Systems. The acceleration of queries using this technique results from the speed of operation on bit vectors and complete reduction of page reads from the disk. In this paper we have tailored the DVA data structures, which we present in the following subsection, to fit in the multilevel relational DBMS situation. Section 4.2 presents the algorithm.

4.1. The Data Structures

In our method, for each base relation, a bit vector, called a Domain Vector (DV) needs to be maintained. A DV helps in determining the presence or absence of a value in a relation's joining attribute (i.e., the primary key attribute, in this situation, for each base relation) by the presence or absence of a 1-bit in the corresponding position in the vector. The correspondence between a value and its position in the DV (denoted by value identifier or vid), at each level, is provided by a Domain Value Table (DVT) at that level. However, in all implementations known to the authors, either Relative Record Numbers (RRNs) or Record Identifiers (RIDs) are used for accessing the records in each relation. Hence, a separate DVT would not be necessary since the primary key relation and the RRNs (or RIDs) of its tuples would provide the mapping.

There is one other data structure that needs to be maintained along with the domain vectors: Domain Value Index (DVI), one for the primary key attribute of each base relation, to provide the mapping between a vid and the address of the tuple containing the corresponding domain value. If the relations are indexed on the primary key attributes then these indices could be used as DVIs instead of maintaining separate structures. The DVs and the DVIs for the base relations given in Figure 2 are shown in Figure 4.1 and 4.2 respectively.

4.2. The Algorithm

The algorithm we present here, accelerates the queries by completely reducing the number of pages that must be read from different base relations. However, it shows significant improvement in performance when queries in multilevel databases involve selections on one or more attributes. To start with, let us assume $[a,b]$ as a sublattice of the security lattice, that needs to be accessed by a user to answer the query, where the system low level $\leq a \leq b \leq$ the level of the user. When the user does not explicitly specify the values of a and b , then $a =$ the system low level, and $b =$ the user's level are assumed. As an alternative, these values could also be $a = b =$ the level of the user, as considered in Smith-Winslett Model [16]. For the rest of the paper we deal with levels that are in the interval $[a,b]$ only.

Before performing the outer join as required by the SeaView model to answer a multilevel query, a Query Vector is constructed for every level, x , and is denoted by QV_x . The number of bits in QV_x is the same as

the number of entries in the primary key relation at level x . A bit is set to 1 in QV_x at position i , if the corresponding key value at relative record number i in the key relation participates in the query. If the query does not involve any selection at level x , the QV_x would be entirely 1-bits. Otherwise, the base relations having the participating attributes are read at each level x and the selected vid positions in QV_x are set to ones. If there is more than one attribute involved in the selection criteria then the smallest participating base relation is read, the selected vids are dropped in the index of the next smallest participating base relation to avoid reading non-participating pages, and then the selection criteria is applied to further reduce the number of vids. By continuing this process for all participating base relations, a fully reduced list is obtained and the query vector is built.

DV.MISSILE _{name,u} = 1111	DV.MISSILE _{range,u,u} = 1111	DV.MISSILE _{speed,u,u} = 0111
	DV.MISSILE _{range,u,c} = 0110	DV.MISSILE _{speed,u,c} = 1100
DV.MISSILE _{name,c} = 11	DV.MISSILE _{range,c,c} = 11	DV.MISSILE _{speed,c,c} = 10

Figure 4.1: The DVs for the base relations

MISSILE _{name,u}		MISSILE _{range,u,u}		MISSILE _{speed,u,u}	
vid	rec#	vid	rec#	vid	rec#
1	1	1	1	2	1
2	2	2	3	3	2
3	3	3	4	4	3
4	4	4	2		

MISSILE _{range,u,c}		MISSILE _{speed,u,c}		MISSILE _{name,c}		MISSILE _{range,c,c}		MISSILE _{speed,c,c}	
vid	rec#	vid	rec#	vid	rec#	vid	rec#	vid	rec#
2	2	1	2	1	1	1	1	1	1
3	1	2	1	2	2	2	2		

Figure 4.2 : The DVIs for the base relations

Since some of the polyinstantiated elements create spurious tuples during the outer join process, they need to be processed in a different manner. The attribute relations that are investigated for this purpose are the ones having the attributes required for the output of the query result only. For each primary key attribute set at level x , a Polyinstantiated Domain Vector, $PDV_{x,y}$, is created in the following way for each level y such that $x \leq y$.

- 1) Let $PDV'_{A_i,x,y}$ be the vector obtained by logically ORing the domain vectors: $DV.R_{A_i,x,z}$ for all z , where $x \leq z < y$ and A_i is the attribute required in the output.
- 2) Next $PDV_{A_i,x,y}$ is constructed by ANDing each $PDV'_{A_i,x,y}$ with the corresponding $DV.R_{A_i,x,y}$. The positions of 1-bits in this vector denote the positions of those vids at level x , that have at least one polyinstantiated element for the particular attribute A_i up to level y .
- 3) $PDV_{x,y}$ is constructed by ORing all $PDV_{A_i,x,y}$ s, which represents the vids having polyinstantiated elements in their records that is visible to users up to level y .

Next, for each level, x , the vids that do not participate in the query and/or do not have any polyinstantiated elements visible up to level y are filtered out. The vector that represents such information is obtained by

logically ANDing QV_x with $PDV_{x,y}$, and is denoted by $PQV_{x,y}$. Note that QV_x represents the key values at level x that participate in the query irrespective of whether or not their corresponding records have any polyinstantiated elements.

Before carrying on any build or probe phase of joins, a table, called Select Omit Table (denoted by SOT_x at each primary key level x) is built. The number of columns in each SOT is the same as the number of attributes needed in the output and the number of rows is the exact number of records that would appear in the output. Each element in such a table at level, x , consists of two components, the first one gives the address of a tuple and the second one represents the attribute classification y of the base relation, $R_{A_i,x,y}$, where the tuple appears.

To construct SOT_x , QV_x is taken first and scanned through to find the position of one-bits in it. The position of such a bit indicates that the corresponding key position would appear in the result. To find out the record number of such a key value in attribute relation A_i that is required in the output, the corresponding position in the domain vector, $DV.R_{A_i,x,x}$ is searched. If the bit is one, then the vid is dropped in the DVI of $R_{A_i,x,x}$ to get the record position and the (record address, x) pair is entered in SOT_x under the column A_i . Otherwise, $DV.R_{A_i,x,z}$, where z is the next higher level of x in the security lattice, is checked. The search continues up to level b . If a 1-bit is detected, the corresponding index is checked and the (record address, z) pair is entered in SOT_x under A_i column. Next $PQV_{x,y}$ is scanned for the presence of 1 bits. But this time the search starts from the DV of base relation $R_{A_i,x,y}$ and, if not found, continues downward in the security lattice until the DV of $R_{A_i,x,x}$ is searched.

After constructing SOT_x , for a given x , the element values in each column are sorted in ascending order. This helps minimize the number of page reads from the disk [14]. Then the records are retrieved from the base relations in the following way. If the element (n,z) appears under the column A_i , the record with address n is retrieved from the relation $R_{A_i,x,z}$ and then the build and probe phases of the join are performed. Although our algorithm is independent of any particular join techniques, previous research [7] has shown that using hash join, DV accelerator outperforms all other existing algorithms. For DVH (Domain Vector Hash) method the reader is referred to [7]. Next we present the algorithm formally:

1. For each level x in $[a,b]$ and for $y \geq x$, do the following:
2. For each A_j that participates in selection, read $R_{A_j,x,y}$ and construct QV_x .
3. For each A_i required in output do:
 - 3.1 Construct $PDV'_{A_i,x,y}$ by ORing $DV.R_{A_i,x,x}$ with $DV.R_{A_i,x,z}$ for each $z < y$.
 - 3.2 Create $PDV_{A_i,x,y}$ by ANDing $PDV'_{A_i,x,y}$ with $DV.R_{A_i,x,y}$.
4. Create $PDV_{x,y}$ by ORing all $PDV_{A_i,x,y}$ s.
5. Construct $PQV_{x,y}$ by ANDing QV_x with $PDV_{x,y}$.
6. Scan QV_x and for the presence of every 1-bit at position k do the following.
 - 6.1 For each A_j required in output, set $z = x$ and do the following:
 - 6.1.1 Check k th position in $DV.R_{A_j,x,z}$. If it is also one, drop the corresponding vid in DVI of $R_{A_j,x,z}$ and get the record address. Insert the (record address, z) pair in A_j column of SOT_x and go to step 6.1. If the k th position in $DV.R_{A_j,x,z}$ is zero, and $z < y$, set z to next higher

- level and continue with step 6.1.1. If $z = y$, go to step 6.1.
7. Scan $PQV_{x,y}$ and for the presence of every 1-bit at position k do the following.
 - 7.1 For each A_i required in output, set $z = y$ and do the following:
 - 7.1.1 Check k th position in $DV.R_{A_i,x,z}$. If it is also one, drop the corresponding vid in DVI of $R_{A_i,x,z}$ and get the record address. Insert the (record address, z) pair in A_i column of SOT_x and go to step 7.1. If the k th position in $DV.R_{A_i,x,z}$ is zero, and $z > x$, set z to next lower level and continue with step 7.1.1. If $z = x$, go to step 7.1.
 8. Sort every column of SOT_x in ascending order.
 9. For every element (n,z) under column A_i of SOT_x bring n th record from relation $R_{A_i,x,z}$ to memory, find the match and concatenate the attribute value.
 10. If a record is complete, write the record to output buffer.
 11. Flush the output buffer when full.

Here we consider a query, in the MISSILE relation, by a user having clearance up to C level:

SELECT * FROM MISSILE WHERE range \geq 400

Since the user does not specify the classes from which the answer should be retrieved, by default, we consider all the classes visible to the user, as mentioned before. As per the algorithm, the base relations $MISSILE_{range,u,u}$, $MISSILE_{range,u,c}$, and $MISSILE_{range,c,c}$ are read and the selection criteria is applied to get the selected key values: KR1, NT5, DNT at U level, FD7 and KR1 at C level. Applying these values to the $DVT.MISSILE_u$ and $DVT.MISSILE_c$ respectively the QVs constructed are: $QV_u = 0111$, and $QV_c = 11$.

Next, $PDV_{u,c}$ is built in the following way.

- 1) $PDV'_{name,u,c} = DV.MISSILE_{name,u,u} = 1111$
 $PDV'_{range,u,c} = DV.MISSILE_{range,u,u} = 1111$
 $PDV'_{speed,u,c} = DV.MISSILE_{speed,u,u} = 0111$
- 2) $PDV_{name,u,c} = PDV'_{name,u,c} \text{ AND } DV.MISSILE_{name,u,c} = 0000$
 $PDV_{range,u,c} = PDV'_{range,u,c} \text{ AND } DV.MISSILE_{range,u,c} = 0110$
 $PDV_{speed,u,c} = PDV'_{speed,u,c} \text{ AND } DV.MISSILE_{speed,u,c} = 0100$
- 3) $PDV_{u,c} = PDV_{name,u,c} \text{ OR } PDV_{range,u,c} \text{ OR } PDV_{speed,u,c} = 0110$

This denotes that the records in the multilevel relation MISSILE, with key values at position 2 and 3 in the $MISSILE_{name,u}$, i.e., NT5, and DNT, have some elements polyinstantiated by user(s) at C level.

Then $PQV_{u,c}$ is created by ANDing QV_u with $PDV_{u,c}$, which is 0110. Thereupon SOT_u is constructed which has three columns one for each attribute in the multilevel relation, MISSILE. First QV_u is scanned and it is noted that the second, third and fourth positions in QV_u have 1-bits. After checking the $DV.MISSILE_{name,u,u}$ it is found that the second bit in it is one. (Of course $DV.MISSILE_{name,u,u}$ need not be checked since it is the primary key attribute, it would be entirely ones.) Dropping vid 2 in DVI of $MISSILE_{name,u}$ the record number obtained is 2 and hence under the name column of SOT_u the element (2,u) is inserted. Next $DV.MISSILE_{range,u,u}$ is checked and a 1-bit is found in second position. So vid 2 is dropped in the index of $MISSILE_{range,u,u}$ and 3 is received as the record number. The pair (3,u) is inserted in the range column of the table. Proceeding in a similar manner, the pair (1,u) is inserted in

SOT_u . This completes the first record of the table. For the other one-bit positions in QV_u , i.e., for vid 3 and 4, the construction of SOT_u is continued in a similar manner. Next $PQV_{u,c}$ is scanned and other records are inserted into SOT_u which is obtained as given in Figure 5.

SOT_u	name	range	speed
	2,u	3,u	1,u
	3,u	4,u	2,u
	4,u	2,u	3,u
	2,u	2,c	1,c
	3,u	1,c	2,u

Figure 5: The SOT_u

MIS	name	range	speed
	NT5 U	450 U	750 U
	DNT U	400 U	750 U
	KR1 U	500 U	800 U
	NT5 U	480 C	1000 C
	DNT U	450 C	750 U
	FD7 C	450 C	900 C
	KR1 C	400 C	null C

Figure 6: The Result of the Query

After constructing the select omit table at the u level, the records are brought into memory from the base relations as described in the algorithm, and then the build and probe phases of hash join are performed. Upon the completion of processing of each record it is written into the output buffer, and the output buffer is flushed when full. Next, SOT_c is built and the join process continues. The final result thus obtained by the algorithm is as shown in Figure 6.

4.3. Security of the Algorithm

The data structures we maintain in this method are classified according to the associated primary key relations, and therefore, they do not signal information downward. For example, the DV that represents the presence or absence of the joining attribute, i.e., the primary key, of each base relation is classified at the same level of the primary key. And always, in a multilevel relation, the primary key classification is dominated by the classification of all other attributes in a record. Hence this is true in case of the base relations. Therefore, a user can see a record if and only if he has access to the key attribute and since the classification of the key attribute is the same as that of the DV attribute, a subject can access the DV if and only if the subject's classification level is higher than or equal to that of the DV level. Since domain vectors apply to queries only, we do not have to worry about violating the *-property of the security model. Again we never allow a subject to retrieve information from a level higher than that of the subject.

5. Performance Analysis

In this section we compare the secondary storage page I/O costs associated in answering a query with two different techniques: one with the DVA algorithm as join accelerator and one without any join accelerator. In this paper we ignore the CPU cost etc. since they are negligible when compared to disk page I/O costs. Our cost model is similar to the models used in ([3], [6], [15], [17]). The values of different fixed parameters are shown in Figure 7.

The total number of page accesses needed in order to retrieve two, four, and eight attributes of the multilevel relation is calculated and presented in the following graphs. These cost figures were calculated using Yao's formula [18]

$$Yao(k,m,n) = m - m * \prod_{i=1}^k ((n - (n/m) - i + 1) / (n - i + 1))$$

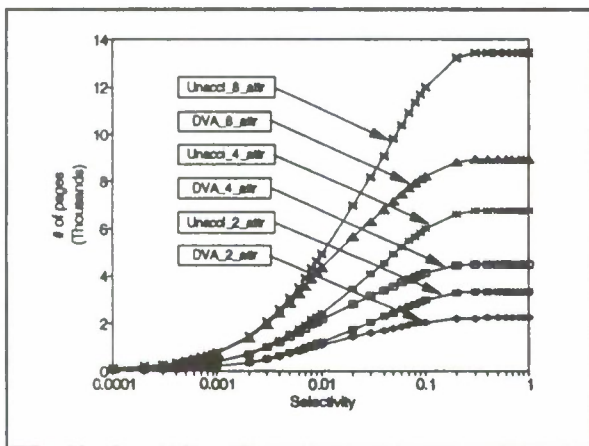
which gives the number of page accesses needed to get k records randomly distributed in a file of n records stored in m pages given that a page is accessed at most once. To retrieve records from the base

relations at different levels, an unaccelerated algorithm accesses the index of each base relation whether the tuples exist in each of them or not. But the DVA algorithm does not access the index unless a record exists in the corresponding base relation. The cost of page access associated with reading the index of a relation having n records in m pages, when k records are retrieved is calculated by Yao(Yao(k, m, n), $m/FO, m$), where FO is the average fan out of an index node in a B^+ tree.

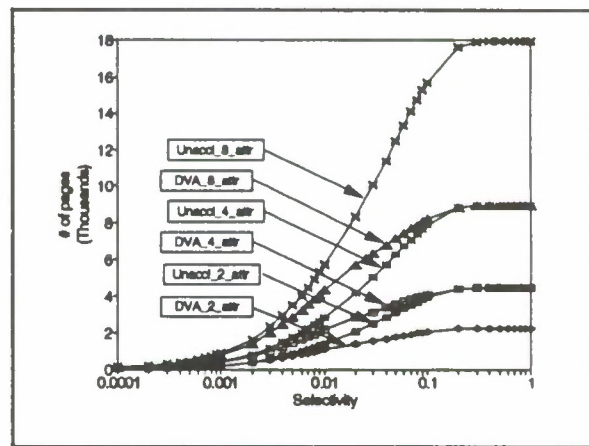
Page size = 4 KB
 Number of tuples in each base relation = 100,000
 Number of pages in main memory = 1,000
 Size of an attribute = 8 bytes
 Size of a value identifier = 4 bytes
 Average fan out of an index in B^+ tree = 287
 Average page occupancy factor = 0.7

Figure 7: Fixed parameters for the cost model

Graph 1 and Graph 2 show the results when the query retrieves the records from two and three different security levels respectively. The polyinstantiation rate is taken as 10 percent and the probability that a record has a null value in the query result (the value may exist at a higher level than accessed) is taken as 0.1. As can be observed, the savings in using the DVA algorithm increases as the number of levels increases, as the number of categories increases, and also as the number of attributes required for output increases. As the selectivity reaches 0.2, the number of pages required to retrieve the records exceeds the total number of pages in the base relations. The reason for this is that there are many page access to the indices. In that case, the algorithm could be changed to sort merge join instead of accessing the index of a relation and hence the total number of pages read will be the same as the total number of pages in the base relations. However, DVA still outperforms any join method in this case.



Graph 1: Two Classification Levels Accessed



Graph 2: Three Classification Levels Accessed

6. Conclusions

In the past few years, database researchers have constantly expressed concern about the performance of multilevel database management systems based on the kernelized architecture. The principal reason for the performance degradation is that processing multilevel queries requires taking repeated joins of the base relations, particularly since joins are among the most expensive database operations. In this paper

we have developed an algorithm to accelerate joins in multilevel secure database systems which are based on a model similar to SeaView. This method violates no security policy and also generates no spurious tuples. Acceleration results from factors such as the speed of operation on bit vectors and the restriction of I/O to only those tuples that participate in the final result. Since the data structures maintained are also classified according to the level of information they contain, our algorithm establishes no direct or indirect downward information flow. As future research, we wish to analyze further possibilities of improving performance of the Jajodia-Sandhu decomposition algorithms ([8], [9]) and acceleration of queries in the replicated architecture.

Acknowledgment

The authors express their gratitude to the referees for their excellent scrutiny of the previous version of the paper and valuable suggestions rendered by them.

REFERENCES

- [1] "Multilevel Data Management Security", Committee on Multilevel Data Management Security, Air Force Studies Board, National Research Council, Washington D.C., 1983.
- [2] D. E. Bell and L. J. LaPadula, "Secure Computer Systems: Unified Exposition and Multics Interpretation", The Mitre Corporation, March 1976.
- [3] J. A. Blakely, and N. L. Martin, "Join Index, Materialized View, and Hybrid-Hash Join: A Performance Analysis", Proc. IEEE Data Engineering Conference, 1990.
- [4] D. E. Denning, "Cryptography and Data Security", Addison-Wesley, Reading, Mass., 1982.
- [5] D. E. Denning, and T. F. Lunt, "A Multilevel Relational Data Model", Proc. of IEEE Symposium on Security and Privacy, p. 220-234, Oakland, CA, April 1987.
- [6] D. J. DeWitt, R. H. Katz, F. Olken, L. D. Shapiro, M. R. Stonebraker, and D. Wood, "Implementation Techniques for Main Memory Database Systems", SIGMOD, p. 1-8, 1984.
- [7] J. Gustafson, W. Perrizo, K. Scott, D. Thureen, and W. Davidson, "DVH: A Query Processing Method using Domain Vectors and Hashing", 2nd International Workshop on RIDE:TQP, p. 116-122, Tempe, AZ, Feb. 1992.
- [8] S. Jajodia, and R. Sandhu, "Polyinstantiation Integrity in Multilevel Relations", Proceedings of the IEEE Symposium on Security and Privacy, p. 104-115, May 1990.
- [9] S. Jajodia, and R. Sandhu, "A Novel Decomposition of Multilevel Relations into single-level Relations", Proceedings of the IEEE Symposium on Security and Privacy, p. 300-313, May 1991.
- [10] B. W. Lampson, "A Note on the Confinement Problem", CACM, (16) 10 p. 613-615, October 1973.
- [11] T. F. Lunt, R. R. Schell, W. R. Shockley, D. Warren, "Toward a Multilevel Relational Data Language", Proc. of the IEEE Symposium on Research in Security and Privacy, p. 72-79, 1988.
- [12] T. F. Lunt, D. E. Denning, R. R. Schell, M. Hechman, W. R. Shockley, "The SeaView Security Model", IEEE Transactions on Software Engineering, Vol. 16, No. 6, June 1990.
- [13] National Computer Security Center, "Department of Defense Trusted Computer System Evaluation Criteria," Technical Report DOD 5200.28-STD, Department of Defense, December 1985.
- [14] W. Perrizo, J. Gustafson, D. Thureen, D. Wenberg, W. Davidson, "Domain Vector Accelerator (DVA): A Query Accelerator for Relational Operations", Proc. IEEE Data Eng., Kobe, Japan, 1991.
- [15] L. Shapiro, "Join Processing in Database Systems with Large Main Memories", TODS, p. 239-264, 1986.
- [16] K. Smith, M. Winslett, "Entity Modeling in the MLS Relational Model", Proceedings of the 18th VLDB Conference, Vancouver, British Columbia, Canada, 1992.
- [17] P. Valduriez, "Join Indices", TODS, p. 218-246, 1987.
- [18] S. B. Yao, "Approximating Block Accesses in Database Organizations", CACM, (20) 4, p. 260-261, Apr. 1977.

DISCRETIONARY ACCESS CONTROL IN OBJECT-ORIENTED DATABASES: ISSUES AND RESEARCH DIRECTIONS

Roshan K. Thomas and Ravi S. Sandhu¹

Center for Secure Information Systems
&

Department of Information and Software Systems Engineering
George Mason University, Fairfax, VA 22030-4444

ABSTRACT In recent years we have witnessed considerable efforts in the research and development of object-oriented database management systems. As object-oriented database technology matures, the availability of adequate access control mechanisms will be crucial to its commercial acceptance. In this paper we discuss discretionary access control issues in object-oriented databases. Our objective is two-fold. One objective is to survey the state of the art in access control concepts and mechanisms as reported in the relevant literature. To do this, we develop a framework to categorize access control issues. The categories include subject to object, inter-object, and intra-object access control. We cover structural and behavioral approaches to access control. Another objective is to identify several research directions and access control issues that are beyond the scope of existing mechanisms. These include authorizations based on separation of duties and multiple approvals, the incorporation of temporal semantics, and transaction based authorization.

Keywords: Object-oriented databases, discretionary access control, integrity, authorization, protection groups, separation of duties, composite objects

1 INTRODUCTION

In recent years we have witnessed considerable efforts in research and development of object-oriented databases. The driving force behind these efforts have come from emerging application domains such as computer-aided design (CAD/CAM), software development, office automation, to name a few. These domains call for modeling capabilities that are beyond the scope of record-based data models. The main attraction of the object-oriented paradigm is its ability to model entities with complex structure and behavior.

With the ever-increasing threats to the security of computing systems, the maturing and commercial acceptance of object-oriented database technology depends to a large degree on the provision of adequate security and integrity mechanisms. In this paper, we survey discretionary access control issues and mechanisms that have been reported in the current literature, and further identify some promising research directions.

In order to fully exploit the benefits of the object-oriented paradigm, it is important that we consider the data model impacts of object-orientation on access control mechanisms. In particular, the data elements and units of access, as well as the different operation types (that need to be supported by the access control mechanism), are all heavily influenced by the underlying data model. At the same time, we must recognize that there are general principles and mechanisms that are unaffected by the object-oriented data model and thus still applicable. An example of this would be the idea of grouping users into access control/protection groups. This would offer the obvious

¹The work of both authors is partially supported by a grant from the National Security Agency, contract No: MDA904-92-C-5140. We are grateful to Pete Sell, Howard Stainer, and Mike Ware for their support and encouragement.

capability of granting (and revoking) privileges to an entire group, thereby eliminating the burden of providing such privileges individually to every member of the group. The harmonious marriage of data model dependent and general access control mechanisms is the key to building a flexible and yet general purpose access control facility for object-oriented databases.

Dittrich [4] has provided a useful taxonomy of object-oriented databases. Structurally object-oriented database systems provide support for the modeling and manipulation of complex (nested) object structures. Behaviorally object-oriented database systems model the behavior of real world entities by allowing the user to define type-specific operators (methods) that make up object interfaces. An object is thus essentially an instance of an abstract data type. Object state is now accessible only through these methods. Fully object-oriented systems provide the capability for modeling both the structure as well as behavior of objects. We will discuss later in the paper, access control mechanisms that are specific to each category.

The current literature in access control and integrity mechanisms for object-oriented databases do not elaborate in any detail issues such as separation of duties, authorizations based on multiple approvals, temporal semantics, to name a few. We identify some promising approaches to address these issues.

The rest of this paper is organized as follows. Section 2 discusses the many issues, approaches, and mechanisms of discretionary access control that have been reported in the literature. Section 3 highlights some research directions, and section 4 concludes the paper.

2 DISCRETIONARY ACCESS CONTROL: ISSUES AND APPROACHES

In this section we give a brief introduction to the object-oriented paradigm and basic concepts in access control, followed by a discussion of access control issues and mechanisms for structurally and behaviorally object-oriented databases.

2.1 Overview of Basic Concepts

In the object-oriented paradigm, the *object* is a central abstraction that models a real world entity. Every object encapsulates some state and is further uniquely identified by an object-identifier. The state of an object is made of the values of its attributes (that describe the real world entity modeled). In behaviorally object-oriented databases the object state is accessible only through the operations (methods) supported by its interface(s). Every operation (method) is associated with a method body that contains some piece of executable code that models the behavior of the corresponding real world entity. Every object belongs to a type that is determined by its *class*, and is thus considered to be an instance of the class. A class is thus akin to an abstract data type definition. Classes can be organized into class hierarchies enabling the sharing of structure and behavior through the mechanism of *inheritance*. A class may inherit from higher classes, but in addition may also contain locally defined structure and behavior. A class lower in the hierarchy is thus considered to be more specialized than the higher superclasses.

When an object references a second object and is related to the latter by an IS-PART-OF relationship, we model the notion that the second object is a part (component) of the first. A collection of related objects in this manner can be treated logically as a single unit called a *composite object*. In the model for composite objects discussed in [9] an individual component may be exclusive or shared. If a component is declared to be exclusive then it can be a component only in one composite object, at any given time. If it is shared, it may be a component of several composite objects.

In addition to composite objects, some systems also support the notion of a *versioned object*. A versioned object consists of a hierarchy of objects called a version hierarchy. Objects in the version

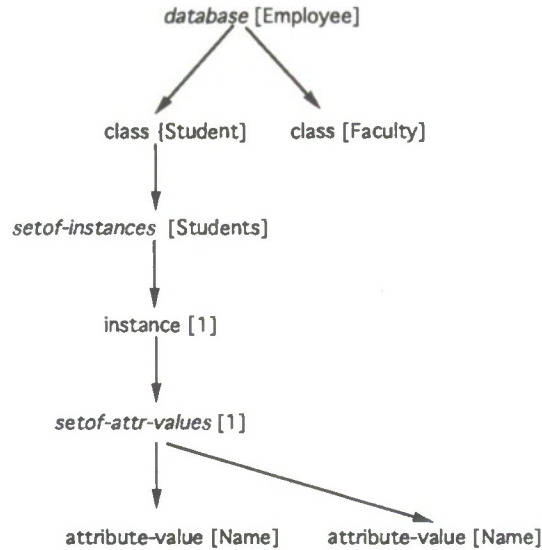


Figure 1: An authorization object lattice for classes

hierarchy are derived from one another, with the root object (called the generic object/instance) storing the history of change in the hierarchy.

Historically, the access control problem has been couched within the framework of subjects, objects, and rights (access types). Within this subject-object paradigm of access control, an *object* refers to any entity that holds data (such as files, records, directories). When we discuss access control in object-oriented databases, we must map this general notion of objects to the narrower meaning of objects in the object-oriented sense.

All access control problems eventually seek an answer to a fundamental question typically posed as follows: Is subject s allowed access of type a on object o ? As given in [14], it is useful to consider the notion of an *authorization* as a 3-tuple (s, o, a) , where $s \in S$, the set of subjects, $o \in O$, the set of objects, and $a \in A$, the set of access/authorization types. An example of an authorization would be *(John, Mydirectory, Read)*. The answer to any access control request can now be obtained by utilizing a function f that determines if the corresponding authorization (s, o, a) is true or false. In [14], the authors advance the notions of implicit, positive, negative, strong, and weak authorizations. A brief look at these concepts is useful for later discussion.

Rather than storing the value of the function f explicitly for all possible triplets (s, o, a) , the idea of implicit authorization allows us to deduce some of these values from ones that are stored (in the authorization base). This may be useful for example, if we want authorization to a class to imply authorization to all instances of the class. A positive authorization gives permission for access ($f(s, o, a) = \text{true}$), while a negative authorization models a prohibition ($f(s, o, \neg a) = \text{true}$). Finally, a strong authorization (including implied ones) cannot be overridden, while weak ones can.

A well known access control principle is to organize subjects into access control groups, based on their roles in an organization [16]. This makes it easier to grant and revoke authorizations to entire groups of subjects/users at a time. The existing proposals on discretionary access control in object-oriented databases, have taken advantage of this. In [14] a role lattice is used to define such groups, and implicit authorizations propagate from the top to the bottom of the lattice.

The approach in [14] also utilizes an authorization object lattice. Thus if we want authorization on a class to imply authorizations on all instances of the class, we would define authorization objects *class* and *setof-instances* and form a lattice with the latter being lower in the lattice. Implicit authorizations are now applied on the lattice (see figure 1). As per the convention in [14], we show in *italics* the nodes from which implicit authorizations may flow to a set of authorization objects.

We end this overview section by giving a useful categorization of both structural and behavioral access control issues in terms of where access is mediated. We distinguish three cases:

- **Subject to Object:** Here we are concerned with how a subject (or principal) establishes an initial authorized point of contact with an object.²
- **Inter-object:** Inter-object access control is concerned with issues such as the visibility and and propagation of authorizations across object boundaries, as a consequence of an initial subject to object authorization.
- **Intra-object:** Here we deal with access control within the internal structure and behavior of an individual object. These issues are thus irrelevant to other objects in the system.

The above categorization allows us to see which dimension of the overall access control problem is tackled by individual approaches, and where these approaches are collectively lacking.

2.2 Structure-based access control

In structural approaches to access control, the access/operation types we deal with are typically read, write, delete, and read-definition. Thus, an access control request poses the basic question: Is subject A allowed to read/write/delete object O? Let us see the details on how this question is answered.

2.2.1 Subject to object access control

Existing approaches in the literature for subject to object structure-based access control are rather straightforward [4, 14]. The basic idea is to group subjects into access control groups and to grant authorizations in terms of access types such as read, write, and delete. These access types are usually ordered such that the authorization for one right may include others. Thus an authorization for a delete may imply authorization for a write, which in turn may imply authorization for a read. In [14] this is accomplished by utilizing implicit authorizations along an access/authorization type lattice.

2.2.2 Inter-object and intra-object access control

We now discuss inter-object and intra-object access control issues. In our discussion, some of the issues are difficult to categorize cleanly as inter-object or intra-object, or both. For example, the inheritance of attributes is an inter-object issue since it involves at least two objects, but at the same time is also an intra-object issue for the object that is inheriting the attributes.

Variable granularity for access units

In structurally object-oriented database systems, the access control mechanisms would have to be flexible enough to support varying granularity of access units. For example, it may be desirable in some applications to have fine-grained access control at the level of the individual attributes of an object. But it may also be desirable to grant access/authorization to larger substructures (such as entire objects or composite objects) as a single unit.

We highlight briefly two contrasting approaches, one in the DAMOKLES database [4] and the other for the authorization model based on the ORION system [14]. We defer discussion on providing varying access granularity in composite objects to a later section. In DAMOKLES, every object (in the data modeling and object-oriented sense) is further broken down into smaller access units called protection objects (p-objects). These p-objects include the descriptive part D consisting of the object's attributes, the structural part S consisting of the components/composite objects, and version part V consisting of the object's versions. To treat all the attributes as a single unit, an authorization is granted on the D part.

²For convenience we use the terms "subject" and "principal" synonymously. In a strict sense, a human user may have several principals, with each principal associated with one or more subjects in the system.

The ORION approach utilizes the idea of implicit authorizations along an authorization object lattice. Thus to grant authorization on all attributes of an instance, we grant authorization on the authorization object type *Setof-Attr-Values* which leads to an implied authorization on authorization object *Attribute-Value*.

Class hierarchy and structure inheritance

Support for class hierarchies and inheritance in the object-oriented paradigm have an impact on how we approach access control issues. In particular, the following questions need to be addressed.

- What effect does allowing implicit authorizations in the class hierarchy have on the reusability of classes? on query processing?
- What should be the semantics for the inheritance of structure (attributes) among classes when subjects have differing authorizations on these classes?

Consider the alternate ways of handling implicit authorizations between a class and the instances of its subclass [14]. Our first option would be for a creator of a class to be given implicit authorizations on all instances of a subclasses derived (specialized) from the class. Queries rooted at the class and spanning lower subclasses can now be evaluated successfully as the required authorization can be obtained. However, this approach makes the classes too interdependent making their potential for reuse very low. The second option would be to prohibit implicit authorizations from classes to instances of derived subclasses. This would encourage the reusability of classes, but this benefit comes at the cost of query failures.

In [21] Spooner has raised some of the issues that arise when subjects have differing authorizations on classes, and inheritance is allowed. To restate an example in [21], consider a class B that is a subclass of another class A. If a subject is authorized to access B but not A, should the subject be allowed to see the inherited attributes from A when he accesses B? The approach taken in [7] would allow access to all the inherited attributes in B. The advantages and disadvantages of this approach need further analysis.

Access control in composite objects

Supporting composite objects requires us to address the following questions (among others).

- What is the implication of several components of the same object being owned by different subjects/users?
- How do we connect rights (authorizations) through composite object hierarchies?
- How do changes in the composite object structure (hierarchy) affect existing and future connections of authorizations?
- What are the semantics to handle conflicting authorizations at points in the composite object hierarchy?
- How do we handle transitive authorizations in composite object hierarchies?

In environments such as those supporting CAD/CAM, it is typical for designers to create and work on the design of individual components. The objects representing these components will thus be owned by different subjects. However when cooperative activity such as the exchange of partial designs or the assembling of entire composite objects are involved, a subject may have to obtain authorization from the individual owners of the composite objects.

In DAMOKLES, the approach to connecting authorizations along composite object hierarchies involves the use of *complex authorizations/rights*. A complex authorization differs from a simple one

as follows. When applied to a root object in a composite object hierarchy, an authorization extends to all current as well as future composite objects connected to this root, so long as they have the same owner as the root.

The approach used in [14] to connect authorizations on composite objects is based on an authorization object lattice defined for composite objects. By making use of implicit authorizations along this lattice we get more flexibility than the hard-wired approach of DAMOKLES. However, the propagation of positive and negative authorizations along the composite object lattice may lead to conflicts. This happens for example if a previously granted authorization on a component object conflicts with a new authorization (implicit or explicit) that is received. As mentioned in [14] conflicts from negative authorizations also arise on objects that are components of more than one composite object. The access control mechanism must reject conflicting authorizations based on some consistent semantics.

Access control on versions

To provide access control on versions, the proposal in [14] once again utilizes an authorization object lattice with authorization objects such as *setof-generic-instances*, *setof-versions*, among others. The notion of implicit authorizations are again used on this lattice. In DAMOKLES [4], an authorization on the V part of an object obtains authorizations on all the versions of the object.

These approaches need to be refined to provide more selectivity on the versions belonging to a version hierarchy. For example, we may want to specify that a subject be authorized for the first/last three versions.

Meaningful interconnection controls among component objects

All our discussions above on access control in composite objects have assumed that component objects are linked in some meaningful way. Existing approaches place the burden on users/subjects for establishing meaningful interconnections and visibilities across component object boundaries. This might be a reasonable expectation in some environments. After all, we would expect a designer in a CAD environment to be knowledgeable enough not to mix and match the components of say, cars and trucks. However, when discrimination between components is not easy, we would like the access control mechanisms to help. For example we could have an access control list that governs how objects are interconnected. The access control list would place restrictions on the IS-PART-OF relationships that can be formed between component objects.

2.3 Behavioral and semantic based access control

2.3.1 Subject to object access control

As mentioned before, subject to object access control is concerned with the authorization of the initial point of contact with an object by a subject. In a behaviorally object-oriented database, this would involve authorization to invoke an initial method in a chain/tree of method invocations. Thus if a subject invoked an initial method m_1 which in turn invoked m_2 , and m_2 in turn invoked m_3 , we are concerned with how the subject gets authorization for m_1 . Authorization for the other methods m_2 and m_3 fall into the category of inter-object and intra-object access control and will be discussed subsequently.

We describe three approaches for subject to object access control that have been reported in the literature. In the first [15], associated with every object is an access control list (ACL) and an object owner. The owner of an object controls through the ACL the other principals that may invoke the operations (methods) defined for the object.

In the second approach [23], access groups based on user roles are defined with the help of a user role definition hierarchy (URDH). A node in the URDH represents an access group. Based on the

access control requirements, the publicly accessible methods are assigned to nodes in the URDH. A subject/user belonging to a particular node in URDH will be allowed to invoke only those methods assigned to that node.

A third approach described in [5] uses the notion of interface objects. Every database object is associated with a collection of interface objects. An interface object supports only a subset of the total methods in a database object. Subjects are allowed to interact with the database objects only by invoking methods defined in their corresponding interface objects. In summary, an effective subject to object access control mechanism is built by defining a collection of interface objects for every database object, and by restricting subjects to one or more interfaces.

2.3.2 Inter-object access control

What are the implications of supporting behavior based access control, across object boundaries? In particular, it is important to recognize that objects are autonomous entities taking part in a distributed computation. Two important questions come to the forefront.

- How do we control visibility and interaction between objects, in terms of behavior?
- What are the semantics for propagating authorizations along method invocation chains and trees?

We consider answers to these questions in turn.

Inter-object method invocation and visibility

If we wish to control the visibility of a method m , then we should restrict the number of client methods than can invoke m . An approach suggested in [2] is to associate a set $\langle \text{invokers} \rangle$ with the definition of every method (such as m). This set contains the names of methods and classes to which m is made visible to. In the case of a class, m is visible to all the methods in the class. Although this appears to be a good first step, several avenues need further investigation. Figure 2 illustrates a method m_0 which is visible to a class c_1 that is part of the invoker set of m_0 . The locally defined method m_1 in c_1 can thus invoke m_0 . Now consider the class c_2 which is a subclass of c_1 and has a locally defined method m_2 and by virtue of its position in the class hierarchy inherits m_1 . Should an invocation of method m_1 locally from class c_1 be treated differently from an invocation of m_1 by m_2 from the subclass c_2 , since in either case m_1 eventually invokes m_0 ? Should access control prevent method m_2 from invoking m_1 as long as m_1 can invoke m_0 but m_2 cannot? If it does not, we may as well make class c_2 part of the invoker set of m_0 .

Authorization propagation through method invocation chains/trees

Here we are interested in coming up with consistent semantics as well as flexible mechanisms for propagating authorizations through method invocation chains/trees. The use of implicit, positive, negative, strong, and weak authorizations need to be studied. Once again conflicts from negative and positive rights may arise.

2.3.3 Intra-object access control

The need for access control resurfaces even within the boundary of an object. It must be recognized that in object-oriented systems, access control and integrity mechanisms are closely linked. This is because methods modify the states of objects and we often enforce access control on method invocations. Integrity after all, is concerned with the improper modification of data.

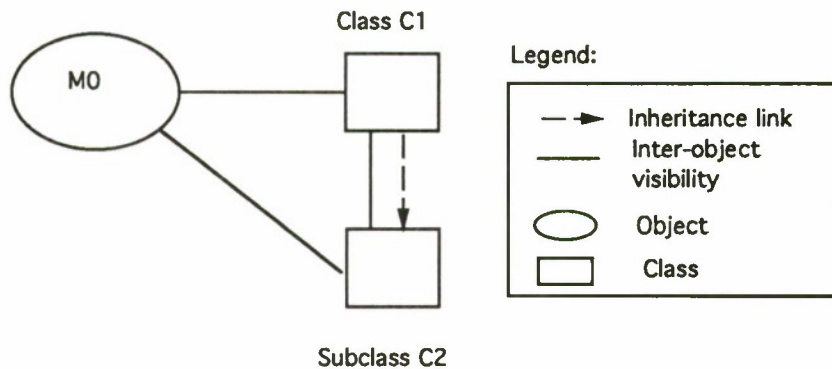


Figure 2: Visibility across methods and classes

Method to attribute visibility

For some applications, it may be desirable to allow only certain methods in the object (or class) to access a local attribute. For example, in a military application an attribute 'Target-coordinate' may be updated only by a method 'Approve-coordinate' which ensures that the new coordinates are not erroneous. An obvious way to achieve this would be for every attribute to maintain a list of methods that are allowed access (to the attribute).

Method to method visibility

Within an object boundary we may want to restrict the visibility of local methods to each other. Again an obvious way to accomplish this would be for every method to maintain some list. More complicated data structures are worth investigating, especially if the notion of implicit authorizations can be applied.

3 RESEARCH DIRECTIONS

In this section, we identify some issues that are beyond the capabilities in current proposals for discretionary access control in object-oriented databases. Addressing these issues would lead to access control models and mechanisms that accommodate the diverse security policies and controls of information management in organizations.

3.1 Transitive rights

The implications of the transitive propagation (taking and granting) of rights in composite-object hierarchies and method invocation chains warrants further investigation. Some discussion of this for method invocation chains can be found in [15] (we do not discuss this work due to space constraints). Given a certain set of explicit authorizations, we would like to know if an access control request from a user will succeed. Also, when a change is made to certain authorizations, what is the overall effect on users?

3.2 Separation of duties and multiple approvals

The operational procedures in many organizations are designed to prevent fraud. Separation of duties and multiple approvals are well known principles to achieve this. For an activity to be authorized, it may need multiple approvals by separate individuals. Recently, Sandhu in [17, 18] has proposed *transactions control expressions* as an approach to implement these in computerized systems. It is based on a database activity model that utilizes the notions of transient and persistent objects. Transient objects include vouchers, purchase orders, sales slips, to name a few. These objects are transient in nature in the sense that they issue a finite set of operations and then leave the system (in a paper world this happens when a form is archived). These operations eventually affect persistent objects such as inventory databases, and bank accounts. The fundamental idea is to enforce controls primarily on the transient objects, and for transactions to be executed on persistent objects only as a side effect of executing transactions on transient objects.

As an example, consider a check processing application where a clerk has to prepare a check and assign an account, followed by three (separate) supervisors who have to approve the check and account, and finally the check to be issued by a different clerk. This can be represented by the following transaction control expressions:

```
prepare • clerk;  
3: approve • supervisor;  
issue • clerk;
```

The colon is a voting constraint specifying 3 votes from 3 different supervisors. Each expression consists of a *transaction* and a *role*. Separation of duties is achieved by requiring the users who execute different transactions in the transaction control expression be all distinct.

We are currently investigating adapting transaction control expressions for transient objects modeled as objects. We would also like to model transaction control expressions as typed classes and objects. In this way we will be able to apply specialized classes of these expressions to specialized classes of transient objects.

In concluding this discussion on separation of duties, we note that the authors in [19, 20] have alluded to the Clark-Wilson integrity model [3] and hence the need to support separation of duties. The proposal in [19] calls for an "AUTHORIZATIONS" object in the system to manage access control and separation of duties. Details on how these and other ideas can be implemented need further investigation.

3.3 Intra-object method control expressions

The scope of transaction control expressions cross object boundaries in that the transactions are public to all objects. It may be desirable in some applications, that private methods in an object (these methods are only accessible within an object boundary) be invoked in a certain sequence, and in addition for separation of duties to be enforced for intra-object accesses. We will be investigating the use of intra-object method control expressions for this purpose.

3.4 Content based authorization

The approaches surveyed in this paper generally do not address content dependent authorization issues in a clean way. This needs more investigation, especially in regard to behavioral approaches. It is not clear if an authorization should be defined in terms of the ability to invoke a certain method on an object. How do we access the object contents for content-based authorization if the method which can access the required attribute(s) cannot be invoked?

3.5 Authorization and temporal semantics

Consider the following access control/authorization requirements:

1. Let $(s_1, o_1, write)$ be true **as long as** $(s_2, o_1, read)$ is true;
2. Let $(s_1, o_1, read)$ be true **whenever** $(s_2, o_1, read)$ is false;
3. Let $(s_1, o_1, read)$ be true if only if subject s_2 has not written to object o_1 **so far**;
4. Grant authorization to subject s_1 for the last three versions created after June 12, 1993, of the versioned object o_1 , and have not been updated **since** subject s_2 was authorized to write object o_1 .

The models of discretionary access control that have been reported in the literature cannot accommodate the above constraints and requirements. The above requirements call for models that unify implicit authorizations, content-based access control, and the semantics of time.

3.6 Transaction based authorization

In all the work we have surveyed and discussed, the access control problem seeks an answer to the question: Is subject s allowed access type a on object o ? An authorization was thus seen as a 3-tuple (s, o, a) . This view of access control (and authorization) is heavily influenced by the subject-object paradigm of access control in general computer systems. We believe it is time to reexamine this view of access control in the context of databases. Why not specify authorizations in terms of transactions and objects. After all, in a strict sense it is transactions (and not subjects) that access and modify the database objects. A similar view is expressed by Clark and Wilson in [3] with transformation procedures being transactions (see rule E2), although their work needs to be adapted to object-oriented databases. We would of course expect the decision to authorize a transaction to depend on among others things, the identity and rights of the user who invokes the transaction.

Another promising research direction is the notion of an *authorization transaction* [22]. Such a transaction is one that is created for every regular database transaction, but is primarily concerned with the acquiring and management of all authorizations and access control information required to successfully commit the database transaction. In particular, this will give us the flexibility to incorporate failure semantics in the management of authorizations. Thus if a particular authorizations fails, we may be able to specify alternate authorizations to be requested.

As an illustration, consider a sales order processing system, where the processing of a sales request involves two transient objects, a purchase order and a sales order. In such an environment, we envision a nested model of authorization transactions. Every transient object is managed by an individual *authorization subtransaction* that executes the local transaction control expressions for the transient object. These subtransactions enforce separation of duties and other access control requirements on the transient object. A root authorization transaction manages these subtransactions, and further enforces separation of duties across transient objects.

4 SUMMARY AND CONCLUSIONS

In this paper we have provided a framework that breaks down discretionary access control issues into three categories: subject to user, inter-object, and intra-object. We identified some of the issues and current proposals in these categories for both structurally and behaviorally object-oriented database systems. While reasonable progress has been made, more work still needs to be done. We have identified some of the areas that warrant further research including authorizations based on separation of duties, multiple approvals, object contents, and temporal semantics. We have also argued for the advancement of transaction based authorization models. Such models would

constitute a departure from the traditional subject-object paradigm of access control, and rely on transactions as a central abstraction for specifying access control in databases.

References

- [1] F. Bancilhon, W. Kim, and H. F. Korth. A model of CAD transactions. *Proc. of the VLDB conference*, 1985, Stockholm, pp. 25-33.
- [2] E. Bertino. Data hiding and security in object-oriented databases. *Proc. of the IEEE Data Engineering Conference*, pp. 338-347.
- [3] D.D. Clark and D.R. Wilson. A comparison of commercial and military security policies. *Proc. of the IEEE Symposium on Security and Privacy*, 1987.
- [4] K.R. Dittrich, M. Hartig, and H. Pfefferle. Discretionary access control in structurally object-oriented database systems. *Database Security II, Status and Prospects*, C.E Landwehr (Editor), Elsevier Science Publishers B.V. (North-Holland)
- [5] D.B. Faatz and D.L. Spooner. Discretionary access control in object-oriented engineering database systems. *Database Security IV, Status and Prospects*, S. Jajodia and C.E Landwehr (Editors), Elsevier Science Publishers B.V. (North-Holland)
- [6] D. Fisherman. IRIS: An object-oriented database management system. *ACM Transactions on Office Information Systems*, 5(1):pp. 48-69, January 1987.
- [7] E. Gudes, H. Song, and E.B. Fernandez. Evaluation of negative, predicate, and instance-based authorization in object-oriented databases. *Database Security IV, Status and Prospects*, S. Jajodia and C.E Landwehr (Editors), Elsevier Science Publishers B.V. (North-Holland)
- [8] W. Kim, R. Lorie, D. McNabb, and W. Plouffe. A transaction mechanism for engineering design databases. *Proc. of the VLDB conference*, 1984, Singapore, pp. 355-362.
- [9] W. Kim, E. Bertino, and J.F. Garza. Composite Objects Revisited. *Proc. of the ACM-SIGMOD Intl. Conference on the Management of Data*, Portland, Oregon, 1989.
- [10] W. Kim et al. Features of the ORION object-oriented database system. In W. Kim and F. Lochovsky, editors, *Object-Oriented Concepts, Databases, and Applications*, Addison-Wesley Publ. Co., Inc., Reading, MA, 1989.
- [11] R. Lorie and W. Plouffe. Complex objects and their use in design transactions. In *Proc. Databases for Engineering Applications*, Database Week, 1983 (ACM), May 1983, pp.115-121.
- [12] D. Maier. Development of an object-oriented DBMS. In *Proc. 1st Intl. Conf. on Object-Oriented Programming Systems, Languages and Applications*, ACM, New York, 1986, pp. 472-482.
- [13] D. Maier. Why isn't there an object-oriented data model? In *Proc. of the 11th IFIP World computer conference*, San Francisco, CA, August-September, 1989, pp. 793-798.
- [14] F. Rabitti, E. Bertino, W. Kim, and D. Woelk. A model of authorization for next-generation database systems. *ACM Trans. on Database Systems*, Vol 16, No. 1, March 1991.
- [15] J. Richardson, P. Schwarz, and L. Cabrera. CACL: Efficient fine-grained protection for objects. In *Proc. Intl. Conf. on Object-Oriented Programming Systems, Languages and Applications*, ACM, New York, 1992, pp. 263-275.

- [16] R.S. Sandhu. The NTree: A two dimension partial order for protection groups. *ACM Tran. on Computer Systems*, Vol. 6, No. 2, May 1988, pp. 197-222.
- [17] R.S. Sandhu. Transaction control expressions for separation of duties. *Proc. of the Fourth Computer Security Applications Conference*, pp. 282-286, 1988.
- [18] R.S. Sandhu. Separation of duties in computerized information systems. *Database Security IV, Status and Prospects*, S. Jajodia and C.E Landwehr (Editors), Elsevier Science Publishers B.V. (North-Holland)
- [19] C.A. Schiller. Potential benefits from implementing the Clark-Wilson integrity model using an object-oriented approach. In *Proc. of the 15th National Computer Security Conference*, Baltimore, MD.
- [20] J.M. Slack and E. A. Unger. Protected groups: An approach to integrity and secrecy in an object-oriented database. In *Proc. of the 15th National Computer Security Conference*, Baltimore, MD.
- [21] D.L. Spooner The impact of inheritance on security in object-oriented database systems. *Database Security II, Status and Prospects*, C.E Landwehr (Editor), Elsevier Science Publishers B.V. (North-Holland)
- [22] R.K. Thomas and R.S. Sandhu. Task-based authorization: A paradigm for flexible and tailorable access control in distributed applications. Submitted for Publication.
- [23] T.C. Ting, S.A. Demurjian, and M.Y. Hu. Requirements, capabilities, and functionalities of user-role based security for an object-oriented design model. *Database Security V, Status and Prospects*, C.E Landwehr and S. Jajodia (Editors), Elsevier Science Publishers B.V. (North-Holland)

REGULATING PROCESSING SEQUENCES VIA OBJECT STATE *

David L. Sherman and Daniel F. Sterne

Trusted Information Systems, Inc.
3060 Washington Road (Rt. 97)
Glenwood, Maryland 21738

Abstract

Assuring the integrity and reliability of computing systems requires, in part, regulating the sequence in which particular kinds of processing occur. An access control mechanism is proposed that regulates the sequence of operations that can be applied to objects. The mechanism limits access according to the operation and the object's current state by consulting centralized tables that describe permissible state transitions. The mechanism's characteristics are examined in the context of a simple purchase order system.

1 Introduction

Assuring the integrity and reliability of computing systems requires, in part, regulating the sequence in which particular kinds of processing occur. For example, Clark and Wilson [4] state that integrity constraints commonly include requirements that "TPs [Transformation Procedures] be executed in a certain order." Similarly, Rushby [7] states that many safety and security issues can be reduced to the need for correct sequencing, citing as examples missile launch sequencing requirements (ready, aim, fire), and requirements that messages be reviewed by a release officer before being distributed.

The previous literature includes discussions of mechanisms and conceptual approaches having some potential to describe processing sequences, but these discussions have generally focused on other concerns rather than sequencing per se [5, 8, 6, 2, 1, 3]. In this paper we explore a centralized access control mechanism designed specifically to constrain processing sequences. The controls are based on the following notions:

- For each object whose validity depends on being processed by a set of applications (TPs) in a particular order, a current state indicator will be maintained.
- An object's current state changes when the object is processed by an application.
- Permissible processing sequences are represented as a table of permissible state transitions. Different sets of states and transitions are associated with different types of information.
- Impermissible transitions can be prevented by a centralized access control mechanism.

*Funded by ARPA contract DABT63-92-C-0020.

The controls have been designed to meet the following requirements:

- They must be flexible, i.e., capable of enforcing a wide variety of processing sequences [6], and must provide a system designer a natural means of describing permissible sequences.
- They must provide a centralized representation of the desired sequence controls; the control of process sequencing must not rely significantly on logic hidden in applications.
- They must be implementable using current technology in a way that satisfies TCSEC B2 or higher architectural assurance requirements. That is, the controls should be a simple extension of secure operating system technology, and should not require full-blown DBMS or transaction processing facilities.
- They must be capable of combining sequence restrictions with dynamic separation of duty constraints [8, 6].

The remainder of this paper is organized as follows. Section 2 outlines a sample application whose requirements are used to illustrate the proposed controls, which are introduced in Sections 3 and 4. Section 5 describes support for separation of duty constraints. Sections 6 and 7 discuss mutual exclusion and incomplete state transitions. Section 8 describes additional ways to restrict modification of an object. Sections 9 and 10 compare and contrast the mechanism to others in the research literature, and point out its strengths and weaknesses.

2 Application Requirements - An Example

To examine the applicability and features of the proposed controls, we present a small but non-trivial set of requirements for an automated purchase order processing system based loosely on the practices of our organization. In this system, we envision each purchase request as a state-controlled object. A purchase request can be created and submitted by any employee. The request is then modified by various application programs as it goes through approval, ordering, receiving, and payment to its final state.

We have identified several operations that one must be able to perform on a purchase request. A few of these operations have been selected to illustrate the kinds of operations that should be supported:

- Any employee of the company can create a request to purchase goods.
- A manager can approve or reject the request made by an employee based on the need for the goods. The manager is not allowed to approve his/her own request.
- A purchasing officer may sign an approved purchase request. The purchasing officer is not allowed to sign a request which he/she has either submitted or approved. (The officer does not actually sign the request but, rather, authorizes a later application to print the purchase order with the officer's signature on it.)
- The originator of a request that has been rejected modifies the request and re-submits it.

The operations must occur in an appropriate sequence. The purchasing officer cannot sign a request until it has all the necessary approvals. The operations also have separation of duty requirements (e.g., the purchasing officer is not allowed to sign a request which he/she has either initiated or approved).

3 Assumptions

We will assume that the state-based controls being proposed are not used in isolation, but are an adjunct to other access controls that exist on a system. For example, a system may provide access controls based on types and domains [3, 9]. We will assume that these other access controls will perform the following functions:

- Limit a user's ability to execute various application programs.
- Limit an application program's ability to access objects according to the type of object and mode of access.

We will assume these other controls apply to all objects, whether they are state-controlled or not. Furthermore, they describe which applications are allowed to modify which types of state-controlled objects. The state-based controls describe when and how each of those applications is allowed to modify the state of the object.

The modification of a state-controlled object consists of two parts: a transition of the object's state, and a change of the object's data content. By definition, every modification of a state-controlled object includes a transition of the object's state, even if the transition is back to the same state. The modification of the object's data is optional. If an application intends to modify the data of a state-controlled object, it must obtain permission to change an object's state before it can obtain permission to modify the object's data.

4 Sequencing

The object-state-based access control mechanism allows for a system to have many types of state-controlled objects and for each type of object [3] to have its own set of defined states. Every state-controlled object must be in one of the states allowable for that type of state-controlled object. Each type of state-controlled object has a state-transition table which describes the access controls to be applied to the modification of objects of that type. Each application which can modify that type of object has one or more rows in the state-transition table. Each row in the table contains the following information:

- Operation: The application that is allowed to modify an object of this state-controlled type.
- From State: The state (or states) in which the object must be for the indicated operation to be allowed. A single state can be specified, or a list of states can be specified. If the object is not in one of the specified states, the application is not allowed to change the object's state or modify the object's contents.
- To State: The state (or states) to which the application is allowed to change the object. A single state can be specified, or a list of states can be specified. Attempts to change the object to states other than the one(s) specified will be denied.

One type of state-controlled object in the hypothetical purchase order system is the purchase request. A portion of the state-transition table for purchase requests would look like:

Operation	From State	To State
Create_Request	NULL	Submitted
Approve_Request	Submitted	Approved Rejected
Sign_Request	Approved	Signed Rejected
Modify_Request	Rejected	Submitted

Sequencing of operations is accomplished by restricting the allowable sequence of object states. Each application's access is limited to certain states, and each state can only be produced by certain applications.

The first row of the state table in the example above is for the Create_Request application, which allows an employee to create a request to purchase goods. The entry for the From State is "NULL" since there is no existing state for a purchase request which has not yet been created. The value "NULL" specifies that the application is allowed to create a purchase request, as opposed to modifying an existing request in some specified state. The entry for the To State indicates that the new purchase request can only be placed in the Submitted state. Create_Request is the only application that can create a purchase request. Create_Request and Modify_Request are the only applications that can cause a request to be in the Submitted state.

The second line of the state table example shows the Approve_Request application which allows a manager to either approve or reject a request which has been submitted. The From State entry allows the application to modify a purchase request only when the request is in the Submitted state. This means that the input to Approve_Request can only have come from Create_Request or Modify_Request. The To State entry specifies that the application can change the state of the request to either Approved or Rejected, but to no other states. Only the Approve_Request application can produce an Approved purchase request.

The Sign_Request application allows a purchasing officer to sign or to reject a purchase request. The application can change an Approved request to either Signed or Rejected. Any purchase request operated on by Sign_Request must have been processed first by Create_Request and then by Approve_Request. Not all purchase requests created by Create_Request will get to Sign_Request, as some will be rejected along the way. But all requests which get to Sign_Request must have come through the pair of applications described. There is no other way for a purchase request to be in the Approved state. This set of constraints is similar to the "assured pipeline" described in [3].

The Modify_Request application allows the originator of a request which has been rejected to modify the request and re-submit it. At this point the modified request is treated exactly like a new request just created by Submit_Request. The From State restricts the application to purchase requests that have been rejected. The To State requires that the purchase request begin the approval process over again by going back to the Submitted state.

Although many desired sequences may be sequential in nature, the ability to specify arbitrary sets of From States and To States allows a wide variety of cyclical and branching sequences to be specified. In particular, requirements cited in [6] associated with crossing out authorization signatures so that transactions can be resigned by alternative authorities are easily supported.

5 Separation Of Duty

Many operations in the system can normally be performed by an individual, but not if he/she has performed an earlier operation on the purchase request. For example, a manager is not allowed to approve a purchase

request which he/she has submitted. When a state-controlled object has separation of duty requirements, the state table has two additional fields that enable the system designer to specify the limitations:

- **Identifier:** The Identifier field is used to associate an arbitrary symbolic identifier with the individual who performs this operation, in order to:
 - Prevent this individual from performing some later operation, or
 - Allow only this individual to perform some later operation.

The identifier is used only to relate two or more rows within the state table and has no meaning outside the state table.

- **Separation of Duty:** The Separation of Duty field is used to indicate that the individual performing this operation either must be or must not be the same individual who performed some previous operation on the object.

Let us add to our purchase order system the requirement that each purchase request be approved by two managers. The state table entries for some of the operations would be:

Operation	From State	To State	Identifier	Separation of Duty
Create_Request	NULL	Submitted	Requester	
Approve_Request	Submitted	Approved1 Rejected	Manager1	NOT Requester
Approve_Request	Approved1	Approved2 Rejected	Manager2	NOT Requester AND NOT Manager1
Modify_Request	Rejected	Submitted		Requester

The person who runs the Create_Request application for a given purchase request becomes known as Requester for determining future access to that purchase request. The first person who runs the Approve_Request application on a given purchase request (when the request is in the Submitted state) becomes known as Manager1 and must not be the same person as Requester, the person who originated the request. The second person who runs the Approve_Request application on a given request (when the request is in the Approved1 state) cannot be the originator or the first approving manager. The only individual allowed to modify a purchase request via the Modify_Request application is Requester, the person who ran the Create_Request application to originate the request.

A Separation Of Duty Log is maintained for each state-controlled object that has separation of duty requirements. The log is a record of who has modified the object. An entry is added to the log each time an operation causes a state change to occur. Note that the access log is used strictly for access control and is not used for detailed auditing. An entry in the log contains the following fields:

- **State:** The state produced by the modification.
- **Identifier:** Specifies the separation of duty identifier associated with this modification. This field is copied from the Identifier field of the line in the state table for the application which produced this modification.
- **Person:** Identifies the individual who executed the application to produce this state. This field is blank whenever the identifier field is blank.

Note that the name of the application producing the modification is not present in the log entry since it is not needed.

If Washington were to run the Create_Request application to create a purchase request, the first log entry for the request would look like:

State	Identifier	Person
Submitted	Requester	Washington

When Adams runs the Approve_Request application on this purchase request, the access control mechanism would see from the Separation Of Duty field in the state table entry for this application that he is not allowed to be Requester. The log shows that Requester is Washington. Since Adams is not Washington then Adams is allowed to run the application. The log would then contain two entries:

State	Identifier	Person
Submitted	Requester	Washington
Approved1	Manager1	Adams

Since Washington is identified as the Requester and Adams is Manager1, neither of them can execute the second Approve_Request application on this purchase request since the state table entry for Approve_Request contains the Separation of Duty requirement "NOT Requester AND NOT Manager1." When Jefferson rejects the request the log looks like:

State	Identifier	Person
Submitted	Requester	Washington
Approved1	Manager1	Adams
Rejected	Manager2	Jefferson

When the state transfer is to a state which the object has previously occupied, the system truncates the Separation Of Duty Log back to the entry for that state. All the operations performed on the object from the time that it left this state until the time that it arrives back in the state will be removed from the log. Recall that this log is used only for access control and not for auditing.

In the example above, Washington is the only person who can run the Modify_Request application to fix the rejected request since Washington is the Requester. When Washington modifies the request, the state of the request is once again set to Submitted. Since the request has previously been in the Submitted state, the Separation of Duty log is rolled back to the entry for that state. This results in the log consisting only of its initial entry:

State	Identifier	Person
Submitted	Requester	Washington

The separation of duty log no longer contains a record of who approved the request the first time, as this information is not needed for future access control decisions. (The contents of the purchase request itself or of a detailed audit trail, however, may contain it.) The request could be approved by Adams again or by some other manager. The only separation of duty requirement that remains is that Washington not approve his own request.

Consider adding a requirement that the second submission of the purchase request must be signed by the same manager who signed it the first time. In this case the object state resulting from the second submission would not be the same as the state produced by the original submission. The two states would have different separation of duty requirements. The Modify_Request application would have to cause a transition to a different state, such as Resubmitted. The access log would not be truncated, allowing future access control decisions to be based on the original submission.

6 Mutual Exclusion

Let us consider the Approve_Request operation in a little greater detail. Approve_Request operates on a purchase request which some employee has previously created in the Submitted state. While running the Approve_Request application, a manager will display the request, decide whether to approve or reject the request, update the request appropriately, and set the new state of the request to either Approved or Rejected. It is also possible that after seeing the request, the manager can decide that he/she does not have enough information to either approve or reject the request at this time and can choose to leave the request unchanged for now.

In order to reliably change the object from its old state to a new state, one must avoid the problems which can be caused by multiple accessors of the object attempting to modify it at the same time. Each application must:

```
Lock the object for exclusive use.
Obtain read access to the data content of the object.
Examine the existing contents of the object.
Decide what the new state should be.
IF the object is to be modified
THEN
    Obtain permission to change the object's state to a specified new state.
    IF the data content of the object is also to be changed.
    THEN
        Obtain write access to the data content of the object.
        Modify the contents of the object.
    ENDIF
ENDIF
Relinquish access to the data content of the object.
(Relinquishing access will cause any requested state change to occur.)
Unlock the object.
```

The first step for the application is to lock the object. All modifications of a state-controlled object require exclusive access to the object. This prevents two or more subjects from simultaneously updating the object. Changing the state of an object is in essence a modification of the object and requires exclusive access to the object even if the object's data content is not being changed!

Having locked the object, the application can examine the object's contents to make any necessary decisions about how to process the object. Since all other applications are prevented from changing the object, the application is assured that its actions on the object are not affected by other applications attempting to modify it. Should the application determine that it cannot change the object, it can close access to the object at this point. Since the application was only granted read access to the data, and never requested permission to change the state, the system is assured that the object has not changed and the system will leave the object's state unchanged. In the hypothetical purchase order system, if the Approve_Request application closes access to the purchase request before obtaining permission to modify its state, the state of the purchase request will remain Submitted.

Once an application has opened an object and determined that changing the object is appropriate, it makes a request to the system for permission to change the state of the object by announcing the state to which the object is going to be changed. The Approve_Request application would announce whether it intends to Approve or Reject the request. If the state table indicates that the application is allowed to make the specified state transition (from its current state to the announced state), then the system grants the application permission to modify the state and internally records the tentative new state. The lock on the file for exclusive use is retained by the application. If the application needs to modify the data content of

the object then the application requests the appropriate permission (append, write, etc.) to the object. If permission is granted then the application is free to modify the object's contents. If an application requests permission to modify the contents of a state-controlled object without having first obtained permission to modify the object's state, the request will be denied.

When the application relinquishes access to the object, the application must specify whether or not it was able to complete its operation on the object. If the application was successful, the system sets the new state of the object to the previously declared value. If the application was not successful, or if the application terminates prematurely, then the system sets the state of the object according to the Error State field, as described in the next section.

Finally, the application unlocks the object.

7 Incomplete State Transitions

In concept, all state transitions occur instantaneously. To an observer of a state controlled object, the object is always in one state or another, and never "in between." In practice, state transitions do not occur instantaneously. Although use of exclusive locks prevents inadvertent visibility into an object during state transitions, an application that aborts prematurely or is unable to perform all necessary actions while implementing a transition could leave an object in an undefined state.

To address this situation, the state transition table also includes an Error State column. Should a process obtain permission to change the state of an object and then fail to close it successfully, the state of the object will be set to the error state specified in the state-transition table. The system designer is free to specify as the Error State a value that the application cannot set under normal conditions.

State table entries for the purchase order system might include:

Operation	From State	To State	Error State	Identifier	Separation of Duty
Create_Request	NULL	Submitted	Incomplete	Requester	
Modify_Request	Incomplete Rejected	Submitted	FROM		Requester
Withdraw_Request	Submitted Approved1 Approved2	Rejected	Rejected		Requester
Trusted_Repair	*	*	ROLLBACK		NOT Requester AND NOT Manager1 AND NOT Manager2

A failure of the Create_Request application results in the partially created purchase request being set in the Incomplete state.

The row for the Modify_Request application has been expanded to allow it to process Incomplete requests as well as Rejected ones. The value "FROM" in the Error State field for this row specifies that when Modify_Request fails to complete a modification of a purchase request, the state will be set to the value from which the application was attempting to make a state change. That is, the state of the purchase request will not be changed; an Incomplete request will remain Incomplete and a Rejected request will remain Rejected. Only the state is returned to its value before the modification; any data contents which have been modified will remain changed.

Care should be exercised when using "FROM" since:

- the application may have modified the object's contents without changing its externally visible state.
- the application can now (covertly) produce objects which are in a state that it might not otherwise be authorized to produce.

Consequently, use of the value "FROM" is discouraged. In the case of `Modify_Request`, the use of "FROM" is acceptable since the purchase request has not yet begun/restarted its approval sequence.

The `Withdraw_Request` application allows the Requester of a purchase request to withdraw the request after it was submitted, so long as it has not yet been signed by the purchasing officer. If `Withdraw_Request` terminates before completing its update to the `Rejected` state, the state is still set to `Rejected`.

The `Trusted_Repair` application allows an accounting administrator to repair a purchase request that may have been improperly processed by an application and left in some unforeseen condition which might prevent further processing. The wildcard symbol "*" in the `From State` and `To State` fields of this row of the state table indicates that the application can make any state change to a purchase request. The trusted application can change the request from any state to any state. The special value "ROLLBACK" in the `Error State` field specifies that the data contents of the object should be automatically backed-up on the open and that both the state and the data contents should be reset to their original values when the application fails to complete the modification successfully. Use of this value is encouraged whenever possible. In order to avoid automatically imposing the overhead of backup for every potential state transition, the system requires the designer to specify "ROLLBACK" wherever it is needed.

One deficiency in these error recovery facilities is that they deal with each state-controlled object in isolation. Consider an application to print a batch of checks for each purchase request whose state is `Payable`, changing the state of each to `Complete`. The application must also debit an accounting ledger for each check printed. Suppose the application terminates abnormally while changing the state of a purchase request. Further suppose that "ROLLBACK" had been specified as the error state for the application. If at the time of the abnormal termination, neither the check had been printed nor the ledger debited, restoring the previous state of the purchase request to `Payable` and restoring its previous contents would be appropriate. However, if either of these processing steps had already been completed when the failure occurred, then a rollback would be inappropriate, and might lead to double payment or duplicate ledger entries. Similarly, under other circumstances, if `Complete` had been specified as the error state for this application, other problems could occur, i.e., failure to ever print a check or debit the ledger. In this case, proper recovery seems to require human inspection and decision making via special application.

These problems could be addressed in part by defining intermediate states `Check_Printed` and `Ledger_Debited` and giving the check printing application authority to manipulate purchase request objects in these states. A more fundamental concern is that copying and preserving a single state-controlled object does not provide a truly adequate rollback capability. What is ideally needed is a *transaction rollback* facility that would undo all internal effects of running the check printing application, including the ledger debit. Traditional DBMS and transaction processing systems provide such capabilities, but are generally regarded as too large and complex to satisfy the modularity, least privilege, and other architectural requirements imposed at the higher evaluation classes of the TCSEC. Our motivation for the mechanisms proposed here is in part a desire to see how far mechanisms oriented towards TCSEC principles can be stretched to solve real world integrity and reliability problems. In this example cited, we may not have succeeded in stretching them far enough. We note, however, in concluding this section, that even full-blown transaction rollback facilities cannot undo the printing of checks. There will always remain certain kinds of system failures whose recovery will require assistance and supervision by human beings.

8 Restricting Modes of Modifications

We have assumed the existence of other access controls, such as domain and type enforcement, which pertain to all objects, not just state-controlled objects. Those controls would specify the modes of access that an application would have to an object. Examples could include: create, read, write, append, delete. By combining these access controls with the object-state controls, the system designer can more precisely control the modification of state-controlled objects. Consider the following examples:

- Read/write capability allows the application to change the data content of the object as well as its state.
- Append-only access allows information to be added to the end of the existing data, but prevents existing data from being modified. In the Approve_Request application example, a reason for the rejection or approval could be added, but the original request could not be modified.
- Read-only access restricts the application's role to that of a filter that either passes or rejects objects. The application can change the state of the object, but cannot change any of the data contents of the object.

In general, a state-controlled object cannot be copied. Although the data content can be copied, the resulting object is not a state-controlled object. This restriction is necessary to prevent objects from being created in the middle of a processing sequence. This requires no additional mechanism beyond the to/from state restrictions described herein.

9 Related Work

The sequence controls proposed here are an elaboration of the controls described by Sterne [9]. They are an outgrowth of attempting to improve on previous proposals for implementing separation of duty and other integrity constraints. A number of these proposals were stimulated by the Clark-Wilson model [4].

Karger [5] proposes extending a secure capability architecture so that it provides specialized access control lists (ACLs) tailored for expressing separation of duty constraints. These ACLs are depicted as being attached to TPs. They can be used to specify that a user should be prevented from executing one TP unless a different user has previously executed another specified TP. The enforcement of the ACLs relies on references to "token capabilities," which are equivalent to the separation of duty logs described here. Karger's paper does not show that the ACL mechanism provides the ability to regulate execution of a TP according to the *state of the particular object* to which it is applied. The paper explicitly avoids discussion of locking and TP error recovery. A drawback of the approach is that it lacks a centralized description of the sequencing and separation of duty constraints in effect. Understanding these requires inspecting the access control lists attached to each TP and data item.

The ideas presented here are most closely derived from Sandhu's transaction control expressions [8]. Sandhu, however, also avoids discussion of locking and error recovery by assuming that TPs are full-fledged transactions, exhibiting serializability and failure atomicity. This implies that they are implemented atop a DBMS or transaction processing system, an assumption at odds with this paper's goal of near term implementability consistent with TCSEC high assurance architecture requirements. The notations proposed by Karger and Sandhu do not appear capable of expressing the range of cyclical and alternative sequencing requirements supported by the mechanisms proposed here. A paper by Nash and Poland [6] discusses a similar idea and points out the need to support "undoing" a processing step (backtracking) under certain circumstances.

Bell [2] has also explored supporting separation of duty and sequencing, using an abstract "Universal Lattice Machine" (ULM). In Bell's approach, TPs are relied upon to add and remove negative access control list

entries (NACLs) to or from individual data objects in such a way that different individuals are prohibited from accessing them at particular times. This approach also lacks a centralized description of sequencing and separation of duty constraints, which are distributed and embedded in the logic of individual TPs.

Badger [1] has proposed a model for describing a variety of integrity requirements including sequencing requirements. The model assumes as a basis the nested transaction paradigm, including serializability and failure atomicity properties.

The domain enforcement mechanism proposed by Boebert and Kain [3] predates the Clark and Wilson model. Boebert and Kain describe its use in building an "assured pipeline" so that every print file is properly labeled with its security classification at the top and bottom of each page. Since the labels must be attached to each file before printing, this represents a form of sequencing requirement. The mechanism associates a domain attribute with each subject and a data type with each object, and constrains the type of objects that can be read and created according to a domain table. The domain table can be set up to force certain types of objects to flow through particular sequences of subjects, which in turn can only execute particular programs.

While a complete comparison of state-based controls and domain enforcement is beyond the scope of this paper, it appears that object state-based controls offer some useful features not readily apparent in domain-based controls.

- A state-controlled object (e.g., an approved purchase order) cannot be fabricated out of thin air by an application in the middle of a sequence. Except for the initial application in a sequence, each application can only produce its output after having received as input an object in the appropriate state.
- Applications need not be given the ability to modify an object in order to carry out a step in processing the object and changing its state. This simplifies the task of assuring that a pass/fail approval filter does not inappropriately modify the object during the approval process.
- Separation of duty constraints are readily integrated with state-based sequencing constraints, i.e., a single integrated mechanism can enforce both sets of constraints.

10 Summary

The proposed controls appear to satisfy the objectives identified in the introduction.

- The controls appear to be highly flexible, and are capable of expressing sequential, alternative, and loop-back sequences. The system designer can tightly constrain the object states that can be produced by each application, or can provide individual applications arbitrary latitude in determining output states on an object-by-object basis. Given familiarity with the concept of state transitions, the controls seem reasonably intuitive and natural.
- The controls provide a centralized representation of the desired sequence constraints. A set of state transition tables, one for each type of state controlled object, represents desired constraints for the entire system and governs their enforcement.
- The controls do not presuppose the existence of an underlying DBMS or transaction processing facilities. They appear to be implementable using secure operating system current technology in a way that satisfies TCSEC B2 or higher architectural assurance requirements. On the other hand, without failure atomicity features, the error recovery capabilities that can be provided are somewhat primitive.
- The controls provide a single integrated approach to enforcing sequence restrictions and separation of duty constraints.

Assuring the integrity and reliability of computing systems in part requires regulating the order in which particular kinds of processing occur. We have proposed an access control mechanism designed to regulate sequences of accesses to objects based on their current states by consulting centralized tables that describe permissible state transitions. We examined the use of the mechanism in the context of a simple purchase order processing system. The mechanism has been designed to support specification of complex processing sequences in a natural manner, to support separation of duty constraints, and to be implementable with high assurance. The mechanism provides useful but rudimentary error recovery facilities. In addition, we've compared the ideas underlying the proposed mechanism with those in the research literature.

References

- [1] L. Badger. "A Model for Specifying Multi-Granularity Integrity Policies." In *Proc. 1989 IEEE Symposium on Security and Privacy*, pages 269-277, Oakland, California, May 1989.
- [2] D. Bell. "Lattices, Policies, and Implementations." In *Proc. 13th National Computer Security Conference*, pages 165-171, Washington, D.C., October 1990.
- [3] W. E. Boebert and R. Y. Kain. "A Practical Alternative to Hierarchical Integrity Policies." In *Proc. 8th National Computer Security Conference*, pages 18-27, Gaithersburg, MD, September 1985.
- [4] D. Clark and D. Wilson. "A Comparison of Commercial and Military Computer Security Policies." In *Proc. 1987 IEEE Symposium on Security and Privacy*, pages 184-194, Oakland, CA, April 1987.
- [5] P. Karger. "Implementing Commercial Data Integrity with Secure Capabilities." In *Proc. 1988 IEEE Symposium on Security and Privacy*, pages 130-139, Oakland, CA, April 1988.
- [6] M.J. Nash and K.R. Poland. "Some Conundrums Concerning Separation of Duty." In *Proc. 1990 IEEE Symposium on Security and Privacy*, pages 201-207, Oakland, CA, May 1990.
- [7] J. Rushby. "Kernels for Safety?" In *Safe and Secure Computing Systems*, pages 210-220, Glasgow, UK, October 1986.
- [8] R. Sandhu. "Transaction Control Expressions for Separation of Duty." In *Fourth Annual Computer Security Applications Conference*, pages 282-286, Orlando, FL, December 1988.
- [9] D.F. Sterne. "A TCB Subset for Integrity and Role-Based Access Control." In *Proc. 15th National Computer Security Conference*, pages 680-696, Baltimore, MD, October 1992.

RENEWED UNDERSTANDING OF ACCESS CONTROL POLICIES¹

Marshall D. Abrams

The MITRE Corporation, 7525 Colshire Drive, McLean, VA 22102

Abstract

Access control policies must be viewed in a modern perspective to support the evolution of information technology (IT) security evaluation criteria. This paper provides observations and definitions that coalesce policy research and development. Traditional definitions and assumptions are extended.

Keywords: Access Control Decision Information, Access Control Policies, Attributes, Availability, Confidentiality, Discretionary, Groups, Identity, Inheritance, Integrity, Malicious modification, Mandatory, Non-discretionary, Policy, Process, Roles, Rules, TCSEC, Trusted Computing Base, Type enforcement, Well-formed transactions

1. Introduction

The purpose of this paper is to raise the level of awareness that the conceptual framework for access control built into the *Department of Defense Trusted Computer System Evaluation Criteria* (TCSEC) [TCSE85] needs to be extended. This paper presents the author's selection of access control policy issues in order to develop the modern perspective necessary to extend the TCSEC conceptual framework. Most of the concepts presented in this paper have appeared in prior research and development (R&D) work, but have not made their way into the thinking of enough IT security practitioners. Some readers may find little stimulation in this paper; others may be incited to near violence.

We describe four non-discretionary Access Control Policies as examples of evolving thought. All of these policies include implementation of a policy similar to Originator Controlled (ORCON), employing some form of non-discretionary access control list. We emphasize the importance of associating Access Control Decision Information (ADI)² with programs and processes when second-order effects of malicious code are considered.

2. Access Control Concepts

Access control is pervasive to practically all IT security. There are quite a few Access Control Policies and mechanisms. This section addresses a set of Access Control Policies that cover a reasonably representative group of access control functions.

Information technology security is defined in [COMM91] to mean the maintenance of confidentiality, integrity, and availability.³ *Integrity* has traditionally been applied in the security community to both data and systems. *System integrity* refers to the quality of the hardware and software that implements a secure system (i.e., that the hardware and software operate as expected).

To Biba, *program integrity* means that programs can be invoked only by programs that are lower or equal in integrity [BIBA77], thereby preventing corruption of higher integrity programs [SHIR81]. Schell [SCHE86] showed that program integrity is just a special case of *data integrity*. Program integrity also includes freedom from modification by malicious code.⁴

Availability differs in kind from the other two components of IT security. Availability cannot be enforced by access controls. It is easy to see that a process may, in general, consume resources in a manner that may prevent other processes from accessing those resources when needed. The observation that a *runaway process* can waste resources,

¹ This work was funded by The MITRE Corporation and the Department of Defense under contract DAAB07-93-C-N651.

² The term Access-Control Decision Information is introduced in [ISO92]. Strictly speaking, TCSEC *labels* are one special case of ADI. Nevertheless, we use ADI as the more general term in this paper.

³ Achieving an acceptable definition of integrity remains elusive; see [INTE91].

⁴ In this paper, *malicious code* includes Trojan horses, viruses, and worms, for example. Unless absolutely necessary, the exact malicious mechanism will not be discussed.

even in a system that implements access controls, was made in [LAMP71, LAMP91]. Quota mechanisms can help, but these are not access controls.

The *kinds* of access mentioned in the Anderson Report [ANDE72] (called *modes* in the TCSEC) were those offered by Multics [ORGA72]. Other platforms offer different kinds of access, including execute and append (both without observe access). Applying the Multics interpretation of the Bell-LaPadula model [BELL75] to dissimilar platforms does not lead to the insight that formal modeling should, and may, be misleading, if not wrong, because of the differences in the platform.

The reference monitor creates subjects and objects as abstractions to manage the IT resources. Subjects are processes, executing in a particular domain,⁵ that request access to passive objects. Many subjects are acting out the wishes of a human user of the IT system. It has been traditional to associate the ADI of these users (e.g., clearance) with the processes acting on their behalf. In some parts of the tradition, such as Biba integrity, ADI have also been associated with processes. Other processes are performing system functions, generally without concern for individual user identities [FRAI83, ABRA91].

3. Mandatory, Discretionary, and Non-Discretionary Policies and Mechanisms

3.1 Traditional Discretionary and Mandatory Access Control

Traditionally Access Control Policies have been divided into two classes: *discretionary* and *non-discretionary*, sometimes called *mandatory*.⁶ The TCSEC defines discretionary access control as “a means of restricting access to objects based on the identity of the subjects and/or groups to which they belong. See [DOWN85] for a classic discussion of issues. The controls are discretionary in the sense that a subject with a certain access permission is capable of passing that permission (perhaps) indirectly on to any other subject.” Traditional mandatory policy implies that the authorization is outside the control of a typical user [SALT75]. In traditional usage, mandatory is the complement of discretionary.

Traditionally, *mandatory* identifies control of policy rules and the ADI employed by those rules being vested in the security administrator's role,⁷ and *discretionary* bound identifies a situation in which authority is not (well) controlled. (Roles are also discussed below.) There are many policy implementations that can satisfy the TCSEC definition of discretionary [NCSC87]; most allow users who can read information to make that information available to other users at their own discretion.

The TCSEC definition of *mandatory security* policy and mechanisms is that mandatory security works by associating ADI with objects to reflect their sensitivity. Similar ADI is associated with subjects to reflect their authorizations. The reference monitor compares the ADI associated with subjects and objects, and grants a subject access to an object only if the result of the comparison indicates that the access is proper for a given security policy.

The dominance comparison described in the TCSEC satisfies three well-known mathematical conditions: (1) reflexivity, (2) antisymmetry, and (3) transitivity. Dominance reflects a set of rules for comparing access classes. Depending on the security policy being enforced, some flows are allowed and others forbidden. This concept of information flow policy was formally defined by Denning [DENN76].

3.2 A New Definition of Non-Discretionary Access Control

In this paper we use *non-discretionary* to identify situations in which authority is vested in some users, but there are controls on delegation and propagation of authority. If one envisions an authority tree rooted in the security administrator, then *mandatory* is the case in which the tree has no branches, *discretionary* is the case in which the branches extend to every user, and *non-discretionary* is the case in which there are branches that do not extend to every user. [SALT75] illustrates this point quite clearly with hierarchy of controllers and non-discretionary Access Control List (ACL) use.

⁵ The *domain* of a process is defined in the TCSEC to be the set of objects to which the process currently has the right to gain access.

⁶ In this paper, we differentiate *non-discretionary* from *mandatory*; the adjective *traditional* is used when necessary to refer to older, non-differentiated usage.

⁷ See [NCSC92] for a guide to the security administrator's responsibilities and relationships to other roles in the U.S. Department of Defense.

3.3 Type Enforcement

Type enforcement employs ADI as the basis of authorization for information flow. Type enforcement is not transitive, but it does satisfy at least the first Denning axiom that the set of security classes is finite. Type enforcement allows information to flow from entity A to entity B through process C if the types of A, B, and C authorize that flow. LOCK [BOEB88] employs type enforcement to provide pipelines. Pipelines can be used to implement functions that would be called trusted in a Bell-LaPadula architecture.

Work at Carnegie Mellon University on type enforcement contemporaneous with Denning's was not addressed in the TCSEC. Jones [JONE73] showed that type protection mechanisms can be used for various aspects of (non-lattice based) memoryless subsystem problems (i.e., variants of the confinement problem). Jones and Wulf [JONE75] show that type protection can support non-discretionary access controls that can be represented as a lattice. Cohen and Jefferson [COHE75] illustrate that type enforcement can support a variety of non-discretionary policies that cannot be represented as a lattice.

3.4 Authority, Global, and Persistent

There must be clear lines of authority controlling IT system security. There is no mandate that the lines of authority must be hierarchically organized. Any organizational structure may be implemented; one that reflects one of the structures of the real world is a reasonable choice. It is important to understand the delegation of responsibility and authority for a given structure to establish access control rules and to the enter values of ADI that these rules use in making access control decisions.

Mandatory security policies are traditionally characterized as being global and persistent, which is understood to mean that the policies apply to all "ordinary" users and cannot be changed, except, perhaps, by users authorized to take on a role, such as security administrator. [ABRA90] discussed two sets of rules that support the properties of being global and persistent: inheritance and authority.

A fundamental weakness of the TCSEC requirements is that they only attempt to control access to containers, not to the information contained. In particular, discretionary requirements do not include explicit inheritance rules that cause ADI to propagate when information is copied from one container to another.

Authorization is a major constituent of any non-discretionary policy. Policy for delegating authority must be explicit. See the discussion in [ABRA90]. Consider the case in which the authorized user is not part of the security administration. A project leader, for example, may establish ADI associated with objects related to his or her responsibilities. This ADI may serve the same function as traditional labels but is under the control of some user, not the security administrator. We can even assume that the project leader acts with authority delegated by the security administrator. [FLIN90] argues that the project leader may exercise better judgment, being more knowledgeable about the information and having a greater personal interest in the object(s) being protected.

3.5 Groups and Roles

A *group* may be defined as a set of users [ISO92]. A *role* is a set of allowed actions. A role allows selected users to apply specified operands to specified objects. A role is typically defined by a set of privileges and a corresponding group of users that are afforded these privileges.

A particularly safe design is to restrict a person acting in a role to executing a well-defined set of role-support procedures needed to carry out the functions of that role. This binding of programs and data is essentially the approach found in [CLAR87]. When the functions and privileges associated with a role are well defined, it may be possible to define a role completely by the transactions it permits. Systems evaluated at the higher levels of the TCSEC provide a set of procedures to implement the security administrator's role [FRAI83].

The name of a group is a form of indirect reference to its members. It is generally more convenient to use the name of the group than to itemize the individual members. When the group and object(s) are controlled by different authorities, there is actually shared responsibility. The use of defined groups in specifying access control may go beyond convenience. The group mechanism directly supports delegation of authority. For example, a defined group may provide a way for the authority who controls access to the resource to selectively delegate access control decisions to the authority who controls the composition of the role or group. This delegation may support lines of authority that could not otherwise be supported by a system based solely on a traditional hierarchical organization.

As discussed above, mandatory access control policy exists when groups and roles are controlled by the security administrator. This is the assumption in [FERR92], but authority over groups and roles may be delegated; see [ABRA91] for an example.

A common challenge in designing roles is to ensure *separation of duty*. Certain actions are sufficiently vulnerable to abuse that no single user should have authorization to perform them. In this case, it is necessary to design distinct roles that ensure separation of functions among two or more individuals acting in these roles while retaining shared responsibility and accountability. Minimum and maximum elapsed time between the separate actions may be specified.

For example, a corporate policy might require that to authorize expenditures, two *signatures* are necessary from a particular group of individuals. Another policy might require that one member in each of two different groups (but not the same individual) be a signatory to the expenditure. Another policy might require that all members of a *board* or *panel* concur to authorize an expenditure. In this last case, one group may be authorized to create a target of type *proposed expenditure*, while another group may be authorized to convert the type of this object to *authorized expenditure* by using a specialized action that checks to be sure its invoker differs from the owner of its object.

There is no generally agreed-upon definition describing how separation of duty should be implemented and how the separation of duty relation should be maintained. [CLAR87] propose that it be determined external to the secure system and encoded in an access control triple of the form: (UserID, TPi, (CDIa, CD Ib, CD Ic, . . .)), which relates a user, a transformation procedure (TP), and the data objects that TP may reference on behalf of that user. [MURR87] describes a system in which a conflict matrix is used; each transaction has a set of associated transactions that the transaction's creator has determined to be conflicting. No single user may execute both a transaction and one of its conflicting transactions.

Separation of duty can be either static or dynamic. Compliance with static separation requirements can be implemented simply by the assignment of individuals to roles and allocation of transactions to roles. Dynamic separation of duties constrains access of a subject to an object based on the previous access history of either subject or object [CLAR87, KARG88, SAND88].

4. Non-Discretionary Access Control Policy Examples

Four non-discretionary Access Control Policies are described in chronological order of publication as examples of evolving thought on non-discretionary Access Control Policies.

Director of Central Intelligence Directive (DCID) 1/7 [DCID81] specifies several policies for control of dissemination of paper documents, as an IT policy to control the dissemination of information. Although DCID 1/7 preceded the TCSEC, the policies it specifies are not (well) addressed by the TCSEC, or in IT systems built to the TCSEC paradigm. The Originator Controlled (ORCON) policy is cited by all examples as motivation for developing non-discretionary access controls that extend the TCSEC paradigm.⁸

ORCON is only one of a number of restrictive control markings defined in DCID 1/7. These markings represent handling policies that limit the authority of recipients of the information to use or transmit it. ORCON requires the permission of the originator to distribute information beyond the original receivers designated by the originator. For the purposes of this discussion, the following extract from DCID 1/7 defines the ORCON marking: "This marking is used, with a security classification, to enable a continuing knowledge and supervision by the originator of the use made of the information involved. Information bearing this marking may not be disseminated beyond the headquarters elements⁹ of the recipient organizations and may not be incorporated in whole or in part into other reports or briefings without the advance permission of and under conditions specified by the originator."

⁸ The earliest work in this area known to the author was conducted by K. Rogers (then) of UNISYS in 1986. Unfortunately, no public reference is available.

⁹ At the discretion of the originator, the term *headquarters elements* may include specified subordinate intelligence-producing components.

4.1 Propagated Access Control (PAC)

The PAC policy and the related Propagated Access Control List (PACL) [GRAU89] were proposed as one way of implementing ORCON. Whenever an authorized subject reads an object with an associated PACL, that PACL becomes associated with the subject. Any new object created by the subject inherits the PACL. PACLs are associated with both subjects and objects.

PACLs include the precedence policy taken from DCID 1/7 that specificity takes precedence over generality (e.g., NO FOREIGN plus REL Canada means that the information is releasable to U.S. and Canadian citizens).

PACLs can be combined. If a subject inherits PACL_A from object_A and PACL_B from object_B, the two PACLs are logically ANDed together to form PACL_AB, which is more restrictive than either PACL_A or PACL_B, containing only those users common to both original PACLs. The permission of the originators of object_A and object_B is needed to release any data to any new subject.

Since PACLs on processes represent the data currently in the address space of the process, the PACL can be nullified by purging this address space.

4.2 Owner-Retained Access Control (ORAC)

ORAC [MCCO90] is similar to PAC in propagating ACLs with non-discretionary enforcement. ORAC goes further, retaining the autonomy of all originators associated with a given object in making access decisions, while basing mediation of requests on the intersection of the access rights that have been granted. ORAC is motivated to implement several of the DCID 1/7 policies in addition to ORCON, namely NO CONTRACTOR, NO FOREIGN, and RELEASABLE TO.

ORAC includes dissemination controls as part of the ADI, which it refers to as labels for historical reasons. The ADI also includes an originator identification and an ACL. The ACL contains originator-designated exceptions to the dissemination controls. Members of the list, who may be individuals or groups, are explicitly identified as allowed or denied access to the data. The originator is allowed to modify the ACLs at any time. When the real-world originator is an organization, as in the ORCON policy, an originator role replaces the individual originator, with authorized individuals performing the role.

When two objects are joined, the new object inherits the ADI from each of the two joined objects. Access to the new object is mediated on the intersection of the parent object's ACLs. The user's subject accumulates ADI from all objects read; this accumulation persists until the user initiates a new subject.

ORAC considers the creator of an object to be its owner. Owners of objects whose content may have flowed into a new object are also considered to have ownership rights upon its content and are retained as prior owners. An object may thus potentially be marked with a series of ACLs, each associated with a different owner. Access mediation is based on the intersection of all ACLs associated with the object. The current owner has privileges for all modes of access; ACL checking is bypassed since the rights of the owner were established upon object creation. Owners may give up or transfer ownership by exercising an *ownership privilege*.

4.3 Originator-Controlled Access Control (ORGCON)

ORGCON [ABRA91] is a strong form of identity-based access control—it explicitly defines authority and delegation of authority, provides for accountability, and has an explicit inheritance policy.¹⁰ In ORGCON, the distribution list ADI is *indelibly attached* to the object (i.e., the distribution list cannot be disassociated from the object, even in the limited cases where copying is permitted). ORGCON is a *read, no-copy* policy. Its formal model [ABRA92] distinguishes among device types in order to deal with the policy that no storage copy of an object is permitted. Information may be “copied” only to the display and printer, but not to any other device types.

There are three roles associated with the ORGCON policy: *originator representative*, *recipient representative*, and *recipient*. When a user logs onto the system, he/she assumes the role of *ordinary user*. An explicit action must then be taken to assume one of the other roles, and some authority has already redefined which users are authorized to assume which roles. When a user assumes another role, all privileges associated with any previous role (e.g., ordinary user) are relinquished while in the current role.

¹⁰ An annotated demonstration booklet documenting the prototype is available from the author on request.

The fundamental concept in ORGCON is that information is distributed among organizations; individuals act in roles relative to these organizations. ORGCON information is owned by its originating organization. Several roles are defined to support the ORGCON policy. The *author* writes a document. The originating organization is represented by one or more individuals acting in the role *originator representative*, a form of message release authority. Any individual may generate information that may eventually be designated with the ORGCON marking, but only an originator representative can mark the information ORGCON and specify a distribution list of recipients. The originator representative role provides access to a process within the Trusted Computing Base (TCB) that performs the marking.

At each recipient organization, a *recipient representative* maintains the list of individuals in the organization authorized to receive ORGCON information. The recipient organization is a group that is composed of the authorized individual recipients of ORGCON information at that recipient organization.

A recipient must have a recipient role at the recipient organization.¹¹ Individuals acting in the role of *recipient representative* specify the individuals who are authorized for a recipient role and are, therefore, authorized to receive ORGCON information. Example recipient roles include the headquarters staff and Commander in Chief (CINC).

Note that ORGCON differs from ORCON by the introduction of the recipient representative, which is believed to be a necessary and practical step. The originator representative cannot be expected to be aware of personnel changes in the recipient organization, nor will he/she be likely to have the privileges to redefine the membership of the recipient group(s). The originator and recipient representatives are authority agents.

The *author* creates a document that is to become an ORGCON object. When the author is ready to have the document distributed, the document is then transferred to the originator representative as an ORGCON-C (candidate), along with a suggested distribution list (a list of recipient organizations). After the appropriate review, the originator representative marks the document ORGCON, binds the distribution list, and the document is transmitted to the recipient organizations. At the recipient organizations, the ORGCON document is distributed to the authorized individual recipients.

4.4 Typed Access Matrix (TAM) Model

TAM [SAND92a, SAND92b] incorporates strong typing into the access matrix model to provide a flexible model that can express a rich variety of security policies while addressing propagation of access rights and the safety problem. The safety problem is closely related to the fundamental flaw in Discretionary Access Control (DAC) that malicious code can modify the protection state. Types and rights are specified as part of the system definition; they are not predetermined in TAM.

Representing the information in the access matrix as an ACL associated with a single object implies that a single command can modify the ACL of exactly one object at the single site where the object exists. Coordinating completion of a single command at multiple sites is unnecessary, as is a two-phase commit. TAM has very strong expressive powers without compromise on safety analysis.

The TAM realization of ORCON is based on the ability in TAM to have multiple parents jointly create a child subject. Confined subject and object types are employed to limit rights. Information flow is inhibited by confined subjects being unable to write to any object or create any objects. The implementation architecture makes use of both ACLs and cryptographically protected certificates. All accesses to subjects and objects are mediated by local subject and object servers responsible for managing that entity. Authentication is also carried out at the time of subject/object access, and must be incorporated in the remote procedure call mechanism of the client-server architecture. The servers must also authenticate the source of every remote procedure call using some well-known cryptographic protocol.

5. Processes

TCSEC Mandatory Access Control (MAC) requires that ADI be associated with all storage objects. Processes are created by execution of a program. Program objects provided as part of the operating environment are usually labeled at or below the lowest user sensitivity so that MAC does not prevent their execution. DAC permissions may also be used to prevent reading the program objects while permitting execution, if platform support is available. Some

¹¹ Note that the ORCON policy identifies the headquarters element of an organization as the recipient. The ORGCON policy has been generalized and does not imply the headquarters element as the recipient.

programs may be classified; perhaps they contain classified algorithms or simply were created by users at sensitivity levels above the minimum. In this case, MAC will restrict access to subjects of appropriate sensitivity and will prevent write-down modification. Processes operate with the current sensitivity of the user on whose behalf they are operating, which may be below the user's maximum clearance level.

Biba integrity assigns ADI to program objects and the corresponding processes. Platforms that support both sensitivity and integrity ADI can implement two policies simultaneously. This is a commendable step in the direction of supporting multiple policies.

When second-order effects are considered, integrity is seen to support other policies, such as confidentiality. Malicious code, which violates the integrity of applications, can be expected to attempt violation of other policies. For example, substitution of malicious code for authorized utilities could be prevented. This would prevent installation of a Trojan horse for an authorized program. Modification to data on which actions of these programs depend may be far more significant than modifications to the code. Integrity controls must also extend over such data.

A well-known countermeasure is to associate a check value, such as a cryptographic remainder or digital signature, with the well-behaved program object and data, and to confirm the check value at the time of attempted execution of the program. The check value is not associated with the process. The check function and its associated cryptographic variables are under the control of the TCB. Integrity controls over the search path may also be necessary. Note that the integrity check value may be distinct from other integrity-related ADI. Lacking the ability to associate multiple integrity ADI with objects, one could use Biba integrity ADI to make program objects integrity-high so that their contents could not be altered.

In the absence of integrity controls, the **-property* was devised to prevent malicious code from downgrading an unauthorized copy of a file. Associating integrity ADI with processes is more general. ADI may be associated with all processes and objects under the control of the TCB. Rules implementing each access control policy may employ the current values of selected ADI in adjudicating attempted access. Inheritance rules govern the assignment of security attributes to entities.

One interesting case of assigning ADI to entities occurs when the entity is newly established within the IT system. Traditional confidentiality usage has been that the identification and authentication process starts a command interpreter process (or equivalent) executing as one result of success. That command interpreter is assigned sensitivity ADI selected by the human user from within his/her authorized set. Identification and authentication may be performed by the command interpreter [FRAI83].

The TCSEC inheritance rule is that each process acquires the ADI of the process that caused it to become active. More generally, program and data objects may have associated ADI; inheritance rules govern how these ADI relate to security attributes of processes created from the programs. System service processes, such as UNIX® *daemons*, do not act on behalf of individual users. Their ADI must, therefore, be set by some other policy. See [FRAI83] and [ABRA91] for examples.

The TP was introduced by Clark-Wilson [CLAR87] to automate the concept of the well-formed transaction. Several proposals for mechanisms to implement well-formed transactions have been suggested [LEE88, SHOC88]. Type enforcement deals with similar concerns. Association of ADI with processes is a general characteristic of all these mechanisms.

A not-so-hypothetical availability policy restricts execution of certain programs to certain hours. These programs might be resource intensive, or disruptive diagnostics, or they might be games. Like the check value described above, ADI is associated with the program; in this case, it is the time periods when execution of the program is permitted. This ADI is compared with the time of day to determine if execution is permitted.

A policy that incorporates date/time and sensitivity is a *press release policy*; this is also an *operation (battle) plan policy*. In this policy, information is considered highly sensitive before a certain time/date has been reached. During the period of high sensitivity, observe and modify access is highly restricted based on specified ADI. When the specified time/date is reached, the information is disseminated as widely as possible for observation only. Integrity of the clock is possibly of great importance. Precision is probably not important, and even a known bound on accuracy may be acceptable.

6. Summary

There is merit in viewing Access Control Policies in a modern perspective to extend the conceptual framework built for the TCSEC. There is a rich set of Access Control Policies, not a binary partitioning into mandatory and discretionary policies; traditional policy notions are retained as bounds. The following observations are emphasized:

- Access control policies that extend TCSEC concepts will be the norm in the future.
- Type enforcement and well-formed transactions can implement non-discretionary access controls not representable on a lattice or even a partial order.
- ADI may be associated with all processes and objects under the control of the TCB.
 - Rules implementing each access control policy may employ the current values of selected ADI in adjudicating attempted access.
 - Inheritance rules govern the assignment of ADI to processes and storage objects.
- A fundamental weakness of the TCSEC discretionary requirements is that they do not include explicit inheritance rules.
- Any global and persistent access control policy relying on ADI under control of the security administrator is *mandatory*.
- Any global and persistent access control policy relying on ADI not directly controlled by the security administrator is *non-discretionary*.
- Policies employing user identities can be discretionary, non-discretionary, or mandatory.
- Groups and roles can support discretionary, non-discretionary, and mandatory policies.
- TCB use of integrity checks to mediate program execution can control malicious modification.
- Associating ADI with processes is directly relevant to supporting integrity policy, and indirectly to supporting confidentiality and availability policies.

The conceptual basis developed for the TCSEC remains valid and useful, but must be enriched in order to maintain relevance to increasingly complex architectures, networks, applications, and distributed systems.

7. Acknowledgments

Comments and suggestions from a number of people have helped focus this paper and sharpen its message. In particular we acknowledge Jeffrey Edelheit, Lester Fraim, Richard Graubart, Steve Lipner, Carol Oakes, Ingrid Olson, James Williams, a friend who wishes to remain anonymous, and the anonymous reviewers.

8. References

- [ABRA90] Abrams, M. D., K. W. Eggers, L. J. LaPadula, and I. M. Olson, October 1990, "Generalized Framework for Access Control: An Informal Description," *Proceedings of the 13th National Computer Security Conference*.
- [ABRA91] Abrams, M. D., J. E. Heaney, O. King, L. J. LaPadula, M. Lazear, and I. M. Olson, October 1991, "Generalized Framework for Access Control: Towards Prototyping the ORGCON Policy," *Proceedings of the 14th National Computer Security Conference*.
- [ABRA92] Abrams, M. D., L. J. LaPadula, and I. M. Olson, 1992, *Generalized Framework for Access Control: A Formal Rule Set for the ORGCON Policy*, M92B0000037, The MITRE Corporation.
- [ANDE72] Anderson, J. P., 1972, *Computer Security Technology Planning Study*, ESD-TR-73-51, Vol. 1 and 2, Hanscom AFB, MA (also available as DTIC AD-758206).
- [BELL75] Bell, D. E., and L. J. LaPadula, 1975, *Secure Computer Systems: Generalized Framework for Exposition and Multics Interpretation*, ESD-TR-75-306, The MITRE Corporation. (Also available through NTIS, Springfield, VA, AD-A023588.)
- [BIBA77] Biba, K. J., 1977, *Integrity Considerations for Secure Computer Systems*, ESD-TR-76-372, Hanscom AFB, MA (also available as DTIC AD-A039324).
- [BOEB88] Boebert, W. E., 1988, "The LOCK Demonstration," *Proceedings of the 11th National Computer Security Conference*.

- [CLAR87] Clark, D. D., and D. R. Wilson, 1987, "A Comparison of Commercial and Military Computer Security Policies," *Proceedings of the Symposium on Security and Privacy*, IEEE Computer Society, pp. 184-194.
- [COHE75] Cohen, E., and D. Jefferson, November 1975, "Protection in the Hydra Operating System," *Proceedings of the 5th Symposium on Operating Systems*, pp. 141-160.
- [COMM91] Commission of the European Communities, 1991, *Information Technology Security Evaluation Criteria (ITSEC): Provisional Harmonized Criteria*, Version 1.2, Luxembourg, Office for Official Publications of the European Communities.
- [DCID81] Director of Central Intelligence, 4 May 1981, *Control of Dissemination of Intelligence Information*, Directive No. 1/7.
- [DENN76] Denning, D. E., 1976, "A Lattice Model of Secure Information Flow," *Communications of the ACM*, Vol. 19, No. 5, pp. 236-243.
- [DOWN85] Downs, D. D., J. R. Rub, K. C. Kung, and C. S. Jordan, 1985, "Issues in Discretionary Access Control," *Proceedings of the Symposium on Security and Privacy*, IEEE Computer Society, pp. 208-218.
- [FERR92] Ferraiolo, D., and R. Kuhn, 1992, "Role-Based Access Controls," *Proceedings of the 15th National Computer Security Conference*, pp. 554-563.
- [FLIN90] Flink, C. W., 1990, "The System V/MLS Project Model—An Approach to a Unified Access Control Interface," *Proceedings of the Sixth Annual Symposium on Physical and Electronic Security*, Philadelphia Chapter, Armed Forces Communications and Electronics Association, pp. A3-1-A3-5.
- [FRAI83] Fraim, L. J., July 1983, "SCOMP: A Solution to the Multilevel Security Problem," *IEEE Computer Magazine*, pp. 26-34.
- [GRAU89] Graubart, R., 1989, "On the Need for a Third Form of Access Control," *Proceedings of the 12th National Computer Security Conference*, pp. 296-303.
- [INTE91] National Computer Security Center, September 1991, *Integrity in Automated Information Systems*, C Technical Report 79-91.
- [ISO92] International Organization for Standardization, 1991, International Electrotechnical Committee, Joint Technical Committee 1, Subcommittee 21, *Working Draft on Access Control Framework*, Document Number 6188, draft.
- [JONE73] Jones, A. K., 1973, *Protection in Programmed Systems*, Ph.D. dissertation.
- [JONE75] Jones, A. K., and W. A. Wulf, October-December 1975, "Towards the Design of Secure Systems," *Software—Practice & Experience*, pp. 321-336.
- [KARG88] Karger, P., 1988, "Implementing Commercial Data Integrity with Secure Capabilities," *Proceedings of the Symposium on Security and Privacy*, IEEE Computer Society Press.
- [LAMP71] Lampson, B. W., January 1974, "Protection," *Proceedings of the Fifth Princeton Symposium on Information Sciences and Systems*, Princeton University, March 1971, pp. 437-443, reprinted in *Operating Systems Review*, Vol. 8, No. 1, pp. 18-24.
- [LAMP91] Lampson, B. W., M. Abadi, M. Burrows, and E. Wobber, October 1991, "Authentication in Distributed Systems: Theory and Practice," *Operating Systems Review*, Vol. 25, No. 5, pp. 165-182.
- [LEE88] Lee, T. M. P., April 1988, "Using Mandatory Integrity to Enforce 'Commercial' Security," *Proceedings of the Symposium on Security and Privacy*, IEEE Computer Society Press.
- [MCCO90] McCollum, C. J., J. R. Messing, and L. Notargiacomo, 1990, "Beyond the Pale of MAC and DAC: Defining New Forms of Access Control," *Proceedings of the Symposium on Research in Security and Privacy*, IEEE Computer Society Press.
- [MURR87] Murray, W. H., 1987, "Data Integrity in a Business Data Processing System," *Report of the Invitational Workshop on Integrity Policy in Computer Information Systems*, National Institute of Standards and Technology, Special Publication 500-160.
- [NCSC87] National Computer Security Center, 1987, *A Guide to Understanding Discretionary Access Control in Trusted Systems*, NCSC-TG-003, Version-1.

- [NCSC92] National Computer Security Center, 1992, *A Guide to Understanding Information System Security Officer Responsibilities for Automated Information Systems*, NCSC-TG-027, Version-1.
- [ORGA72] Organick, E., 1972, *The Multics System: An Examination of its Structure*, M.I.T. Press.
- [SALT75] Saltzer, J. H., and M. D. Schroeder, September 1975, "The Protection of Information in Computer Systems," *Proceedings of the IEEE*, Vol. 63, No. 9, pp. 1278-1308.
- [SAND88] Sandhu, R., December 1988, "Transaction Control Expressions for Separation of Duties," *Proceedings of the Fourth Aerospace Computer Security Applications Conference*.
- [SAND92a] Sandhu, R. S., 1992, "The Typed Access Matrix Model," *Proceedings of the Symposium on Research in Security and Privacy*, IEEE Computer Society, pp. 122-136.
- [SAND92b] Sandhu, R. S., and G. S. Suri, 1992, "Implementation Considerations for the Typed Access Matrix Model in a Distributed Environment," *Proceedings of the 15th National Computer Security Conference*, pp. 221- 235
- [SCHE86] Schell, R. R., and D. E. Denning, 1986, "Integrity in Trusted Database Systems." *Proceedings of the 9th National Computer Security Conference*.
- [SHIR81] Shirley, L. J., and R. R. Schell, 1981, "Mechanism Sufficiency Validation by Assignment," *Proceedings of the Symposium on Security and Privacy*, IEEE Computer Society, pp. 26-32.
- [SHOC88] Shockley, W. R., October 1988, "Implementing the Clark/Wilson Integrity Policy Using Current Technology," *Proceedings of the 11th National Computer Security Conference*, pp. 29-37.
- [TCSE85] U.S. Department of Defense, December 1985, *Department of Defense Trusted Computer System Evaluation Criteria*, DOD 5200.28-STD.

BSD IPC MODEL AND POLICY

*Sandra G. Romero
Casey Schaufler
Nelson Bolyard*

Silicon Graphics, Inc.
2011 North Shoreline Boulevard
Mountain View, California 94043

sgr@sgi.com
casey@sgi.com
nelson@sgi.com
Romero@dockmaster.ncsc.mil
Schaufler@dockmaster.ncsc.mil

ABSTRACT

The Berkeley Software Distributions (BSD) Inter-Process Communication (IPC) mechanism implements a transparent interface which has traditionally been used to provide access to Internet Protocol (IP) networks. This paper proposes a model and related security policy for use in designing and implementing the BSD IPC mechanism of sockets within a system intended for evaluation against the Department of Defense (DoD) Trusted Computer System Evaluation Criteria (TCSEC^[1]). The proposed model and policy have been incorporated into the analysis of the formal security policy model for a system currently in evaluation at level B1. The paper begins with a brief refresher on the basics of BSD IPC for those not readily familiar with that form of IPC. A concise description of the proposed model is included, preceded by a discussion of the motivation behind its development. Next, there is a description of the security policy to be enforced on the model showing its consistency with the Bell LaPadula formal security policy model^{[2][3][4][5]} and the Biba formal integrity policy model^[6]. Finally, contained in an appendix is a description of a UNIX implementation of the model and policy.

1. BACKGROUND

As described in "*The Design and Implementation of the 4.3BSD UNIX Operating System*"^[7], the BSD IPC services are designed to address three particular goals. The primary goal was to provide access for the Defense Advanced Research Projects Agency (DARPA) Internet to application programs. A secondary goal was to add capabilities to support multiprocess applications such as database servers. A third goal was to provide support for resource sharing in a distributed environment.

The BSD IPC services support notions of communications domains and sockets. The communications domain defines the semantics and capabilities of the underlying communications mechanism (e.g., Transmission Control Protocol layer of the Internet Protocol suite, i.e., TCP/IP). The socket provides an interface through which a program can interact with the communications domain in a transparent (communications domain independent) fashion.

A communications domain is not directly accessible to an application. The underlying system must know how to format, package, or otherwise digest the data being communicated on the program's behalf. In the Internet Domain, for example, the data is included in a packet which includes source and destination addresses as well as options which can identify attributes of the information. It would not be appropriate for each application to be trusted to produce legitimate Internet Domain packets, due to data integrity, portability, and security concerns.

A socket is created on request by the application. Upon creation, it is associated with a communications domain, a type such as stream or datagram, and, optionally, a protocol. The socket is explicitly bound, or associated, with a name space entry by a separate operation. The program may then either read or write using the socket. The type of read or write operation will depend on the type of socket, as defined at the

time of creation.

For a more comprehensive discussion of BSD IPC communication mechanisms please refer to Chapters 10, 11, and 12 of *"The Design and Implementation of the 4.3BSD UNIX Operating System"*^[7].

2. MOTIVATION

The process of trusted system evaluation begins with the characterization of the system to be evaluated. Ideally, this would be done in the early design phases of the system, when policy still outweighs history and interfaces are mutable. This was not the situation in which we found ourselves. The system we were taking into NCSC B1 evaluation was a conglomeration of interfaces and implementations based on distributions from several vendors and significant in-house development. Most interfaces were frozen by history and standardization. Perhaps most perplexing was the fact that no attempt had been made to state, much less model, the system's security policies.

We chose to approach the required modeling effort by associating each of the system call interfaces with a set of objects upon which they acted. It was our assertion that an entity should be considered an object if and only if it could be manipulated by one or more system calls. Our assumption served us well through the file system objects, the process objects, and the System V IPC objects, producing policies which proved both simple and obvious.

Our confidence wavered slightly when we encountered the BSD IPC system calls. All of the system calls associated actions with sockets, yet in some cases the data never actually got attached to the local socket; rather it was enqueued on a different socket, that of the intended recipient. Clearly the object in the "write" case was not the socket associated with the sending program. Any access control decisions would have to be made on the receiving end of the connection, as that's where the attributes of the receiver reside. To further confuse the issue, User Datagram Protocol (UDP) datagrams are sent without any guarantee of delivery, so the sender is explicitly not allowed to make access control checks, but is still allowed to write.

It finally dawned on us that any model which would describe the BSD IPC mechanism would have to deal with at least two distinct objects. One object was obvious, that being the receiving socket. Our initial assumption was that the sending socket was the second object, giving us a symmetrical model. However, there was one significant flaw; this model couldn't be used to describe how a message got from the sender's socket to the receiver's socket. In fact, data was never added to the sender's socket. Thus, this model did not reflect reality.

So what was the missing object? By following the code path in the implementation of the system call which was used to send data via sockets, we found that the data was not stored in any form by the sender. Rather, it was passed directly to the protocol (e.g., the TCP/IP protocol or the UNIX Domain "protocol"), for transport to its intended destination. Here then was the missing object. This transport became the second object in our model.

3. MODEL DESCRIPTION

The basic BSD IPC model consists of a single subject type and two object types. In all cases the subject is a process. Depending upon whether a subject wishes to send a message or receive a message the object is either a transport object or a socket object, respectively. Thus, a process sends a message (appends) to a transport object and receives a message (reads) from a socket object. In the course of passing data from a sending subject to a receiving subject via BSD IPC, there are three access control mediation operations which occur: the first on the sending side; the remaining two on the receiving side.

In attempting to send a message (append) to a transport object, the sending process retrieves attributes from the socket associated with that process and bundles those attributes with the data to be placed on the transport. It is at this point that the first of the access control mediation operations is performed. That first check guarantees that the attributes of the sending process (e.g., label) are appropriate for appending to the transport object (e.g., within the label range associated with that transport object).

In addition to the subjects and objects described above, there exists (in kernel space) a trusted process or BSD IPC "trusted agent"; essentially an intermediary between the transport object and the receiving socket

object. It is this agent which actually performs the two access control mediation operations at the receiving end of a BSD IPC connection. The first check occurs when the agent reads data from a transport object, and guarantees that the attributes associated with that data are consistent with the attributes associated with that transport object. This is merely a verification of the access control mediation that was performed when the sending process appended to the transport object. The second occurs when the agent checks to see whether delivery to a receiving socket is appropriate, and guarantees that data is not delivered to a socket which does not carry the appropriate attributes. If either of these checks fails, the data is considered undeliverable and is discarded.

Figure 3-1 below depicts both the basic BSD IPC model and the added complexity introduced by the BSD IPC "trusted agent":

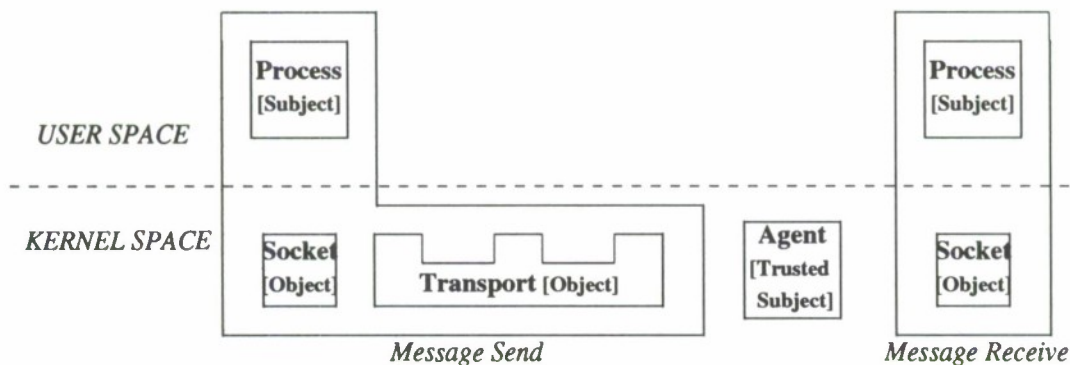


Figure 3-1. BSD IPC Model

Both BSD IPC object types conform to a basic named object definition. That definition states that each named object is comprised of two distinct parts, each of which is kept in one or more distinct storage objects. The first part represents the object attribute information, which contains access control information. The second part represents the object data. This structure is depicted in Figure 3-2 below:

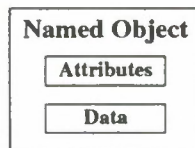


Figure 3-2. BSD IPC Object Definition

Each of the BSD IPC basic object types can be further divided into two distinct flavors. In the case of the socket object type, there are the datagram and the stream socket object types. Although these two object types differ in terms of the means by which they are accessed and manipulated, they are identical in terms of their representation as depicted by Figure 3-3 below. In the case of the transport object type, there are the Internet Domain and the UNIX Domain transport object types. These two object types differ both in terms of the means by which they are accessed and manipulated, and in terms of their representation, as depicted by Figures 3-4 and 3-5 below.

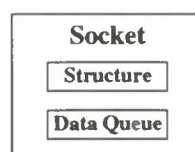


Figure 3-3. Datagram/Stream Socket Object Type Definition

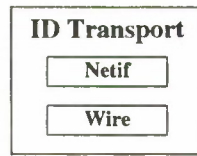


Figure 3-4. Internet Domain Transport Object Type Definition

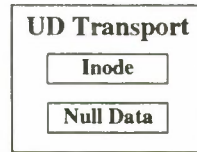


Figure 3-5. UNIX Domain Transport Object Type Definition

4. POLICY DESCRIPTION

As can be seen from Figure 4-1 below, depicting all potential communication access paths, the BSD IPC model allows only certain communication paths between a "Process" or an "Agent" and any of the defined socket or transport object types: a "Process" can read both the attributes and data of either socket object type, but can read only the attributes of either transport object type; a "Process" can write only the attributes of either socket object type, but can write both the attributes and data of either transport object type; an "Agent" can read only the attributes of either socket object type, but can read both the attributes and data of either transport object type; and finally, an "Agent" can write only the data of either socket object type, and an "Agent" can write neither the attributes nor data of either transport object type.

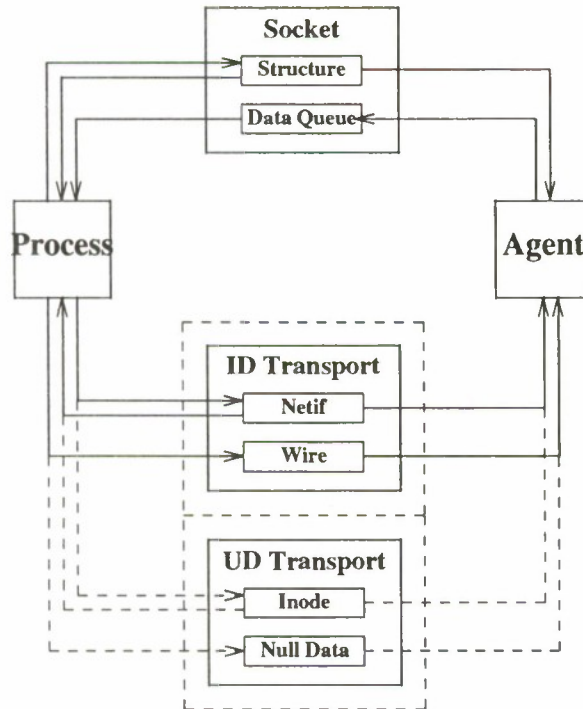


Figure 4-1. BSD IPC Access Paths

Since each object type is defined to be comprised of two distinct parts, attributes and data, and since the accesses allowed may differ for either of those two distinct parts (as described above), it follows that the

security policy can be more readily and clearly defined if partitioned into access control policies representing the defined object structure. Thus, the security policy has been partitioned into four basic access control policies: Attribute Read, Attribute Write, Data Read, and Data Write. For the socket object type, these access control policies do not differ between datagram and stream socket objects. However, for the transport object type, the access control policy does differ between the Internet Domain and the UNIX Domain transport objects.

The complete security policy for the BSD IPC Model is defined in Sections 4.1 through 4.4, below. For purposes of simplicity in describing the entire sensitivity and integrity security policy for the model, the term "label" will be meant to imply a composite of both a standard sensitivity label and an integrity label. This composite "label" in conjunction with the fact that the Biba integrity model^[6] is the dual of the Bell LaPadula sensitivity model^{[2][3][4][5]} dictates an adjustment in the application of the dominance principal. Thus, in the policy statements that follow, the phrase "the label of ... dominates the label of ..." first will be referring to a composite "label" as described above, and will be interpreted to mean that the sensitivity portion of the first composite "label" must dominate the sensitivity portion of the second composite "label" in the traditional sense of dominance, while the integrity portion of the first composite "label" must be dominated by the integrity portion of the second composite "label", thus incorporating into the statement of the security policy the duality of the Bell LaPadula sensitivity model^{[2][3][4][5]} and the Biba integrity model^[6].

4.1 Attribute Read Access Policy

- Socket: The attribute read access policy with respect to the BSD IPC socket object type is to allow access if and only if the requesting process is the socket object creator or one of its descendants, or if the requesting process is the BSD IPC "trusted agent".
- Transport: Internet Domain: The attribute read access policy with respect to the BSD IPC Internet Domain transport object type is to allow access.
- UNIX Domain: The attribute read access policy with respect to the BSD IPC UNIX Domain transport object type is to allow access if and only if the label of the requesting process dominates the label of all of the directories in the path, the file permission bits allow a user with the process' user ID execute access to each of the directories in the path, and the label of the requesting process dominates the label associated with the UNIX Domain transport object, or if the requesting process is the BSD IPC "trusted agent".

4.2 Attribute Write Access Policy

- Socket: The attribute write access policy with respect to the BSD IPC socket object type is to allow access if and only if the requesting process is the socket object creator or one of its descendants. The exceptions to this are the socket user ID and label attributes. Only the Superuser process can change the user ID of a socket it has created, and only the Superuser process can change the label attribute associated with a socket object.
- Transport: Internet Domain: The attribute write access policy with respect to the BSD IPC Internet Domain transport object type is to allow access if and only if the requesting process is owned by the Superuser. The exception to this is the Internet Domain transport label range attribute. The label range attribute associated with an Internet Domain transport object is set at system generation, i.e., when the object is implicitly created. The label range attribute associated with an Internet Domain transport object remains immutable until system shutdown, i.e. until such time as the object is implicitly deleted.
- UNIX Domain: The attribute write access policy with respect to the BSD IPC UNIX Domain transport object type is to allow access if and only if the label of the requesting process dominates the label of all of the directories in the path, the file permission bits allow a user with the process' user ID execute access to each of the directories in the path, and the label of

the requesting process equals the label associated with the UNIX Domain transport object. The exception to this is the UNIX Domain transport label attribute. Only the Superuser process can change the label attribute associated with a UNIX Domain transport object.

4.3 Data Read Access Policy

Socket: The data read access policy with respect to the BSD IPC socket object type is to allow access if and only if the requesting process is the socket object creator or one of its descendants.

Transport: Only the BSD IPC "trusted agent" process can read data from a transport object.

4.4 Data Write Access Policy

Socket: Only the BSD IPC "trusted agent" process can write data to a socket object.

Transport: Internet Domain: The data write access policy with respect to the BSD IPC Internet Domain transport object type is to allow access if and only if the current process label is contained within the label range associated with the Internet Domain transport object.

UNIX Domain: The data write access policy with respect to the BSD IPC UNIX Domain transport object type is to allow access if and only if the label of the requesting process dominates the label of all of the directories in the path, the file permission bits allow a user with the process' user ID execute access to each of the directories in the path, the label of the requesting process is equal to the label associated with the UNIX Domain transport object, and the file permission bits associated with the UNIX Domain transport object allow a user with the process' user ID write access to the UNIX Domain transport object.

5. CONCLUSIONS

The prospect of modeling the BSD IPC interfaces has proven sufficiently daunting to the extent that several vendors have omitted them from their secure systems. Other vendors have proposed alternative interpretations of the network interface in order to avoid defining subject and object relationships. We found that, although it required significant contemplation, the specter was much worse than the battle and the model simpler than any of us had envisioned.

Our experience with this model has been completely positive. Not only does it allow an implementation that is secure, but it accommodates all of the interfaces handed down to us. It works in total harmony with the Bell LaPadula sensitivity^{[2][3][4][5]} and the Biba integrity^[6] models, producing a composite which is wholly consistent with each. When we implemented discretionary access control on the interfaces it was obvious from the model how to do so.

A good security model should make it easy to determine the characteristics of the system it describes. Our model reflects the interfaces we inherited, and identifies the areas in which access control can and should be done. It allows an implementation of access control which can be accommodated by both the model and the system. This model is successful in that it makes the process of enforcing the system policies easy to design and describe in the contexts of security and capability.

A. UNIX IMPLEMENTATION

This appendix has been included in order to describe an implementation which conforms to the above described model. The model described in this paper is being used as part of the evaluation evidence for the Trusted IRIX/B operating system under development at Silicon Graphics. Trusted IRIX/B uses the popular socket interfaces from 4.3BSD. System calls provide the only mechanism by which a subject may access an object. The file system interfaces described are included solely because of the file system object representation of the UDS transport objects.

A.1 UNIX Modules

For purposes of discussion, each system call will be considered a module. The following system calls are identified as implementing BSD IPC, and enforce the stated access control policy:

• accept	Attribute Write Access Policy for BSD IPC Socket object types.
• bind	Attribute Write Access Policy for BSD IPC Socket object types. Attribute Write Access Policy for BSD IPC UNIX Domain Transport object types.
• chmod	Attribute Write Access Policy for BSD IPC UNIX Domain Transport object types.
• chown	Attribute Write Access Policy for BSD IPC UNIX Domain Transport object types.
• close	No access control decisions are made.
• connect	Attribute Write Access Policy for BSD IPC Socket object types.
• exit	No access control decisions are made. [Other processes may be <i>killed</i> as a side effect of this call. The security semantics of the <i>kill</i> call apply.]
• fcntl	Attribute Read Access Policy for BSD IPC Socket object types. Attribute Write Access Policy for BSD IPC UNIX Domain Transport object types.
• getlabel	Attribute Read Access Policy for BSD IPC UNIX Domain Transport object types.
• getpeername	Attribute Read Access Policy for BSD IPC Socket object types.
• getsockname	Attribute Read Access Policy for BSD IPC Socket object types.
• getsockopt	Attribute Read Access Policy for BSD IPC Socket object types.
• ioctl	Attribute Read Access Policy for BSD IPC Socket object types. Attribute Read Access Policy for BSD IPC Internet Domain Transport object types. Attribute Write Access Policy for BSD IPC Internet Domain Transport object types. Attribute Write Access Policy for BSD IPC UNIX Domain Transport object types.
• listen	Attribute Write Access Policy for BSD IPC Socket object types.
• lstat	Attribute Read Access Policy for BSD IPC UNIX Domain Transport object types.
• read	Data Read Access Policy for BSD IPC Socket object types.
• recv	Data Read Access Policy for BSD IPC Socket object types.
• recvfrom	Data Read Access Policy for BSD IPC Socket object types.
• recvmsg	Data Read Access Policy for BSD IPC Socket object types.
• recvmsg	Data Read Access Policy for BSD IPC Socket object types.
• select	Attribute Read Access Policy for BSD IPC Socket object types.
• send	Data Write Access Policy for BSD IPC Internet Domain Transport object types. Data Write Access Policy for BSD IPC UNIX Domain Transport object types.
• sendmsg	Data Write Access Policy for BSD IPC Internet Domain Transport object types. Data Write Access Policy for BSD IPC UNIX Domain Transport object types.
• sendto	Data Write Access Policy for BSD IPC Internet Domain Transport object types. Data Write Access Policy for BSD IPC UNIX Domain Transport object types.
• setlabel	Attribute Write Access Policy for BSD IPC UNIX Domain Transport object types.
• setsockopt	Attribute Write Access Policy for BSD IPC Socket object types.
• shutdown	Attribute Write Access Policy for BSD IPC Socket object types.
• socket	Attribute Write Access Policy for BSD IPC Socket object types.
• socketpair	Attribute Write Access Policy for BSD IPC Socket object types.
• stat	Attribute Read Access Policy for BSD IPC UNIX Domain Transport object types.
• unlink	Attribute Write Access Policy for BSD IPC UNIX Domain Transport object types.
• write	Data Write Access Policy for BSD IPC Internet Domain Transport object types. Data Write Access Policy for BSD IPC UNIX Domain Transport object types.

A.2 Datagram Socket Object Type

Attributes:	The attributes associated with a <i>datagram socket object</i> are stored in its structure . These attributes include information about the socket type, supporting protocol, socket state, and the transport type.
Data:	The data associated with a <i>datagram socket object</i> is stored in its data queue and is the data queued for receipt.
Names:	A <i>datagram socket object</i> can be named in the following way:

- By a file descriptor

Manipulation: A *datagram socket object* can be created with the following system calls:

- *socket(2)*, *socketpair(2)*

A *datagram socket object* can be deleted with the following system calls:

- *close(2)*, *exit(2)*

The attributes associated with a *datagram socket object* can be read with the following system calls:

- *fcntl(2)*, *getpeername(2)*, *getsockname(2)*, *getsockopt(2)*, *ioctl(2)*, *select(2)*

The attributes associated with a *datagram socket object* can be written with the following system calls:

- *bind(2)*, *setsockopt(2)*, *shutdown(2)*, *socket(2)*, *socketpair(2)*

The data associated with a *datagram socket object* can be read with the following system calls:

- *read(2)*, *recv(2)*, *recvfrom(2)*, *recvmsg(2)*, *recvmsg(2)*

The data associated with a *datagram socket object* can be written only by the BSD IPC "trusted agent".

A.3 Stream Socket Object Type

Attributes: The attributes associated with a *stream socket object* are stored in its **structure**. These attributes include information about the socket type, supporting protocol, socket state, and the transport type.

Data: The data associated with a *stream socket object* is stored in its **data queue** and is the data queued for receipt.

Names: A *stream socket object* can be named in the following way:

- By a file descriptor

Manipulation: A *stream socket object* can be created with the following system calls:

- *accept(2)*, *socket(2)*, *socketpair(2)*

A *stream socket object* can be deleted with the following system calls:

- *close(2)*, *exit(2)*

The attributes associated with a *stream socket object* can be read with the following system calls:

- *fcntl(2)*, *getpeername(2)*, *getsockname(2)*, *getsockopt(2)*, *ioctl(2)*, *select(2)*

The attributes associated with a *stream socket object* can be written with the following system calls:

- *accept(2)*, *bind(2)*, *connect(2)*, *listen(2)*, *setsockopt(2)*, *shutdown(2)*, *socket(2)*, *socketpair(2)*

The data associated with a *stream socket object* can be read with the following system calls:

- *read(2)*, *recv(2)*, *recvfrom(2)*, *recvmsg(2)*, *recvmsg(2)*

The data associated with a *stream socket object* can be written only by the BSD IPC "trusted agent".

A.4 Internet Domain Transport Object Type

Attributes: The attributes associated with an *Internet Domain transport object* are stored in its **Netif**. These attributes include information about the transport name, IP address, and the MAC label range.

Data: The data associated with an *Internet Domain transport object* is stored on the wire and is the data in transit destined for a receiving socket object.

Names: An *Internet Domain transport object* can be named in the following way:

- By an address

Manipulation: An *Internet Domain transport object* can not be created with any system call. An *Internet Domain transport object* is created implicitly at system generation. An *Internet Domain transport object* can not be deleted with any system call. An *Internet Domain transport object* is deleted implicitly at system shutdown. The attributes associated with an *Internet Domain transport object* can be read with

the following system call:

- *ioctl(2)*

The attributes associated with an *Internet Domain transport object* can be written with the following system call:

- *ioctl(2)*

The data associated with an *Internet Domain transport object* can be read only by the BSD IPC "trusted agent".

The data associated with an *Internet Domain transport object* can be written with the following system calls:

- *send(2)*, *sendmsg(2)*, *sendto(2)*, *write(2)*

A.5 UNIX Domain Transport Object Type

- Attributes:** The attributes associated with a *UNIX Domain transport object* are stored in its **Inode**. These attributes include information about the file system rendezvous point, i.e., its size, MAC label, owner, group, and DAC permission bits.
- Data:** The data associated with a *UNIX Domain transport object* is the data in transit destined for a receiving socket object.
- Names:** A *UNIX Domain transport object* can be named in the following way:
- By an address
- Manipulation:** A *UNIX Domain transport object* can be created with the following system call:
- *bind(2)*
- A *UNIX Domain transport object* can be deleted with the following system call:
- *unlink(2)*
- The attributes associated with a *UNIX Domain transport object* can be read with the following system calls:
- *getlabel(2)*, *lstat(2)*, *stat(2)*
- The attributes associated with a *UNIX Domain transport object* can be written with the following system calls:
- *chmod(2)*, *chown(2)*, *fcntl(2)*, *ioctl(2)*, *setlabel(2)*
- The data associated with a *UNIX Domain transport object* can be read only by the BSD IPC "trusted agent".
- The data associated with a *UNIX Domain transport object* can be written with the following system calls:
- *send(2)*, *sendmsg(2)*, *sendto(2)*, *write(2)*

REFERENCES

1. National Computer Security Center, "Department of Defense *Trusted Computer System Evaluation Criteria*", DoD 5200.28-STD U.S. Department of Defense, Washington, D. C., December 1985
2. D. Elliott Bell and Leonard J. La Padula, "Secure Computer Systems: Mathematical Foundations", ESD-TR-73-278, Volume I, AD 770 768, Electronic Systems Division, Air Force Systems Command, Hanscom AFB Bedford, Massachusetts, November 1973
3. Leonard J. La Padula and D. Elliott Bell, "Secure Computer Systems: A Mathematical Model", ESD-TR-73-278, Volume II, AD 771 543, Electronic Systems Division, Air Force Systems Command, Hanscom AFB Bedford, Massachusetts, November 1973
4. D. Elliott Bell, "Secure Computer Systems: A Refinement of the Mathematical Model", ESD-TR-73-278, Volume III, AD 780 528, Electronic Systems Division, Air Force Systems Command, Hanscom AFB, Bedford, Massachusetts, April 1974
5. D. Elliott Bell and L. J. La Padula, "Secure Computer Systems: Unified Exposition and Multics Interpretation", ESD-TR-75-306, Electronic Systems Division, Air Force Systems Command, Hanscom AFB, Bedford, Massachusetts, March 1976
6. K. J. Biba, "Integrity Considerations for Secure Computer Systems", Mitre TR-3153, The Mitre Corporation, Bedford, Massachusetts, April 1977
7. Samuel J. Leffler, Marshall Kirk McKusick, Michael J. Karels, and John S. Quarterman, "The Design and Implementation of the 4.3 BSD UNIX Operating System", Addison Wesley, 1989

AN EXAMINATION OF FEDERAL AND COMMERCIAL ACCESS CONTROL POLICY NEEDS

*Mr. David F. Ferraiolo
Mr. Dennis M. Gilbert
Ms. Nickilyn Lynch*

National Institute for Standards and Technology
Computer Systems Laboratory
Gaithersburg, MD 20899

ABSTRACT

In a cooperative effort with government and industry, the National Institute of Standards and Technology (NIST) conducted a study to assess the current and future information technology (IT) security needs of the commercial, civil, and military sectors. The study was documented in NISTIR 4976, Assessing Federal and Commercial Information Security Needs (1). The conclusions of the study address basic security needs of IT product users, who include system developers, end users, administrators, and evaluators. Security needs were identified based on existing security organizational practices. This paper reviews the access control findings of the NIST study and explores how an expanded set of access control objectives might be applied in a variety of application environments.

Keywords: *access control objectives, access control policy, policy objectives, trusted systems*

INTRODUCTION

Information technology (IT) systems are integral to the functioning of the federal government and private industry in meeting their individual operational, financial, and information technology requirements. The inability to protect the confidentiality, integrity, and availability of the IT systems and the sensitive information they contain could have serious impact on an organization. The impact could be financial or legal and affect human safety, personal privacy, and public confidence. In the extreme, the ability of the organization to perform some or all of its mission could be impacted.

In order to assess the current and future security needs of the commercial, civil, and military sectors, the National Institute of Standards and Technology (NIST), in

a cooperative effort with government and industry, conducted a study. The primary objectives of the study were to determine a basic set of information protection policies and control objectives for addressing the secure processing needs within all sectors and to identify protection requirements and technical approaches that are used, desired, or sought. This information is to be considered for future federal standards and guidelines.

To ensure a variety of perspectives, the NIST study team met with 28 organizations: 17 federal agencies, 10 commercial organizations, and 1 state government. Companies representing energy, financial, communications, insurance, manufacturing, computers, and service were included. Activities included law enforcement, benefits delivery, medical/hospital, nuclear/energy, space exploration, defense, tax system, information collection and dissemination, air traffic, and service center operations. Contractors participated in a number of the federal agency meetings.

BACKGROUND

The U.S. government has been involved in developing security technology for computer and communications security for some time. Although there has been much progress, many think that the current set of security technology does not fully address the needs of all, especially those organizations outside the Department of Defense (DoD).

The current set of security criteria, criteria interpretation, and guidelines has grown out of research and development efforts of the DoD over the past two decades. The primary U.S. computer security standard, the Trusted Computer System Evaluation Criteria (2) (TCSEC), consists of security features and assurances, derived from DoD security policy. The TCSEC focuses especially on those policies created to prevent the unauthorized disclosure of classified information. The result is a collection of security products built to TCSEC requirements that do not fully address unclassified sensitive security issues. The NIST study indicated that the TCSEC requirements can be useful in providing computer security in non-DoD sectors. However, in many instances, they provide only a partial solution and are used in place of a more appropriate set of controls.

Several efforts are in progress to develop a new set of criteria (3)(4) that incorporates the positive aspects of the TCSEC, provides a set of minimum protections for a common set of expectations by users and vendors, and is consistent with related international harmonization efforts. The most far reaching of them is the Federal Criteria for Information Technology Security (FC).

DRIVING NEED FOR ACCESS CONTROL

Both federal government and corporations were found to rely heavily on information processing systems to meet their individual operational, financial, and information technology requirements. The corruption, unauthorized disclosure, or theft of resources disrupts operations and can have financial, legal, human safety, personal privacy, and public confidence impacts. Methods must be found to prevent computers from being used in such acts as fraud, harassment, or terrorism.

Organizations processing and storing classified information focus on preventing unauthorized observation/disclosure of data as the basis for protection. For these organizations, the unauthorized flow of information from a high level to a low level of sensitivity was the principal concern.

For the federal government, requirements exist for protecting the privacy of personal information. These requirements come from the Privacy Act of 1974. The Act provides privacy safeguards by requiring each federal agency to protect the personal information it collects, maintains, uses, and disseminates. Additionally, when an agency contracts for services, the contractor must protect the information subject to the Act's requirements.

Although not always mandated by law, protecting the privacy of personal information is also relevant to commercial sector organizations. The need to protect sensitive data from unauthorized access results from operational environment (including threats) and data sensitivity factors. These factors include legal obligations and self-imposed requirements, including confidentiality of salary, performance, and health (mental, drug, and alcohol related illness), as well as data involving litigation.

Privacy issues were perceived as particularly critical in medical and insurance applications. Educational, employment and personnel, banking and financial institutions, and credit bureaus also acknowledged protecting the privacy of individuals as a high organizational priority.

The need to preserve customer, insurer, and stockholder confidence was cited as principle motivators for organizations in promoting access control requirements for many organizations. The vice president of a major bank described the need to "provide a good service at a reasonable cost" as an important capability of most savings and financial institutions, but described the need for "maintaining a general sense of customer confidence" as critical.

The basis for protection takes on a specific meaning for those organizations, such as banks, credit companies, and insurance companies, concerned with preventing unauthorized distribution of financial assets. These businesses are subject to federal regulatory requirements of the Federal Trade Commission and

the Federal Reserve Board under the Fair Credit Billing Act, Equal Credit Opportunity Act, and Truth-in-Lending Act.

Security issues can also be unique to a specific industry. Of all the organizations interviewed, only PACBELL and BELLCORE were concerned with preventing unauthorized use of long-distance telephone circuits. Only hospitals and those who develop hospital systems were concerned with preventing unauthorized distribution of prescription drugs.

Professional standards, health and safety, prevention of embezzlement, good business practices, avoidance of conflict of interest, and profit were also stated as key factors in an organization's basis for protection.

ACCESS CONTROL APPROACHES

The access needs and control policies of each organization interviewed varied. Many of these policies consider application, site, organizational, industry, or agency-unique factors.

Access control policies are context-dependent; it is not possible to know the environment in which such control will be applied. Not all stated access control policies can be easily mapped and implemented using the existing access control framework of the TCSEC. The TCSEC specifies two types of controls: Discretionary Access Controls (DAC) and Mandatory Access Controls (MAC). Since the TCSEC's appearance in December of 1983, DAC requirements have been perceived as being technically correct for commercial and civil security needs, as well as for single-level military systems. MAC is used for multi-level secure military systems, but its use in other applications is rare. The need for access controls more appropriate to the commercial and civil sector than that of DAC was found to exist. There is a need for DAC, but DAC falls short when implemented alone to solve the wide breath of security problems facing sensitive processing environments.

The remainder of this section describes the applicability of DAC and MAC, as well as other access control approaches, to the policy needs of those organization interviewed.

Discretionary Access Control

As defined in the TCSEC and commonly implemented, DAC is an access control mechanism that permits system users to allow or disallow other users access to objects under their control:

A means of restricting access to objects based on the identity of subjects and/or groups to which they belong. The controls are

discretionary in the sense that a subject with a certain access permission is capable of passing that permission (perhaps indirectly) on to any other subject (unless restricted by mandatory access control). (2)

DAC, as the name implies, permits the granting and revoking of access privileges to be left to the discretion of the individual users. A DAC mechanism allows users to grant or revoke access to any of the objects under their control without the intercession of a system administrator.

DAC plays an important role in supporting security requirements of many organizations, especially within engineering and research environments where the discretionary sharing of access and exchange of information is important. For many organizations, the end-users must be able to specify what access other users have to resources that they control. Within these environments, the need for users to access information is dynamic and changes rapidly over short periods.

Although appropriate in specific environments, many organizations expressed concerns about relying solely on DAC as the primary means of protection. Specifically, they were concerned with the propagation of access rights, reliance on the cooperation of users, and, to a lesser extent, DAC's vulnerabilities to a Trojan Horse.

Some organizations expressed concern over exactly who has the capability to specify group membership. By granting membership to a group, user access rights to protected data can change dynamically without the knowledge of the owner of that data. For some organizations, the ability to specify group membership was described as appropriately placed at the project level, while for other organizations, group membership was more appropriately placed at the security officer level. In addition, the ability to list group membership before granting access privileges to that group was considered by some as a necessary part of this capability.

The most common approach to implementing DAC is through access control lists (ACLs). The TCSEC encourages ACLs as appropriate for user-controlled access rights. However, when centrally administered, ACLs can become clumsy and difficult to maintain. In centrally administering DAC, the system administrator assumes responsibility for ownership of all resources, determining what resources and modes of access are needed for the performance of each user's function within the organization. For each new user or every change in responsibility, the administrator establishes the appropriate access rights within the system. Additionally, when a person leaves the organization, the administrator deletes the person from all ACLs within the system.

Separation of Transactions

A transaction can be thought of as a transformation procedure (5) (a program or a portion of a program) plus a set of associated data items. The term transaction is used in this paper to refer to a binding of transformation procedure and data storage access. (6) This is not unlike conventional usage of the term in commercial systems. For example, a savings deposit transaction is a procedure that upgrades a savings database and transaction file. A transaction may be quite general, e.g., "read savings file." Note however, that "read" is not a transaction in the sense used here, because the read is not bound to a particular data item, as "read savings file" is.

The importance of control over transactions, as opposed to simple read and write access, can be seen by considering a simple banking transaction. Tellers may execute a savings deposit transaction, requiring read and write access to specific fields within a savings file and a transaction log file. An accounting supervisor may be able to execute correction transactions, requiring exactly the same read and write access to the same files as the teller. The difference is the process executed and the values written to the transaction log file.

Separation of transactions is a design and implementation approach to partition task-oriented sets of programs and data. This set can be made available to a specific user who is allowed access only to these resources. A group of available transactions define a particular task that can be assumed by a user. The underlying access controls are achieved through a combination of administrative and transaction-design decisions.

Because of the stable functionality and the deterministic characteristics of transactions within some organizations, security engineers, or those knowledgeable of security issues facing an organization (i.e., privacy, data integrity, etc.), often play an important role in specifying access-control decisions during the design and development of a transaction. For example, for one organization, transactions were designed to retrieve an entire customer record minus the customer's social security number. In addition, design-time access control decisions can consider aggregation problems that are difficult to address within conventional run-time access control environments. Security guidelines are addressed by the designer and developers of a transaction or by direct involvement in the design and development effort of proposed transactions.

Once a transaction has been developed and introduced into the operational environment, a security administrator may assign the named transaction to specific users or user groups.

A major insurance company enforces its corporate policy through the use of separation of transactions. Within the organization, each unit (performing a specified task) is assigned a collection of transactions to perform an assigned task

or function. A unit security administrator determines "what users get access to what transactions," within that unit. The security administrator can add and delete users to the unit, but cannot assign transactions to individuals outside the unit.

Role-Based Controls

Many organizations preferred a centrally administered, non-discretionary set of controls to meet their security policies and objectives. During the course of this study, organizational policies and objectives included maintaining and enforcing the rules and ethics associated with a judge's chambers, and the laws and respect for privacy of diagnosing ailments, treating of disease, and administering of medicine within a hospital. To support such policies, a capability to centrally control and maintain access rights is desirable. The security administrator is responsible for enforcing policy and represents the organization as the "owner" of system objects. Access control decisions were found to be based on the roles individual users take on as part of an organization. This includes the specification of duties, responsibilities, obligations, and qualifications. For example, the roles included doctor, nurse, clinician, or pharmacist associated with a VA hospital. The doctor's role includes privileges to perform diagnoses, prescribe medication, or add an entry to (not simply modify) a record of treatments performed on a patient. The privileges defined for the role of pharmacist include those to dispense (not prescribe) prescription drugs.

The determination of membership and the allocation of privileges to a role is not so much in accordance with discretionary decisions on the part of a system administrator, but rather in compliance with organization-specific protection guidelines. These guidelines derive from existing laws, ethics, regulations, or generally accepted practices. The guidelines are non-discretionary in the sense that they are unavoidably imposed on users. For example, a doctor can prescribe medication, but cannot pass that privilege on to a nurse.

In addition, roles can be composed of other roles. For example, the role Doctor within a hospital system can be composed of the roles Doctor and Intern. Granting membership to the role Doctor implies access to all transactions defined by Intern. However, membership to the role Intern does not imply Doctor privileges.

Once roles are established within the system, the privileges associated with these roles remain relatively constant or change slowly over time. The administrative task is then to grant and revoke user membership to the set of specified roles within the system. The capability of an administrator to simply grant or delete membership to existing roles has been described as desirable. When a user's function changes within the organization, a mechanism needs to be available to allow easy deletion of existing roles and granting of new ones. Finally, when a person leaves the organization, all of that person's memberships to all roles are deleted. For an organization that experiences a large turnover of personnel, a

role-based implementation security policy is a good logical choice.

Access Based on Separation of Related Duties

Although more of a policy than a mechanism, separation of related duties is used in deterring fraud within financial systems. Such duties can include authorizing, approving, and recording transactions, issuing or receiving assets, and making payments. Separation of related duties refers to the situation where different users are given distinct, but often interrelated tasks such that a failure of one user to perform as expected will be detected by another. For separation of related duties to be effective, computer capabilities must be partitioned. These capabilities must be accessible only to users or processes associated with specific tasks. For example, for many financial applications, a common requirement was to separate users who authorize or commit the expenditure of funds from those authorized to place orders for services and equipment. The IRS has used this policy as a requirement from the outset and a system with this capability was developed especially for them.

Access Based on the Principle of Least Privilege

The principle of least privilege was described by some of those interviewed as an important control approach in meeting security policies and objectives. This principle gives the user no more privilege than is necessary to perform a job. Implementing least privilege requires identifying the user's job, determining the minimum set of privileges required to perform that job, and restricting the user to a domain with those privileges. Least privilege allows a user to have different levels of privilege at different times, depending on what task is being performed. By denying access to transactions and privileges that are not necessary for the performance of their duties, those privileges cannot be used to circumvent the organizational protection policy. Least privilege is particularly important for those systems where there is a "privileged user" or "superuser" capability that otherwise grants a wide set of privileges to users that need only a subset of those privileges.

The principle of least privilege is similar to separation of transactions. It differs in that separation of transactions restricts the set of programs that can access data and places restrictions on which users can execute what programs. Least privilege restricts a user's access to data by denying users privileges that are not necessary to do their job.

Several organizations expressed the need for an operating system capability that supports the principle of least privilege. This capability is currently supported in upper end secure systems (B2 (2) and above), but many organizations expressed the desire to see this capability at a more basic level.

Label-Based Mandatory Access Controls (MAC)

MAC is a non-discretionary access control which restricts users' access to data on levels implemented through labels. These are (1) the level associated with the trust of the user, i.e., clearance, and (2) the level associated with the sensitivity of the data.

For many, the ability to isolate and share information on a non-discretionary, formal "need-to-know" basis is required. The formal "need-to-know" has primary emphasis on categories and, to a lesser extent, on hierarchical levels. The term "category" is used to describe non-hierarchical separation. Outside the DoD, few organizations use hierarchical levels. Most non-DoD organizations have employees and users belonging to one level, but with different responsibilities, i.e., in different categories.

For example, a commercial organization recently formed strategic alliances with some of its competitors. Although this organization has always allowed access to some of its most sensitive proprietary information by outside consultants, this access was narrow and limited. Because of the frequency and scope of past access requirements, physical and procedural measures could provide the necessary isolation. However, as its corporate relationships changed, so did the need for access controls. The reality is that those other organizations are only partners on one front, while still fierce competitors on another. Further, when the access needs of a third and fourth partner are considered, which are different from the first, the physical and procedural controls of the past become impractical. The only real solution may be label-based mandatory categorization. The significant issue is "who can read what information?" Controlling the flow of a specific type of data from one category (say company A), to another (say company B) is where TCSEC MAC applies.

As business alliances become a corporate reality of many companies, the ability to rely on label-based mandatory access controls becomes more important.

Object-Label Association

The ability to associate a label (not necessarily used in access decisions) with an object was described as a needed capability by several organizations interviewed. These labels carry warning, advisory, and other information associated with an object and are not used for making mandatory access control decisions. For example, within a hospital system, a label associates a warning with a prescription drug, i.e., "not for use if person has high blood pressure". The association between a drug and a warning is an important relationship. For medical systems in general, the capability to associate an information label describing the quality of an x-ray, CATscan, sonogram, or any other image shared among medical professionals, can be a vital capability.

CONCLUSION

Each organization viewed its access control needs as unique. Access control mechanisms need to be applied on a case-by-case basis in meeting individual computer security threats. Frequently the available products, which incorporated more limited access control policy objectives, lacked adequate flexibility to be easily and cost-effectively adapted to the variety of functional environments in which they were applied.

This paper has examined a variety of traditional and evolving access control policy objectives. If these were accepted by vendors and users, and incorporated into commercially marketed systems and products, users would have available a valuable set of sought-after protection tools. Armed with an expanded, widely accepted set of control policy objectives, and products based on them, users would have greater assurance that their protection needs at the operating system level, the application level, the organizational level, and the site level were being met.

REFERENCES

1. Assessing Federal and Commercial Information Security Needs, Ferraiolo, D., Gilbert, D., and Lynch, N., NISTIR 4976, November 1992.
2. U.S. Department of Defense Trusted Computer System Evaluation Criteria (TCSEC), DoD 5200.28-STD, December 1985.
3. Minimum Security Requirements for Multi-User Operating Systems: A Protection Profile for the U.S. Information Security Standard, National Institute of Standards and Technology, 1992 draft.
4. Federal Criteria for Information Technology Security, National Institute of Standards and Technology & National Security Agency, 1993 draft.
5. "Comparison of Commercial and Military Computer Security Policies", IEEE Symposium on Computer Security and Privacy, Clark, D.D. and Wilson, D.R., 1987
6. "Role Based Access Controls", 15th National Computer Security Conference Proceedings, Ferraiolo, D. and Kuhn, R., 1992
7. A Guide to Understanding Discretionary Access Control in Trusted Systems, NCSC-TG-003, September 1987.
8. Computers at Risk - Safe Computing in the Information Age, National Research Council, National Academy Press, 1991.

An OPEN SECURITY ARCHITECTURE

**A discussion of a "workstation-centric"
architecture designed to provide control
and security for critical data in an
environment of PC workstations,
networks and mobile computing.**

© 1993
Fredric B. Gluck
Datamedia Corporation
Security and Control Products
20 Trafalgar Square
Nashua, NH 03063 USA
Tel: 603-886-1570
Fax: 603-598-8268
dmcnh!gluck@uunet.uu.net
CSI: 72143,524

INTRODUCTION

This paper outlines an Open Security Architecture (OSA). OSA is an architecture that will provide the basis for the selection, design and integration of products providing security and control for a network of desktop personal computers, "mobile" notebook computers, servers and mainframes. The purpose of this architecture is to provide an environment where:

- acceptable and workable controls can be placed on sensitive data.
- user productivity and existing investments in applications are not negatively impacted by the addition of control and security.
- data flow around the organization, and the investment that has been put in place to support this capability (e.g. local-area, wide-area, and telephone-based networks), can still be used to enhance information exchange between users.
- all workstations, regardless of their location, operating system, or capability to connect to a network, can be included and easily administered under this architecture.

BACKGROUND

With the increasing implementation of solutions driven by "downsizing" and the increasing use of PCs to access, compile and generate sensitive data, the need for information control and protection has grown dramatically. This need has been created because in the move toward downsized applications and the addition of networks as an integral part of the data processing solution, there has been a large-scale, rapid migration of data from the glass wall of the computer centers to distributed points on a LAN that are much closer to the end-user. Today, files, data, messages and resources that five years ago would not have existed outside the secure environment of the data center, are now routinely found in local workgroups and remote offices.

With this dramatic change in data processing architecture, questions such as the following are often asked:

- How is protection and control for sensitive data being applied?
- How is the growing explosion of "mobile computing" driven by laptops and notebook PCs effecting how sensitive data is being moved throughout the organization?
- How can data be protected when and if it moves outside a secured area?

Although many different opinions exist as to the right answers to these questions, decision makers charged with finding answers to these questions have already concluded that whatever solution they choose, the existing investment in hardware, software, applications and networks must not be devalued by the addition of control and security.

Questions and concerns such as these are the driving force behind a number of solutions that have been developed to assist in the control and security of data. For ease of analysis, these solutions can be divided into three groups:

Server-based solutions: These solutions are designed to protect data when and if the data resides on the server. These solutions, usually part of the Network Operating System file system, give the System Administrator the ability to designate certain files, directories and volumes as protected areas with certain characteristics such as READ-ONLY or NO ACCESS. Although these solutions may be practical in some situations (i.e. diskless workstation environments), server-based solutions suffer in three areas. First, files have to be physically resident on the server to be protected. Second, this type of protection cannot be applied to notebook PCs or other mobile computing solutions. Finally, users must be actively involved and alter their work methods to make this solution effective.

Physical link: Physical link products use encryption engines at either end of a hub and are designed to prevent interception of data and information as it flows from the workstation to the server. These solutions certainly provide protection for data while it is in transit between network hubs, but do nothing to protect data that is at rest on either a workstation or a server.

Workstation-based products: Workstation add-on products protect data residing on a workstation. These products rely on some type of file protection and encryption scheme and allow or prevent access to a file based on a user's identity. The shortcoming of these solutions is that most of these products make the assumption that protected data remains on a workstation and will not be securely moved between users with such tools as Electronic Mail.

Today, the problem that organizations are facing as they implement one or more of these related, but architecturally disconnected, security methods is how these solutions work together to provide an environment where data is protected at rest, where data is protected in transit, and where the exchange of data between users is not hindered.

A CLOSER LOOK AT THE CRITICAL ISSUE

Given the need to protect data in transit and at rest and retain an existing computing and network infrastructure, the first task in developing an overall architecture to solve the problem is to gain a clear understanding or model of how data moves through an organization. Understanding this model will provide a foundation that allows security and control to be gently overlaid onto this infrastructure.

A Model of Moving Data

With today's available technology, data moving through an organization can take many paths. As an example, examine the computing and electronic mail infrastructure that most office workers have at their disposal. This infrastructure allows a document to be created from data that may have been gathered from a number of different sources, some of which may be considered sensitive. As data arrives at a user's workstation, it is processed and combined with other sensitive and non-sensitive data, then easily distributed to other members of the organization. In the process of being distributed, this data takes different paths, and can pass through many different computing platforms and operating systems. The recipient can in fact end up receiving and reading the document on a workstation with a different hardware architecture and different operating system than the system on which the document was created.

Adding mobile computing users to this model raises its level of complexity in two ways. First, information created at a secured workstation can be distributed to users with notebook PCs through unsecured areas using E-mail, modems and other services. Second, data residing on mobile PCs is more vulnerable to theft and compromise because of its location and easy portability.

Highlighting the fact that sensitive information can be a part of this scenario raises a whole new set of questions. If the user mentioned above creates a document containing sensitive information, how does this user distribute the document to others and how does this user ensure that the sensitive information in the document is protected from the point where it was created to the point that it is read by the recipient? Sadly, the way that many organizations are trying to answer this question is to prevent their data processing and communication infrastructure from being used to transport data that is sensitive to the organization. The effects of this decision are the creation of "decision bottlenecks" that develop because sensitive information has to be moved or processed in an inefficient manner.

Why Is an Architecture Needed?

The issue of data protection involves the proper balance of investment, return and available technology. Attempting to balance these three items without an effective architecture for applying security and control to the PC/LAN environment will most likely make effective data protection cost prohibitive. An unbalanced solution will also require a forced integration of a number of unrelated and possibly incompatible myriad of solutions.

Further support for this statement can be found by examining the range of solutions available to solve the problem of data security in a mixed, mobile environment. At one end of the cost/return curve are a number of highly secure solutions used by government agencies. Although these solutions provide very secure environments, they are most likely cost prohibitive for many commercial implementations. At the other end of the curve are "commodity", off-the-shelf products that are inexpensive, but may not provide a suitable level of protection. This raises questions such as:

- Am I willing or able to invest a large amount of dollars to get a "maximum" level of security?
- Am I willing to risk weak security so I don't have to invest heavily?

Perhaps the most important questions of all are:

- Is there a suitable solution that provides an adequate level of security at a reasonable cost? What will this solution look like and what are the architectural features of such a solution?

Why Server-Based Security Falls Short of Addressing this Issue

In an attempt to answer these questions and address the issue of data control and security, many organizations have tried implementing a type of security scheme that places corporate servers at the hubs of enforcement. These types of solutions provide only a minimal level of protection for sensitive data as it moves around the organization, and they have a number of shortcomings.

At its lowest level, server-based security forces users to identify themselves with a user identification and password before access to the server is granted. This is suitable except that it leaves the workstation vulnerable to unauthorized access.

In some cases, a higher level of file or directory level security is implemented by providing users with a secure "vault" on the server where sensitive information can be placed. Unfortunately, this scheme does not match the way most users work. Simple observation of a cross-section of users will most likely show that these users gather, create, then store data on the local hard disk of their local, private workstation. Few, if any users, can be encouraged to place their sensitive information in this "server" vault on a regular and disciplined basis.

To further support this fact, one only needs to analyze how organizational servers are used. These servers probably has shared programs (such as E-Mail) and shared data areas. If private, protected area for each users have been created, they are probably empty and unused.

If performing this analysis is not convincing enough, ask users how many of them have taken the responsibility to regularly move their data to a backup storage area. If users have difficulty performing this simple task, what expectations can you have that users will place their secure data into a vault area?

OSA vs the Proposed OSI Security Architecture

Currently under discussion is an architecture designed to add security functionality to the Open Systems Interconnection Reference Model (OSIRM). This OSIRM (and GOSIP) architecture targets security and protection for a very specific area of the overall network. As described in an article titled "Building the New OSI Security Architecture" by Dan Minoli, which appeared in the June 1992 edition of Network Computing Magazine. . .

"OSI security functions are concerned only with those aspects of a communications path that permit open systems to achieve secure transfer of information between each system." The article continues, "It is important to understand that the OSI Security Architecture is not concerned with the security of hosts or servers seen as discrete entities -- it is not concerned with security measures needed within the systems themselves or any given installation. The definition of security services to support security measures in end systems, installations and organizations is outside the scope of the standard."

Thus, the proposed OSI architecture is designed to provide a set of services designed to secure point-to-point communications and connections between endpoints in a network environment. It is designed to protect data in transit from one place to another. No doubt these functions are a critical part of the overall security solution. However, if only solutions described by OSI are implemented, large amounts of data at rest on workstations are still left unprotected. What is required is an architecture that **complements** OSI and specifically addresses security measures for information that resides in systems at the network endpoints (i.e. workstations and mobile computers).

DEFINITION OF AN "OPEN SECURITY ARCHITECTURE"

The Open Security Architecture described in the remainder of this paper is proposed as an alternative to "server-centric" security and as a complement to the OSI architecture. Open Security Architecture (OSA) takes a "workstation-centric" approach to the problem and has been developed to address this problem of implementing affordable, "open" security and protection for sensitive data. It assumes that data and protection begin at the workstation and flow outward through the organization.

The remainder of this paper will examine the design goals and features of OSA, present a list of functions provided by OSA-compliant products and describe how these functions address the issues raised above. As presented here, OSA will assist in determining how to implement security and control throughout an organization.

Design Goals of OSA

The design goals of OSA were developed to provide data integrity, maintain data availability and provide authorized access to the data at a reasonable cost, in mixed environments of PCs, networks and mobile computing without significant negative impact on user productivity.

The 6 design goals of OSA are as follows:

OSA DESIGN GOAL #1: Information Flow and Mobility

OSA assumes that a solution for data control and security should not impede the flow of information around an organization. If users are used to sending and receiving unsensitive information through links such as electronic mail, this capability must be usable by those persons who want to move sensitive information around the organization. Security solutions embracing OSA should not impede the mobility of those users who rely on notebook PCs. These solutions must take into account how these mobile systems can participate in the overall organizational security scheme.

OSA DESIGN GOAL #2: Time of Unprotection

With OSA-conformant solutions, the entity to be protected (e.g. a file) should be protected automatically and transparently at the time it is created. It should never exist in an unprotected state. This criteria dictates that when a file is created, protection should be immediately applied. The protection should be applied as specified by the organizational security policy and should not require the file's creator to take any action at all. Protection of entities at their creation time removes the responsibility from the user of having to move items from an unsecured workstation to a server-based "vault".

OSA DESIGN GOAL #3: Protection as an Integral Part of the Data

Protection applied to a file will always remain with the file regardless of where the file is located or how the file is transported. Protection should be in effect no matter where the file physically resides. If, for example, the file is attached to an E-mail message and sent to another user, the protection applied to the file should not be altered. If the file is copied to a diskette and surface-mailed to a regional office, the protection should remain with the file. It should not be necessary to risk unprotecting the file so that it can be transferred to another user.

OSA DESIGN GOAL #4: Centralized Administration

For cost effectiveness, the security scheme implemented with OSA-compliant products must be able to be administered from a central location under the control of a single organizational security officer. This security officer must be able to implement the organizational security policy and bring all endpoints (workstations) under his protection. Provisions must be made to handle endpoints that are local users connected via a network, or remote users who have no formal communication facilities to the central site.

Provisions must also be made to provide centralized administration in heterogeneous environments of workstations and servers. For example, from a central point, a Security Officer should be able to perform administration functions for a workstation executing DOS and Windows as well as for a UNIX-based workstation. Implementing this level of administration requires an organized, central database of user, workstation and other information that can be accessed to perform these functions.

OSA DESIGN GOAL #5: Integration with Existing Infrastructure

The security scheme must be implemented without disrupting the existing program, data and network infrastructure that is in place. It must function effectively regardless of the type of network or transport capabilities that are in place.

OSA DESIGN GOAL # 6: Modularity

The most cost-effective solution must be modular. This means that you can begin to implement security and control at any point on the cost/return curve shown in Figure 1. The point of the curve you select will depend on the level of security you require and the amount of resources you can invest. Most importantly, as additional security is needed and additional resources become available, the move up the curve to higher levels of security must be a smooth process. You should not have to nullify or discard any of your existing investment or resources.

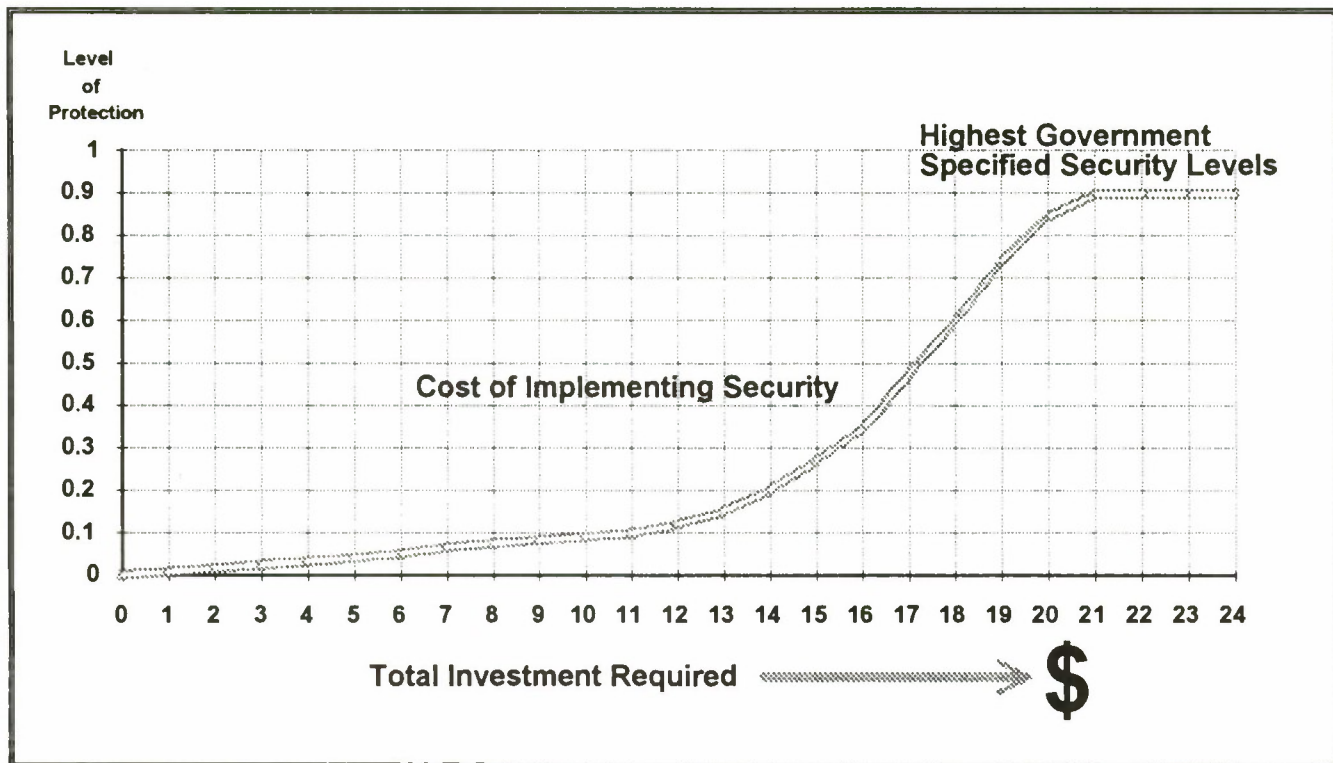


Figure 1 - Cost Return Curve for Implementing Security and Control

SECURITY AND CONTROL FUNCTIONS PROVIDED BY OSA PRODUCTS

Given both the model of data movement outlined above and the design goals of OSA, we can now examine the specific functions and purpose of products built to conform to this architecture.

The six central functions of OSA-compliant, "workstation-centric" solutions are as follows:

Identification and Authentication (I&A)

I&A is designed to ensure that only authorized users have access to protected items (e.g. workstations, files and directories). I&A begins when the workstation is powered on and the user is required to supply a valid user identification and password. Once the user is verified, the system can securely and assuredly pass this information to other parts of the security system or other parts of the organization that require it (e.g. servers and mainframe computers). I&A should be performed at one location and passed around the organization as required.

For proper implementation of I&A, there needs to be different levels and options for verifying a user's identity.

At the entry level of I&A, OSA products provide an option for user-ID and password verification. This solution is ideal for low-end PCs and for PCs that cannot have hardware easily added to them.

For mid-range solutions, I&A can be based on the concept of something known and something possessed. This is the level of security that one typically finds in Automated Teller Systems. Users are issued a "token" that functions as a secure container for user identification and other security related information such as encryption keys. A protected system cannot be activated until the user presents a token (something possessed) and a valid password (something known). When properly implemented, this token can not only grant access to a protected PC, but can also be used to grant access to other secure areas of an installation.

At the high end of the I&A spectrum, OSA provides a method to address rigorous I&A using clearly defined Application Programming Interfaces (APIs). This more rigorous I&A requirement might call for the integration of biometric devices such as fingerprint scanners.

Secured I&A (Common Sign-on) to Hosts and Mainframes

To deal with the myriad of systems that a single user might access, OSA specifies a Common Sign-On (CSO) capability. CSO can be invoked to securely and transparently pass user I&A information from a workstation to another system such as a mainframe. When combined with other architectural features of OSA, this CSO capability allows all connection procedures to hosts and mainframe systems to be designed and controlled by the Security Officer. CSO also ensures that all access to a host originates from a secure workstation and that a host is screened from unsecured and uncontrolled logins.

Protection of Files

File protection uses a set of information about the user to control (i.e. grant or deny) access to a file. This information is obtained both from the I&A process and from a secured database of user profile information.

Functions that provide file protection are implemented with the following goals in mind:

- The amount of time that a file remains unprotected must be kept to an absolute minimum. As soon as the file is created, protection should be applied.
- The protection applied to the file should travel with the file as it moves through the organization.
- The protection should remain in force at all times regardless of how the protected file is transported.
- The integrity of the data should not be effected. Protected data should remain available to all current applications that were in use before security was implemented. For example, a spreadsheet file should remain available to an application regardless of the protection that has been applied.
- The security-related information contained in or with a file must be able to be laterally transferred between systems of different types. For example, file protection applied on a DOS based system and interpreted by a DOS-based security kernel must be translated as needed so that when a protected DOS file is sent to a UNIX system, the file protection remains intact.

How OSA Implements File Protection

OSA-conformant products provide file protection using a technique that makes protection and control an integral part of the file itself. This protection is implemented by placing a "label" on a protected file that pairs user identification with the actions that the user can perform on the file. It is similar to placing an address label on an envelope that details who can open and read the contents.

Labeled file protection functions as follows. :

When a user or security officer takes the action to protect a file, there are three items that must be specified. These items are:

- **The owner of the file.** An owner has complete control over the file, is granted full access to the file and is the only person who can alter the contents of the file's label.
- **Users of the file.** Users are any person in the "user community" that may attempt to access this file.
- **Privileges.** Privileges specify what actions a user can perform on a protected file. Privileges are paired with each user who may attempt access to the file. Each pair of user/privileges specifies what capabilities the user has when access to the file is attempted.

The owner, user, and privilege information is combined with other control information, formatted into a label and attached to the file as a "header". After attaching the label, additional protection can be invoked by transparently encrypting both the label and the entire contents of the file. Figure 2 outlines this procedure.

All accesses to protected files are strictly controlled by the security system and rely on the information in the label. Whenever a file is accessed, the security system checks the file to see if it is protected. If the file is protected, the user is granted only the capabilities for access as specified in the file's label.

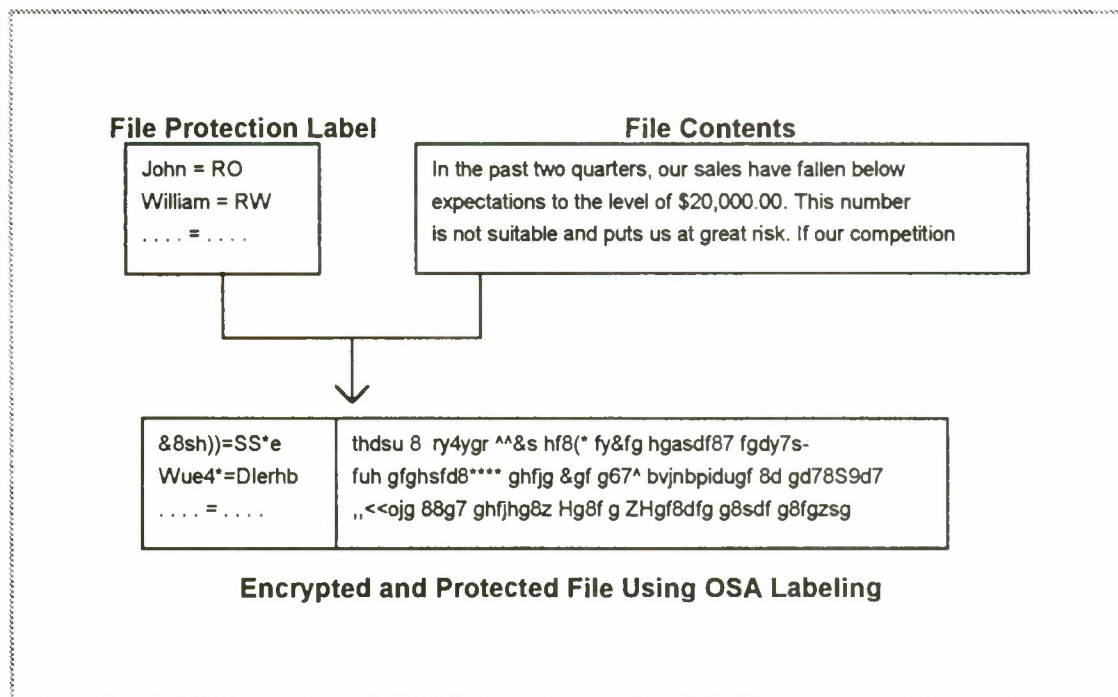


Figure 2 - File Labeling

Note specifically that protecting a file does not change the overall structure of the file. This is why, with regards to file maintenance, transport and location, protected files can be treated in the same manner as unprotected files. In addition, because the label is made an integral part of the file, the file's protection travels with the file as it moves throughout the organization.

Administration

Experience has shown that if the administration of a solution is too expensive or requires too many resources to be effective, the solution will most probably not be implemented. Effective security administration as specified by OSA and shown in Figure 3 ensures the following:

- **Central Point of Control:** The most effective way to implement security is from a single central point. OSA specifies that a single central point (usually an organizational security officer) should be the person who has the capability and responsibility to implement the organizational security policy. This capability is referred to as Central Site Administration (CSA).
- **Trusted Users to Assist the Security Officer:** To assist the Security Officer in performing day-to-day duties, OSA specifies that a number of Trusted Users can be created by the Security Officer. These Trusted Users can perform a subset of security duties as needed. Using this scheme, for example, the SO could appoint a group of trusted persons to act as auditors. The auditor's responsibility would be to gather, convert and analyze audit data as necessary. This scheme of trusted users also allows the SO to appoint a set of users in field offices to take care of the daily chores of user registration and assistance. These trusted users could operate under the organizational security scheme, but do not have the capability to alter it. They offload the single Security Officer from having to deal with day-to-day operational problems and issues.

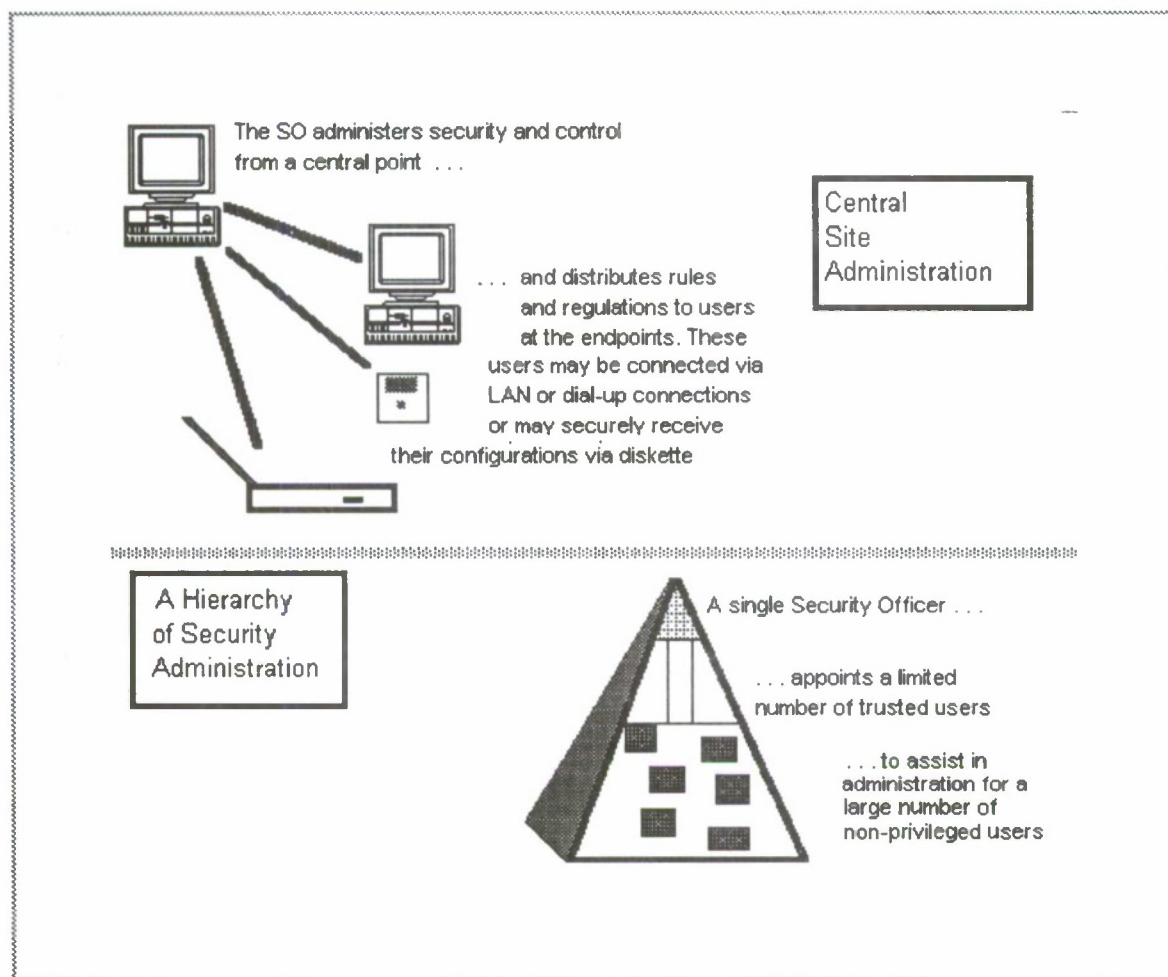


Figure 3 - Effective Security Administration

- Minimal User Involvement at the Endpoints:** To keep the security administration and implementation simple, OSA specifies that there must be automated procedures available to install and change the security and protection configuration. At a minimum, users should only be required to execute a simple procedure to install the base security software and a security configuration built for them by the Security Officer or Trusted User.
- Application of the Security Policy to All Endpoints:** With the increasing number of mobile computer systems (notebook PCs) that may not always have access to a LAN, it is important to be able to bring mobile systems, remotely located systems and local systems into the organizational security scheme. The method specified by OSA to accomplish this is an extension of Central Site Administration. OSA specifies a method so that security configurations can be securely transferred to a user over a network (perhaps via electronic mail or shared file areas) or if a network is not available, via electronic transfer using a modem or via a floppy diskette.
- Cross Platform Application of the Security Policy:** To deal with heterogeneous networks, the Central Site Administration portion of the OSA architecture should be able to be applied to workstations executing different operating systems. This will require a centralized "repository" of information to coordinate user registrations and other information.

Auditing

The function of auditing under the OSA architecture is to provide a constant evaluation of the strength of your security scheme. Auditing should provide the Security Officer with a set of manageable data that can be analyzed to detect where in the security scheme violations have taken place. With the results of this information, the security policy can be adjusted to plug any "leaks" that have occurred.

With this purpose in mind, auditing as specified in the OSA architecture is designed to provide the following functions:

- **Recording of Critical Events:** As part of the organization's security policy, a set of events that defines a "security violation" must be defined. This set of events is used by the Security Officer to determine what information is placed into the audit log. Proper design and implementation of an auditing subsystem as specified by OSA dictates an approach that is granular. A granular approach allows the SO to audit only the events that are important and preserves workstation or server disk space required to keep the log. The granular approach also prevents "information overload", a condition that arises when the SO is overwhelmed with so much data that the time required in analysis does not produce valid results about the effectiveness of the security scheme.
- **Uploadable Audit Information:** Auditing the critical events of an installation requires that the record of events from many, widely-dispersed workstations be collected at a central point, and then combined for analysis. The OSA architecture allows audit files to be output in standard "database-ready" format and transferred via Electronic Mail or other means to a central site. This allows the Security Officer the means to prepare an overall organizational-wide view of the effectiveness of the security policy using analysis and database tools that most likely are already available.
- **Ease of Use:** So that breaches in security can be immediately detected at the workstation where they occurred, OSA specifies that the SO should have a set of easy-to-use, basic analysis tools available. These tools are designed to provide an "instant view" of workstation activity with a minimal effort and should not destroy or alter any data that may later be consolidated at a central site.
- **Real-Time Security Alerts:** Workstations connected to LANs have the additional advantage of a "real-time" connection that is always active. By extending the capabilities of existing network-management tools and combining them with, data-link drivers and local auditing capabilities, a centrally located Network Administrator can be alerted if and when a security related event (such as an failed I&A at a workstation) occurs.

Creation of Security-Aware Applications

As more and more mission critical applications are "downsized" from the mainframe to the workstation, it is necessary to make provisions to ensure that these applications are executed in some sort of controlled environment. OSA addresses this need by specifying a method for making applications "security-aware". A security-aware application has security as an integral part of its functionality and design. As an example, security-aware applications may be built with the following capabilities:

- **Use of I&A Information:** The application begins execution only after it obtains the user identity from the security kernel. It may force the user through a process of reauthentication to ensure that the same user who booted the system and was cleared through I&A is in fact the same user who is executing the application.
- **Execution only in a secured environment:** The application will not execute on an unsecured system. Such an application may be designed so that it only executes on a system that has been secured, or on a system that has been designated by the SO to be a member of a certain group of workstations. This controlled execution ensures that if the application is unlawfully moved to another system, it will refuse to execute. This capability is ideal for sensitive, workstation-based applications that deal with the electronic transfer of funds, for example.

The benefit of providing the ability to make applications security-aware is that the results of authentication are made available and information such as the User-ID, User Privileges, File Access Capabilities, Group Membership, etc. can be accessed and used as needed.

BENEFITS OF OSA

After examining the main features of the OSA architecture it is practical to briefly mention the benefits that can be expected by implementing control and security products that conform to this architecture.

Implementation

OSA-conformant products are designed so that they can be "gently overlaid" over an existing data processing infrastructure. This means that all existing applications, networks, and datafiles will not have to be altered as security and control is implemented. Not having to alter applications, disk structures or other critical items on existing PCs means that the cost of implementation can be kept to a minimum.

Network Transparent

Since OSA makes use of a labeling scheme that does not alter the structure or integrity of a workstation file (e.g. DOS files), OSA products can be implemented regardless of the type, number, or mix of networks that are in place. As long as your existing servers and networks can transport and store files in their native format, OSA products can be used to apply security to these files. OSA does not require that existing files or secured files be "translated" between formats.

Authorized Information Flow

OSA products are designed so that they will not inhibit your flow of information around the organization. The file labeling scheme used by OSA does not affect the capability of your E-Mail system to move files to and from many endpoints. In fact, use of OSA and the file labeling scheme can encourage use of these systems and speed the flow of critical information since files can now be moved over wide areas after they are protected, and both endpoints can be assured that the file will only be accessed by those who have authorization.

Low Administration Costs

Through Central Site Administration, OSA products allow administration of a large, widely distributed base of systems. The Security Officer at a central site can not only bring users under the organizational security policy, he can, if necessary, give some security duties to Trusted Users at these remote sites. With Central Site Administration, it is not necessary for a SO to visit or even have a network connection to all of the PCs under his control. Central Site Administration does not burden the end user with a long, detailed, drawn-out installation process and does not require the user to be literate in the application or control of the organization's security policy.

Little or No Impact on User Productivity

Perhaps the most important benefit of products designed to the OSA architecture will be that these products will, for the most part, be transparent to the user. Users on systems secured with these products will not have to alter their applications or the way they do their work. Products implementing OSA should be designed to operate at a level where security is enforced, and users are only notified when a security violation occurs. There are no additional menus or commands that users have to learn in order to use a secure system.

SUMMARY

This paper has outlined a workstation-centric architecture that brings control and security into an organization by beginning at a point where the data is created (i.e. the workstation) and moving outward.

Although different from most of the other piecemeal security solutions available today, this design provides a flexible set of criteria and functionality that can be used to evaluate a wide range of data control and security solutions for a mixed environment of workstations, servers and mobile computers. This evaluation framework, when implemented, will lead to the addition of control and security in a manner designed to preserve the investment in, and enhance the use of your existing information infrastructure.

REFERENCES

1. Carnahan, Lisa J. *A Local Area Network Security Architecture*, Proceedings: 15th Annual National Computer Security Conference, October 1992, pp 340-341
2. Minoli, Dan, *Building the New OSI Security Architecture*, Network Computing Magazine, June 1992

ADMINISTRATION OF ACCESS RIGHTS IN A MULTI-VENDOR SYSTEM – A CASE HISTORY

Lee J. Becker
Defense Mapping Agency
Information Systems Directorate
8613 Lee Highway
Fairfax, VA 22031-2138

Craig A. LaBarge
Martin Marietta
Management & Data Systems
P.O. Box 8048
Philadelphia, PA 19101

William S. Buonanni
Martin Marietta
Management & Data Systems
P.O. Box 8048
Philadelphia, PA 19101

Abstract

The administration of access rights becomes increasingly more complex as different vendor platforms are combined to make up a single system. Each vendor has unique methods for employing access controls. The Defense Mapping Agency's Digital Production System (DPS) is a large, multi-vendor system which operates in a classified environment. The DPS's tightly-coupled architecture and geographically-dispersed installations created unique challenges in the development, implementation, and administration of a system-wide access control policy. This paper describes the approach used to categorize the DPS users based on job functions, how the access needs for each category were determined, and describes the development of a system-wide model for the assignment of access rights and privileges. The methods used for implementing and maintaining the access control information are also provided as well as an assessment of how well this approach is working.

Keywords: Access control, computer security, multi-vendor, policy, administration, access rights, privileges

Background

The Defense Mapping Agency (DMA) Digital Production System (DPS)

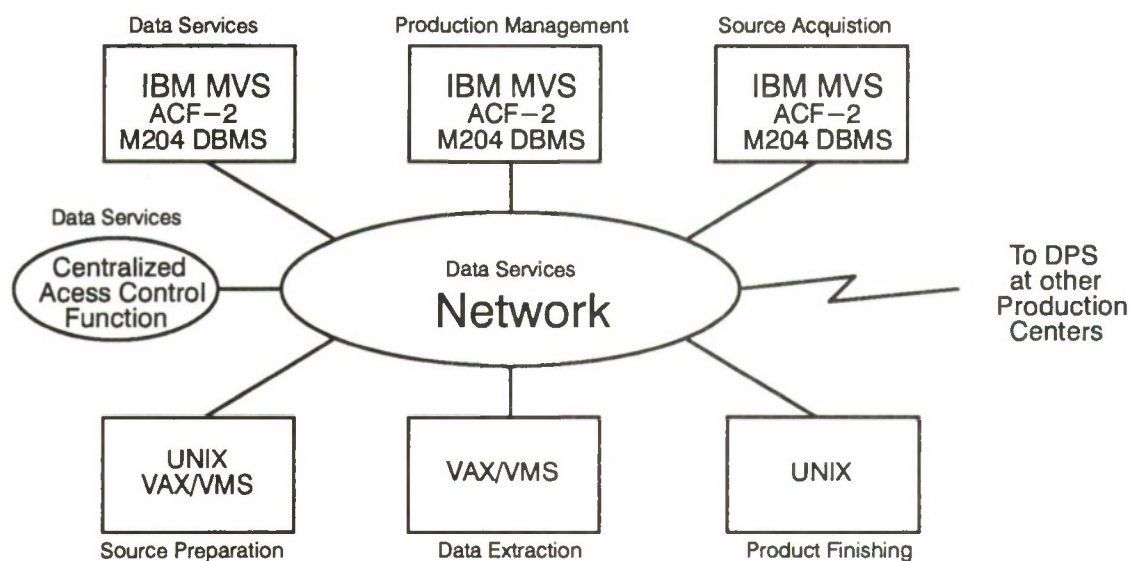
The DPS is the culmination of the DMA's Modernization Program whose mission was to streamline the existing production processes and utilize an all digital source technology. The DPS functionality is designed to support a wide variety of Mapping, Charting, and Geodetic (MC&G) products and spans the entire production process, from initial planning through product finishing. The DPS development effort produced approximately 8.5 million lines of application software code. Successful operation of the DPS requires a large user population with a diversity of roles and functions and an equally diverse range of applications and data types.

The DPS includes six interconnected segments at each of three DMA production facilities. There are three data server segments and three production (client) segments at each location. The data server segments are the Data Services Segment (DS/S), the Production Management Segment (PM/S), and the Source Acquisition Segment (SA/S). The production segments include the Source Preparation Segment (SP/S), the Data Extraction Segment (DE/S), and the Product Generation Segment (PG/S). The DS/S provides centralized security services, communications services which include device access authorization tables, and provides for the archiving and retrieval of audit trail information. The other segments perform their missions in addition to providing security mechanisms for the local enforcement of access policies. The primary DPS outputs at each center are Digital Product Masters which are copied and distributed to customers and output in lithographic media ready for hardcopy reproduction and distribution.

System Architecture

The DPS system was designed and developed to support a “system high” security operating mode. The requirements for the system were defined in early 1985 which pre-dated the Department of Defense (DoD) adoption of the National Computer Security Center (NCSC) Trusted Systems Evaluation Criteria (TSEC)[1]. The DPS is a multi-vendor system environment: IBM mainframes (DS/S, PM/S, and SA/S) using ACF2, VAX/VMS (SP/S and DE/S), UNIX (PG/S and SP/S), and some application specific (proprietary) security features. Essentially, the DPS provides C2 functionality with the exception of object reuse. The architecture is semi-distributed (see Figure 1) in that system access control is centrally supported by DS/S while data access control is allocated to the segment which stores the data. All DPS users are authenticated by DS/S prior to accessing any DPS resources. Requests for access to data stored in the DPS are adjudicated by the segment holding the data.

Server Segments



Production (Client) Segments

Figure 1. Overview of DPS Architecture at a Single Production Center

Problem Statement

With the potential for up to 8,000 DPS user accounts across the three DMA locations, the DPS presented an enormous security administration challenge. This challenge was initially realized during the architecture development. Due to cost considerations and the full complexity of the task not being fully understood, the capability to automate access control administration was not included in the DPS design. During segment-level testing, the administration of access control within individual segments was manageable and worked well. However, during the integration of all six segments into a single system, the lack of a system-wide procedure frequently halted verification activities and access denials began to delay the overall implementation.

The DPS architecture requires off-line pre-coordination of user access rights between segments. To create a typical DPS user account, access rights and privileges must be established in multiple segments. A user identifier and system password must be created in DS/S; local data and function access rights must be established in the user's home segment; and, depending on the user's function, data access rights may be needed in one or more server segments. A standardized approach was required to ensure consistent application and enforcement of security policies across the system. Incomplete specification of access

rights denied a valid user access to necessary resources, interrupting production; while specifying access rights too broadly provided a potential for unauthorized access to data and resources.

Problem Solution

Given the DPS implementation schedule, a mechanism for security administration had to be quickly developed to facilitate: (1) system-level testing, (2) system demonstrations, and (3) exercise and rehearsals prior to start-up to a full production environment. The primary objective was to establish security authorizations for contractor and government personnel participating in the delivery and start-up activities (up to 75 people per segment) and allow those personnel to quickly gain access to the system, in accordance with security policy, while relieving the system engineers and cartographic personnel of the detailed implementation of administrative security procedures.

After the primary objective was achieved, the focus shifted to development of a new procedure which allowed for a smooth transition of production personnel into the DPS operations environment. To accomplish the objectives, the approach taken built upon the existing requirement allocation, performed during the system design phase, where requirements had been allocated to operations. A standard access control model was developed for the system, categorizing the DPS users according to function. These categories were derived from segment staffing and training plans and thus provided the basis for building "user access groups." Once the groups were defined, a standard access profile was constructed for each group which defines: (1) data access across the entire system, (2) device access requirements, (3) and specific system/segment functionality which are required to perform the users' activity. A procedure was then developed based upon the access profiles to be used by security administration personnel to establish user accounts in a timely and uniform fashion across all DMA components; a highly desirable goal for any security administration function [2]. A major goal was to replace an existing 12-page System Access Request form with a greatly simplified one-page form.

The following sections expand upon the profile development process, as well as the implementation methodology and maintenance approach.

Profile Development

The development of a system-wide model for user access rights was primarily a manual effort involving the following steps:

1. Determine the access control policies to be enforced
2. Categorize the system users by function or role
3. Determine the access rights required to fulfill each function or role
4. Develop a system-wide convention for documenting the user profiles

The result was a low-cost approach which provided an integrated view of DPS access control policies and minimized impact to the system design and transition activities.

Policy

Since the access profiles were intended to serve as a detailed system model for access control, it was first necessary to understand the more general policies applicable to the system. The user profiles and the detailed access rules to which they refer must accurately reflect the policies they are expected to enforce. A set of guidelines was developed to document all known constraints on the granting of DPS access rights. The guidelines were derived from several sources including DoD and DMA security requirements and policy, customary DMA security practices, and architecture-driven constraints. Examples of DPS policy guidelines are:

- All DPS users are authenticated by the DS/S centralized access control function.
- Production users are not permitted access to operating system command line functions.
- Access to security functions are limited to the segment ISSOs.

Since several persons were involved in compiling the profiles, a common set of groundrules helped to achieve consistency.

Categorizing the Users

To simplify the administration and maintenance associated with access rules, the DPS adopted a policy of specifying access rules in terms of functions or positions rather than individuals. When a user's job function changes, that user need only be associated with a new group; in general, no access rules need to be changed. The practice of creating access rules on a user-by-user basis is avoided in order to ensure that access controls are uniformly implemented at all DPS installations.

An initial list of user groups for each segment was developed by examining the segment functionality being provided, the segment and system-level operations concept documents, and the segment staffing plans. The groups were defined solely in terms of the specific functions they perform on the system and not in terms of the DMA organization to which they belong. This was done to make the DPS access control rules immune to any future change in the DMA organizational structure. A one-to-one correspondence between the names given to the user groups and the actual DMA job titles was not always possible. In many cases, several job titles may be associated with a single DPS user group for access control purposes. A mapping of job titles to DPS user group names was provided as part of the implementing procedure.

This effort resulted in the definition of approximately 85 DPS user groups. Approximately 25 percent of the groups consist of users directly involved in MC&G production process. The remainder of the user groups contain supervisory, administrative, and system support functions.

Determining Access Rights and Privileges

By far, the most difficult and time-consuming part of the process was determining the rights and privileges each type of user requires in order to perform their function on the system. This required a careful mapping between user groups, the functions they perform, the data they require or produce, the devices they need to access, and the level of privileges they require. The access rights needed to be broad enough so as not to impede system usability, but narrow enough to preserve data confidentiality and integrity.

This task required the synthesis of information from a variety of sources. System interface control documents (ICD) provided the technical requirements for the interactions between the segments including the use of network security services and the allowable data exchanges. Production models for the DPS provided information regarding the data inputs and outputs for each DPS production task. This information, along with the knowledge of the functions performed by the DPS user groups, allowed for the initial definition of DPS access profiles.

A matrix was created for each segment relating the user groups to the data rights, device rights, and privileges they require. These matrices were carefully analyzed to determine if there was an appropriate level of granularity. Wherever possible, groups with similar access rights were combined in the interest of simplicity. Groups were combined only when it could be determined that data confidentiality and integrity would not be significantly degraded. In some cases, the matrix showed two or more groups with identical access rights except for one or two privileges. The groups were merged if the extra privileges could be eliminated without degrading functionality or if the risk of retaining the privileges was deemed acceptable. Conversely, data integrity concerns occasionally led to the definition of additional user groups having more tightly-defined access rights.

User directly involved in DPS production are typically restricted to a small set of applications and require few system privileges. This allowed production users to be organized into a relatively small number of groups, with each group having a large number of members. The remainder of the users, mostly system support personnel, were organized into more numerous groups with fewer members in each group. This approach was taken to allow a finer granularity of control over powerful system privileges.

The involvement of the segment development contractors and the production organizations in this exercise was an absolute necessity. The development contractors provided critical information regarding the interactions between the applications and the security features. The production organizations, being the end users of the system, provided invaluable insight into operational use of the system.

Documenting the Profiles

Having defined an access profile for each group of DPS users, a common format was used to document the profile information. The profiles document internal segment access rights as well as access to system-wide data and resources. Standard conventions were developed for use in specifying access rights between segments. As shown in Figure 2, the content of each profile consists of five sections:

Data/Function accesses. This is a broadly-defined category in order to accommodate the wide variety of access control implementations used across the system. When access rights are defined in terms of specific files or high-level data qualifiers, a standard notation was used to indicate the segment in which the data is stored, the data type or file name, the access modes permitted (e.g., read, write, etc.), and the segment(s) in which access rights must be registered. (The latter information is required since some data transfer operations involve multiple segments.) Depending on the access control implementation in a given segment, this category might also list the specific menu panels that may be accessed or specific segment functions that may be performed. This section may also specify that users falling under this profile are to be associated with a user group that has been predefined within a segment using the features of ACF-2, VAX/VMS, UNIX, or other means. For example, a computer operator within a given segment might be assigned the appropriate set of rights by simply associating that user with a locally-defined (i.e., segment-specific) group for access control purposes. Regardless of the form that access rights take across the system, the profile indicates the rights required for each type of user and the segment in which those rights are to be established.

Device Rights. The network access control function checks for valid device identifiers at logon and prior to establishing network connections on behalf of the user. To support the establishment of user device rights, the access profiles provided a list of generic device types from which a user may logon as well as other network hosts to which a user may establish logical network connections. Devices are indicated using a system-wide naming convention for network devices.

Data Base Privileges. These include any privileges required by user in order to access data from within a data base management system (DBMS). A user accessing data within one of the IBM main-frame-based segments may require privileges to be established within the DBMS in addition to those established within the ACF-2 security software. Data base privileges are identified using the syntax required for the particular DBMS.

Segment-level Privileges. These are privileges which may be granted at the segment level and generally granted only to the users of that segment. For example, the profile for an ISSO of an IBM-based segment might include the ACF-2 privileges "SECURITY" and "ACCOUNT" while the profile of a VAX/VMS-based segment might include the privileges: "SECURITY" and "SYSPRIV."

System-level Privileges. These are privileges provided by the network which may be granted to users in any of the attached segments. One common example is the THIRD PARTY LOGOFF privilege which allows a privileged user, such as the Segment ISSO, to perform a network logoff of another user within that segment.

The access profiles are stored off-line from the DPS in a personal computer-based data base maintained by the System ISSO. The profiles are distributed in hardcopy form to Segment ISSOs at each DPS location.

Procedural Implementation

The concept of system-wide user access profiles was introduced by integrating it into a simplified procedure for processing requests for DPS logon accounts, keeping the technical details transparent to the user community. The procedure defines the coordination between segment ISSOs that must take place in setting up system-wide access rights for a new user. Figure 3 shows an overview of the coordination procedure.

By providing the access profiles to the ISSOs as a system-wide model for granting access rights, the process of setting up access rights across the system is made transparent to the users. To request an account for a user, the user's supervisor needs only to provide basic identifying information about the user, indicate

PROFILE #: RSP01	<div style="border: 2px solid black; padding: 5px; display: inline-block;"> DPS ACCESS CONTROL PROFILE </div>	1/15/93		
<table style="width: 100%; border: none;"> <tr> <td style="width: 35%; border: 1px solid black; padding: 5px;"> SEGMENT: SP </td> <td style="width: 65%; border: 1px solid black; padding: 5px;"> JOB POSITION/FUNCTION: CARTOGRAPHER </td> </tr> </table>			SEGMENT: SP	JOB POSITION/FUNCTION: CARTOGRAPHER
SEGMENT: SP	JOB POSITION/FUNCTION: CARTOGRAPHER			
<div style="display: flex; justify-content: space-between;"> <div style="width: 48%;"> <p style="text-align: center; margin-bottom: 10px;"><u>DATA SUBGROUPS/OTHER</u></p> <p><i>(System-wide data access rights)</i></p> <div style="margin-bottom: 20px;"> <div style="display: flex; justify-content: space-between;"> <div> (PM) PRFEAS (W) (PM) (SA) SADATA (R) (SA/DS) (DS) HISELO (R) (DS) </div> <div style="text-align: right;"> <i>Segment(s) in which permissions are required</i> <i>Access mode (i.e., read, write)</i> <i>Data high-level qualifier</i> <i>Segment in which data is located</i> </div> </div> </div> <p>OTHER INTERNAL SEGMENT PERMISSIONS:</p> <div style="margin-bottom: 20px;"> .LOGIN.TA (VAX ONLY) .PROFILE_ADHOC (UNIX ONLY) </div> <p><i>Local, segment-specific access rights. The exact nature and syntax varies from segment to segment.</i></p> </div> <div style="width: 48%;"> <p style="text-align: center; margin-bottom: 10px;"><u>EQUIPMENT/DEVICE ID</u></p> <div style="margin-bottom: 20px;"> A84 (ALL) C85 (ALL) H19B002 H10A001 </div> <p><i>Host and workstation rights using standard system device identifiers</i></p> </div> </div>				
<div style="display: flex; justify-content: space-between;"> <div style="width: 48%;"> <p style="text-align: center; margin-bottom: 10px;"><u>DATA BASE ACCESS</u></p> <div style="margin-bottom: 20px;"> (DS) PSWD, X'FF', HIGH, ALL </div> <div style="display: flex; justify-content: space-between;"> <div style="width: 45%;"> <i>Segment in which permissions are required</i> </div> <div style="width: 55%;"> <i>Data base permissions in segment-specific syntax</i> </div> </div> </div> <div style="width: 48%;"> <p style="text-align: center; margin-bottom: 10px;"><u>SEGMENT PRIVILEGES</u></p> <p>VAX PRIVILEGES:</p> <div style="margin-bottom: 20px;"> GRPNAM DETACH GRPPRV SHARE </div> <p><i>Local segment privileges</i></p> </div> </div>				
<div style="display: flex; justify-content: space-between;"> <div style="width: 48%;"> <p style="text-align: center; margin-bottom: 10px;"><u>SYSTEM PRIVILEGES</u></p> <div style="margin-bottom: 20px;"> CHANGE PASSWORD </div> <p><i>Privileges granted by centralized access function</i></p> </div> </div>				
<p><u>MISCELLANEOUS:</u> <i>(Any additional ISSO information or instructions required)</i></p>				

Figure 2. Access Control Profile Example

the user's job function, and certify the user's clearance and need-to-know. The ISSO in the user's home segment determines the applicable profile based on the user's job function and establishes local segment access rights according to the profile.

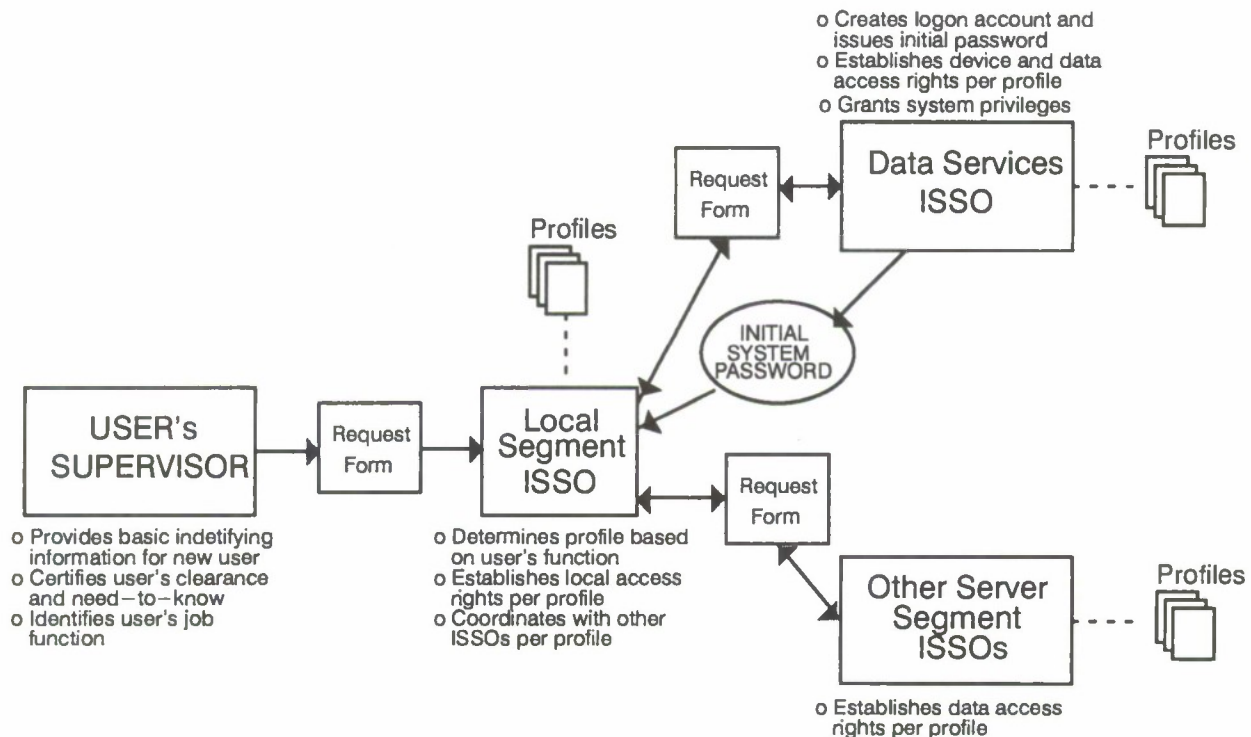


Figure 3. Access Request Coordination

The profile also indicates the other segments in which the user will require access rights. The local segment ISSO can then forward the request form to the ISSOs in the other affected segments. These segment ISSOs establish rights for the new user according to the profile, sign the form, and return it to the originating segment ISSO for tracking purposes. At a minimum, all requests for new accounts are forwarded to the local Data Services Segment ISSO who establishes the network logon account and issues a temporary password. Since all intercenter data transfers are accomplished as process-to-process transactions, users do not normally require accounts at other DPS locations.

Profile Maintenance

Several aspects of the development of access control rules are analogous to the development of software. Even the most carefully developed access control rule set will contain bugs and will require changes and updates over its lifecycle.

Leveraging off the similarity with software maintenance, the same configuration management tools used for software problem reporting are also used to report DPS access control problems. The Segment ISSOs use DMA's Automated Configuration Management System (ACMS) to report problems and track them through resolution. ACMS, with terminals located at all DMA production centers, allows for proposed access control changes to be coordinated between the System ISSO and the ISSOs of all affected segments.

In practice, discrepancies reported via ACMS generally involve errors in the profiles or errors in entering the profile information into a segment's access control lists, or failure to follow the coordination procedure. ACMS is also used by the segment ISSOs to recommend improvements or refinements in the access profiles.

If the coordination results in the need to change a profile to correct a problem, the System ISSO updates the profile and distributes hardcopy updates to all holders of the access profiles. In emergency or time-crit-

ical situations, Segment ISSOs are authorized to make temporary changes to the profiles, provided that a follow-up discrepancy report is entered into ACMS for formal coordination and resolution.

Current Assessment

The profiles and coordination procedures have been used successfully at all three DPS locations for over a year. After conducting initial training sessions and procedure walk-throughs for ISSOs and supervisory personnel at each center, the concept of profiles was readily accepted. Although users are advised to allow five working days for the processing of access requests, the coordination process is often completed in less than one day.

The number of discrepancy reports involving access control problems dropped sharply in the first few months following implementation of the procedure. Prior to implementing the procedure as many as 50 discrepancy reports were received in a single month. Today, there are typically 1 or 2 reports of problems at most in a given month and most are attributable to routine maintenance of the profiles due to changes in configuration or operations. Access denials due to errors in specifying access rules have become a relatively infrequent occurrence.

Having a system-wide model for access rights allows the segment ISSOs to quickly diagnose and resolve the few problems that do occur. The profile maintenance procedures that have been established provide the ISSOs with a means of recommending changes and enhancements to the profiles, continually adding to the quality and accuracy of the profiles.

Summary

In this paper we described an inexpensive and practical approach to the administration of access control in a large, complex, multi-vendor system. We described the methodology used to establish a uniform set of user groups and to define an appropriate set of access rights and privileges for each group. We also described a method for the ongoing maintenance of the access profiles that takes advantage of existing configuration management procedures.

References

- [1] DoD Directive 5200.28, "Security Requirements for Automated Information Systems (AISs)," 21 March 1988.
- [2] Janus Associates, "Information Security Administration Model: A Management Model to Help Identify the Best Practices of the Administration Function Within the Security Group," Computers & Security, July 1992, pp 327-340.
- [3] Jonathan D. Moffett and Morris S. Sloman, "The Source of Authority for Commercial Access Control," IEEE Computer, February 1988, pp 59-69.

USE OF THE TRUSTED COMPUTER SYSTEM EVALUATION CRITERIA (TCSEC) FOR COMPLEX, EVOLVING, MULTIPOLICY SYSTEMS

Howard L. Johnson
Information Intelligence Sciences, Inc.
1903 So. Franklin Street
Denver, Colorado 80210
(303) 777-4266

Melvin L. De Vilbiss, MAJOR (USA)
National Security Agency
9800 Savage Road
Fort G. G. Meade, MD 20755-6000
(410) 859-4463

Contract: This work was partially accomplished under contract number MDA904-91-C-6011. The viewpoints expressed here are of the authors and not necessarily those of the Government.

ABSTRACT

This paper deals with complex systems, systems made up of systems. To make the protection problem manageable, we divide the complex system into pieces, addressing each piece the way simple systems are now treated with the Trusted Computer System Evaluation Criteria (TCSEC). Each piece, called a domain of constant policy (DOCP), has a single policy supported by a single TCB (division/class).

As in a simple system, we determine division/class using DoDD 5200.28, Enclosure 4. Using the DOCP's associated n-tuple (n operational security policy parameters such as clearances and classifications), a risk index is identified, subject to modification by the Designated Approving Authority (DAA).

Connected DOCPs are subject to cascading risk, requiring a search that considers each pair of potentially intercommunicating DOCPs. Identified risk increases can result in an increased risk index, called exposed risk index. This is a primary factor used to determine DOCP division/class. Risk contributing DOCPs are candidates for operational policy changes or added mechanisms.

Optimal operational policy is determined through requirement and design iteration, i.e., seeking lowest affordable risk. A revised division/class selection is assigned to a DOCP based on this iterative process. An interface policy is developed, constraining communications to conform to all security policies, including local policies, e.g., two-man rule, and mutual suspicion. Global policy is developed across DOCPs, consistent and mutually supportive in areas such as identification/authentication, audit, and trusted recovery. The result is a better defined DAA certification and accreditation objective, which helps to more precisely define the procurement specifications.

INTRODUCTION

The purpose of this paper is to provide a methodology to assist the heads of DoD components at procuring, certifying, and accrediting complex, evolving, multipolicy systems against the TCSEC [1] requirements, consistent with and accommodating the guidance provided in the Trusted Network Interpretation (TNI) [2] of the TCSEC and the Trusted Database Management Interpretation (TDI) [3] of the TCSEC. This proposed methodology can be used when any or all of the following system characteristics exist: a) complex - the system is made up of systems; b) evolving - part of the system exists and the rest of the system is being added; and c) multipolicy - different parts of the system support different policies requiring different modes of operations, hence, different divisions/classes.

Eight new terms critical to the concepts of this paper are introduced and defined below. Other important terms are taken from the TCSEC [1], the TNI [2], the TDI [3], and DoDD 5200.28 [4]. The new terms are:

Domains of Constant Policy (DOCPs) - Unique pieces of the system, each with a single policy and an associated TCB. DOCPs were first introduced in [5] and became Air Force Guidance in [6].

N-tuples - Operational security policy parameters associated with a DOCP used to determine division/class. "n" may be different for each procurement. A basic set of n-tuple parameters, e.g., classification and clearance, is required to determine the remaining derived parameters, e.g., risk index, exposed risk index, mode and division/class. See the last paragraph of the "Domains of Constant Policy" section below for an example.

Operational Security Policy - Design and operational choices that satisfy regulatory security policy, e.g., the 5200.28 documents [1, 4, and 7]. This policy includes established operational policies (DOCPs and security parameters (n-tuples)), and security rules of operation [8, Section 2.3].

Exposed Risk Index - An adjusted risk index for a DOCP determined from DoDD 5200.28 [4], Enclosure 4, that considers exposure (cascading risk) from other DOCPs.

Contributed Risk - The summed amount of increase in exposed risk potentially contributed by a single DOCP to all other DOCPs. Two DOCPs could potentially increase the risk index of a third DOCP from its original level, i.e., providing an exposed risk index. However, in the analysis, only the highest level of calculated exposed risk from any one of the contributing DOCPs is used to increase the risk index of the third DOCP. Nevertheless, each of the contributing DOCPs receives an increase in contributed risk.

Solely Contributed Risk - The risk contributed by a DOCP which could not have also been contributed by another, summed across all other potentially contributing DOCPs.

Interface Policy - Policy established for control of data flow between each pair of communicating DOCPs.

Global Policy - System level requirements to be satisfied by all DOCPs, e.g., audit, recovery, and identification/authentication.

DOMAINS OF CONSTANT POLICY

DOCPs are, in general, nonoverlapping subsets of the system, that, in combination, completely cover the system. In figure 1, we see a composition of an automated information system. If we divide the system into well defined pieces then define their interface policies with one another as well as the global policy each piece must support, we use the DOCP methodology. A DOCP consists of a well-defined boundary, where an isolation mechanism exists or can be employed, and an n-tuple defining security characteristics. The isolation is required to ensure that communications is taking place only over known, designated channels. Each DOCP will have a TCB for support of its own security requirements, however, some of the mechanisms, e.g., audit, may be shared with another DOCP. This is the only

exception to the nonoverlapping principal. The n-tuple that represents operational policy can be simple, e.g., clearance and classification levels, or complicated, e.g., with categories and other parameters.

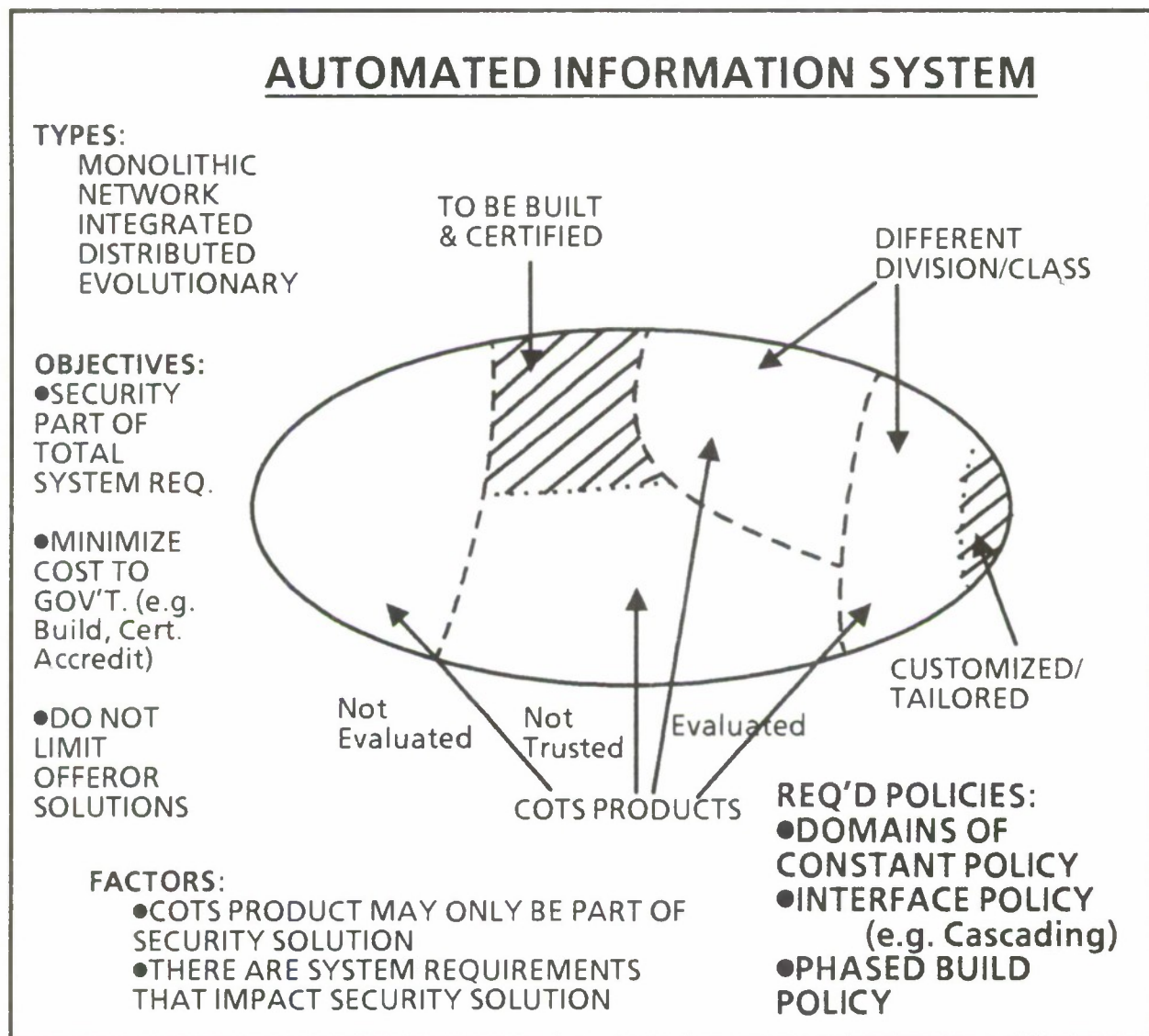


FIGURE 1: AUTOMATED INFORMATION SYSTEM

For the purposes of this document, the n-tuple parameters considered to be basic are values of the parameters: minimum classification of data; maximum classification of data; minimum security clearance; maximum security clearance; categories, e.g., compartments/caveats; build status, e.g., existing, EPL product [9], to be built; and level of assurance achieved, e.g., EPL evaluation at some level [9], certification evaluation at some level, no evaluation, or other. Those n-tuple parameters considered to be derived are: risk index, exposed risk index, mode, and division/class. Thus in this case $n = 11$, where 7 are basic and 4 are derived.

RISK ASSESSMENT

A DOCP and its n-tuple are working entities in the sense that tradeoff decisions based on a propagated risk assessment concerning policy, costs, and mechanisms

may make it necessary to change the (one or more) DOCPs and their characteristics. Many of the concepts of propagated risk were presented in [10]. It is only after these adjustments are completed that the derived policy parameters, e.g., exposed risk, mode, and TCSEC division/class, are finalized.

A small part of the risk management process for simple systems is the risk assessment procedure identified in DoDD 5200.28 [4], Enclosure 4, that identifies a **risk index** using some of the operational security policy, with other considerations, to guide the DAA in making adjustments. The same procedure is used for DOCPs with the exception that the cascading risk from intercommunicating DOCPs is also taken into account. Exposure is represented by changes to the operational security parameters, i.e., one or more of the the n-tuples, before Enclosure 4 is applied. The exposed risk is a new risk index value called the **exposed risk index**.

Contributed risk is the summed amount of increase in exposed risk potentially contributed by a single DOCP to all other DOCPs. As explained in the terms listed above, two or more DOCPs could have potentially changed the risk level of yet another DOCP from its original level, but in the analysis technique, only one is considered. Nevertheless, they all receive an increase in contributed risk. **Solely contributed risk** is the risk contributed by one DOCP which could not have been also contributed by another.

The **exposed risk** can be decreased by changing either the local operational policies or the operational policies of the contributing DOCP(s). The contributed risk factors are an indicator to the DAA where the changing of policy or the implementation of guards may do the most good in reducing the risk of the overall system. This is all done before mechanisms are considered, thus, as you might guess, this is the first of two iterations. The two contributed risk factors, contributed risk and solely contribute risk, help identify to the DAA the areas where changes in operational security policy can have the largest risk reduction advantage. It is here, identification of the largest risk reduction advantage, that this methodology will have the greatest (most positive) impact on the certification and accreditation requirements. Manipulation of the operational security policy, based on propagated risk analysis, will provide the DAA the most flexibility in cost/risk tradeoff decisions while defining the eventual certification and accreditation objectives. This in-turn, better defines the design and development requirements, hence procurement specifications are better defined. The propagated risk assessment is repeated to assess the shared risk aspects of the adjustments.

INTERFACE POLICY

There needs to be an explicit interface policy considered between each DOCP and every other DOCP with which it communicates. The interface policy can be thought of as an augmentation to the exportation policy of the TCSEC, however, in many cases, both exportation and importation concerns are expressed. The need for a trusted path to share and mediate security variables also should be assessed. In sending data, a DOCP must support intercommunication (exporting) policies established by its division/class.

A DOCP has two interface responsibilities: 1) it must ensure that data it **sends** continues to be supported by the policies imposed on it and, 2) it must appropriately handle data it receives based on any policy information known about that data.

The policy can be discretionary and/or mandatory and includes categories, e.g., compartments, caveats, need to know. The responsibility for establishing the policy, linking it to the data, and assuring proper understanding by the receiver is required of the sender. Policy can be preestablished based on data identification through DOCP agreements, communicated via labels, or communicated and implemented manually by security administrators.

Sending DOCPs must be assured that data is being released into a system that can be trusted to interpret and carry out the policy. Factors to consider include the potential for eavesdropping, spoofing, or policy alteration.

Once data is in the possession of a receiving DOCP, it becomes the responsibility of that TCB to impose its knowledge of the policy on that data and treat it accordingly. Suspected or actual violations of interface policy must be treated as a special case and the data protected.

A DOCP may not be affordably and certifiably able to support division/class increases determined by considering exposed risk. Special communications mechanisms or added protection features within a potential receiving DOCP may help to ameliorate this situation, i.e., decrease the exposed risk. This can provide an operational solution that must be agreed to by a potential sending DOCP. In any case, the DAA from a sending DOCP has the ultimate responsibility for adequate enforcement of his or her own DOCP security policy.

In a policy of mutual suspicion, a sending DOCP must establish interface policy consistent with the level of trust it has established for potential receiving DOCPs. If the level of trust determined does not coincide with the certification and/or accreditation level given that DOCP, the sending DOCP should further restrict the communication policy, beyond that normally implied by the TCSEC and its interpretations, to a level where the sending DOCP is willing to accept the remaining risk. Similarly, if a receiving DOCP cannot trust the content or policy associated with data provided by another DOCP, then a receipt and handling policy must be established consistent with the risk the receiving DOCP is willing to accept. This policy may be more restrictive than that required by the TCSEC and its interpretations.

GLOBAL POLICY

Global considerations pertain to systems for which there can be or has been no accreditation against a well defined global policy such as that stated in the TDI. If TCBs share mechanisms, e.g., identification/authentication or audit, each individual TCB must be certified alone, using that mechanism. The DAA must use the evidence from those certifications to ensure consistency with interface policy between the entities and any policy of which this shared mechanism is a part.

To be secure, either there shall be no sharing between DOCPs of discretionary controlled data, the entire connected system should satisfy a single previously established discretionary access control policy, it must be accomplished by sharing access control mechanisms, or DOCPs must share access control information between mechanisms, ensuring a secure protection and a system that cannot be defeated because of time lags and communications threats. In older systems that do not allow subjects to access objects in other systems, this requirement is often satisfied because only standard messages are formatted and allowed to be transmitted. In these cases the subjects do not have access to objects beyond the scope of their own TCB.

Even if each TCB has its own data for identification and authentication, the information for individual users that may potentially request access in more than one TCB or may have access to objects in more than one TCB, must be consistent. The individual cannot assume more than one identity or be performing two functions simultaneously, unless the system security has accounted for such support. There must be a way to associate audit records generated by different TCBs for the same individual subject.

Someone must be assigned the authority and assume the responsibility of security administrator for each of the TCBs. In addition, a security administrator must represent the authority of each hierarchical stage of DAAs.

Implications of failure of one of the component TCBs must be reviewed from the standpoint of impact to all of the other intercommunicating entities. A way to cooperatively shut down and recover in a secure manner must exist.

Component TCBs following the subsetted TCB principles set forth by the TDI need not be concerned with additional interface and global policies beyond those stated within the TDI.

PROTECTION ASSESSMENT

With the operational policy (DOCP and n-tuples), interface policy, and global policy established, design can be accomplished based on the divisions/classes chosen. Upgrades to existing architectures will probably involve providing mechanisms to support the global and interface policies. System and TCB isolation may need to be enhanced. Compensation for previously ignored exposed risk may involve manual or automated guards, and strict interface control. Some mechanisms may be replaced to take advantage of technology advances. New and replacement designs will take advantage of EPL products [9] where possible.

Besides protection mechanism assessment, there needs to be an assessment of assurance. This includes determining the evaluation rigor used, or planned to be used, in testing and evaluating the DOCP. In both upgrade and new systems with EPL products, a strategy for certification must be developed that maximizes the use of prior evidence, while not diminishing the quality of the assurance.

It is at this point a second iterative analysis should be undertaken to take into account the success of the proposed mechanisms in meeting the regulatory and operational security policy. It allows reexamination of the process all the way back to the specification of operational policy. The two contributed risk factors, i.e., contributed risk and solely contributed risk, again help identify to the DAA the areas where changes in operational policy can have the largest risk and cost reduction advantage. The protection assessment can be redone considering actual architectural solutions. What remains is a statement of the residual risk within the system. The DAA must determine the acceptability of the risk and, if required, the process must be reviewed and corrected.

The results of this second iterative analysis may cause revisions to the operational security policy and security architectural design. At this point, new development may begin. The (revised) operational security policy is used along with regulatory security policy as a basis for defining certification and accreditation objectives.

These certification and accreditation objectives should then be translated into procurement objectives.

The DOCP approach does not preclude use of the TNI and TDI in any way. It does, however, deal with the cases in which those interpretations cannot be employed. In using the TNI, the network system, the policy, and the NTCB become associated with a single DOCP which uses the TNI and TCSEC in the specified manner. It is then possible to develop an interface policy and a global policy which considers this network DOCP as part of an even larger system. When using the TDI, the rules are employed to determine more and less primitive TCBs within a DOCP. The TDI system policy can then be treated as a policy or a single TCB within a larger system.

SUMMARY

Manipulation of the operational security policy will provide the DAA the most flexibility in cost/risk tradeoff decisions while defining the eventual certification and accreditation objectives. This will facilitate more precise procurement specifications. We recommend DOCP use as the conceptual approach in complex, evolving, multipolicy systems for the following reasons:

- o It offers a solid, intuition supported approach for procurement administrators and DAAs
- o It requires the statement of operational policy (DOCPs and n-tuples) from the using organization and architecturally reflects that in the design
- o It enforces precise system covering boundary definition
- o It allows, and in fact encourages, cost/risk tradeoffs and iteration of operational policy assignment
- o It can be applied to pre-regulatory (TCSEC, TNI and/or TDI) systems where the interpretation of TCB must be made
- o It does not preclude, and in fact supports use of the TNI and TDI
- o It forces consideration of cascading risk, requires interface policy, requires global policy
- o It accommodates/promotes use of EPL products since the basic building block entity of a system (a DOCP) has a single policy represented by a division/class requirement of the Orange Book
- o It addresses security interface requirements to be satisfied if an EPL product component is going to be integrated into the overall security of the AIS system which may contain other EPL products, existing secure systems, or "to be custom built" specifications

NCSC-TG-024, A Guide to the Procurement of Trusted Systems [11], a four volume series, pertains to simple systems and is in the process of publication. A goal is to develop a version of this guideline series to be consistent with the ideas of this paper. Volume 2, Language for Specifications and Statements of Work, will be the first document to be revised as a STRAWMAN using this concept.

The STRAWMAN was delivered to the Government in March 1993, as Howard Johnson's last deliverable under contract before his passing on 14 May 1993.

BIBLIOGRAPHY

- [1] DoD 5200.28-STD, "Trusted Computer System Evaluation Criteria," December 1985
- [2] NCSC-TG-005, "Trusted Network Interpretation (TNI) of the Trusted Computer System Evaluation Criteria (TCSEC)," National Security Agency, July 31, 1987
- [3] NCSC-TG-021, "Trusted Database Management System Interpretation (TDI) of The Trusted Computer System Evaluation Criteria (TCSEC)," National Security Agency, April 1991
- [4] DoD Directive 5200.28, "Security Requirements for Automated Information Systems (AISs)," March 21, 1988
- [5] Johnson, H.L., "An Approach to Security Test," AFCEA 4th Annual Symposium on C3I Technology Information and Security, Philadelphia, PA, 16-18 August 1988
- [6] AFSSM 5031, "Complex System Guide, Air Force Special Security Manual," Air Force Cryptologic Support Center, Air Force Intelligence Command, 1991
- [7] DoD 5200.28-M (Draft), "Automated Information System Security Manual," April 29, 1991
- [8] NCSC-TG-010, Version 1, "A Guide to Understanding Security Modeling in Trusted Systems," October 1992
- [9] "Information Systems Security Products and Services Catalogue," Prepared by the National Security Agency, (Published Quarterly)
- [10] Johnson, H.L., and J.D. Layne, "Modeling Security Risk in Networks," Proceedings 11th National Computer Security Conference, NIST and NCSC, October 17-20, 1988, pp. 59-64
- [11] NCSC-TG-024, Version 1,
 - Volume 1/4, "A Guide to Procurement of Trusted Systems: An Introduction to Procurement Initiators on Computer Security Requirements," December 1992
 - Volume 2/4, "A Guide to Procurement of Trusted Systems: Language for RFP Specifications and Statements of Work - An Aid to Procurement Initiators," (In publication)
 - Volume 3/4, "A Guide to Procurement of Trusted Systems: Computer Security Contract Data Requirements List and Data Item Description Tutorial," (Draft)
 - Volume 4/4, "A Guide to Procurement of Trusted Systems: How to Evaluate a Bidder's Proposal Document - An Aid to Procurement Initiators and Contractors," (Draft)

A PROTOTYPE DISTRIBUTED AUDIT SYSTEM

By

Debra L. Banning
SPARTA, Inc.
615 Nash St.
El Segundo, CA 90245

Security auditing systems are used to detect and assess unauthorized or abusive system usage. Historically, security audits were confined to a single computer system. Recent work examines ways of extending auditing to include heterogeneous groups of computers (distributed systems). This paper describes the design and prototype development of a Distributed Audit System (DAS) which was developed with funding received from Lawrence Livermore Laboratory and through the Master's thesis effort performed by the author at California State University, Long Beach. [1] The DAS is intended to provide collection, transfer, and control of audit data on distributed, heterogeneous hosts.

INTRODUCTION

The problem to be solved by the prototype is: "How can we control audit data amongst heterogeneous hosts in a network?" It has been a long time goal of intrusion detection designers to provide a means for determining abnormal activity amongst users on a stand alone system. However, when different types of systems become interconnected in a network, interoperability between the systems becomes lost. The primary need for interoperability lies in defining a standard communications structure between different types of hosts. To do this it is also necessary to define a standard set of information that would be collected amongst different types of operating systems.

The DAS prototype is designed to provide this communications structure and framework for a standard definition of manageable audit data. The DAS prototype uses network management protocols and a graphical user interface to provide control over security audit data at distributed hosts from a centralized location. It is designed to take advantage of, but not duplicate, the many intrusion detection systems currently available or under development.

AUDITING AS A NETWORK MANAGEMENT FUNCTION

Network management protocols provide a mechanism for transmitting network performance information from remote nodes to a central collection point. The collection and reporting process for performance data and audit data are very similar. From review of the protocols it was determined that the network management protocols could be adapted for collecting, reporting, and transmitting audit information in a distributed network. This section gives a brief description of network management protocols and their applicability to distributed auditing.

Introduction to Network Management

Network management is accomplished by managers at local management stations and agents at remote managed nodes exchanging monitoring and control information via protocols and shared conceptual schema about a network and its components. The shared conceptual schema mentioned above is a priori knowledge about "managed objects" concerning which information is to be exchanged. Managed objects are abstractions of system and networking resources (e.g., a protocol entity, an IP routing table, or in this case, auditing resources) that are subject to management. Managed objects have attributes, operations, and notifications that are visible to managers. The internal functioning of the managed object is not visible to the manager. Currently, an agent is responsible for conversions between a managed system's internal format of managed objects and the external format of managed objects (i.e., the form expected by the manager).

Using management services and protocols, a manager can direct an agent to perform an operation on a managed object for which it is responsible. Such operations might be to return certain values associated with a managed object (i.e., get a variable), to change certain values associated with a managed object (i.e., set a variable), or perform an action, such as self-test, on a managed object. In addition, the agent may also forward to the manager notifications generated asynchronously by managed objects (e.g., send updates periodically).

Network Management Architecture

The Network Management architecture described here consists of a Management Information Base (MIB) containing a list of managed objects, the International Organization for Standardization (ISO) Common Management Information Services (CMIS)/Common Management Information Protocol (CMIP) Manager and Agents [2]. The Managers and Agents exchange information based on the managed object definitions contained in the MIB, and the ISO network management protocols that facilitate the exchange of this information.

CMIS/CMIP Manager and Agents

The Common Management Information Services (CMIS) are the set of services provided by the Common Management Information Service Element. The Common Management Information Protocol (CMIP) supports these services. A CMISE-service-user is the part of an application process that makes use of the Common Management Information Service Element. An invoking CMISE-service-user, or "manager", may invoke a management operation. A performing CMISE-service-user, or "agent", is the process that performs a management operation invoked by a "manager."

CMIS/CMIP supports a full set of basic services to facilitate standardized communication between CMIP managers and CMIP agents for monitoring and controlling network resources. The CMIP application can be run over a full OSI stack of protocols; however, it is also possible to run CMIP over a TCP/IP transport stack. In either case, CMIP always uses the ISO application layer services of the Association Control Service Element (ACSE) and the Remote Operations Service Element (ROSE). ACSE is used to establish and release associations between application entities. ROSE is the ISO equivalent of a remote procedure call. ROSE allows the invocation of an operation to be performed on a remote system. A CMIS/CMIP manager and agent applications could use adaptations of the ISO Common Management Information Service Element (CMISE) to exchange information and commands for the purpose of auditing.

CMISE provides to managers the ability to "multicast" operations to be performed on a group of managed objects. Through CMISE services, a manager can perform a single operation on a group of managed objects. A distributed audit mechanism could use such a service to assist in responding interactively to network attacks.

CMISE also provides facilities for a managed "agent" to send multiple linked responses to a manager. An Audit Agent (AA) could use this type of service to send detailed information to an Audit Manager.

Management Information Base (MIB)

A MIB is a list of managed objects, described in external format, which are considered useful for a particular application. A managed object is an abstract representation of a network resource that is subject to management. The objects of the MIB are defined in terms of their attributes that can be affected by management protocols. CMIP-based AAs instantiate the objects that are defined in the MIB and CMIP-based AMs operate on those MIB objects.

A MIB has been developed for the management of the Internet. The Internet MIB contains managed objects considered essential for either fault or configuration management. The managed objects can be read-only or read-write, and help a manager determine the status of his network elements. Using the Internet MIB as a model, an Audit Management Information Base (Audit MIB) can be developed.

The Audit MIB would define the managed objects upon which the CMIP-based Audit Managers may operate. The definitions of these managed objects would be derived from end-user requirements. In the DAS, the end-user requirements come from the needs of the audit analysis technique used at the AM. The Audit MIB must define managed objects such that the AM's audit analysis tool would be capable of detecting abnormal behavior at the AA. Once the managed objects have been defined in the Audit MIB they are not removed. Commands issued from the AM are used to create/delete instances of the defined managed objects such that information may be obtained about them.

It is envisioned that an Audit MIB will encourage growth in distributed audit applications as has been seen in the area of Network Management. Implementations of AAs will have a standardized set of objects (the Audit MIB) that they will instantiate, therefore, it will not be necessary to create new agents for each new audit application. Likewise, implementors of distributed auditing applications will be able to concentrate on the specific application and not have to invent all of the supporting processes. Complete standardization of the Audit MIB is being addressed by standards committees and will take the cooperation of the computer security and vendor community.

Note that the Audit MIB object definition is written in Abstract Syntax Notation (ASN.1) as defined by ISO/CCITT [3]. This notation provides a mechanism for uniquely defining the semantics and syntax of the objects in a machine independent fashion that increases the scope of interoperability for the Audit MIB.

DISTRIBUTED AUDIT SYSTEM DESIGN AND PROTOTYPE

The DAS design consists of 4 primary components:

- (1) Audit Agent (AA)
- (2) Audit Manager (AM)
- (3) Audit Data Communication System (ADCS)
- (4) Audit Management Information Base (Audit MIB)

Figure 1 shows the overall DAS architecture. These components are described in the following sections.

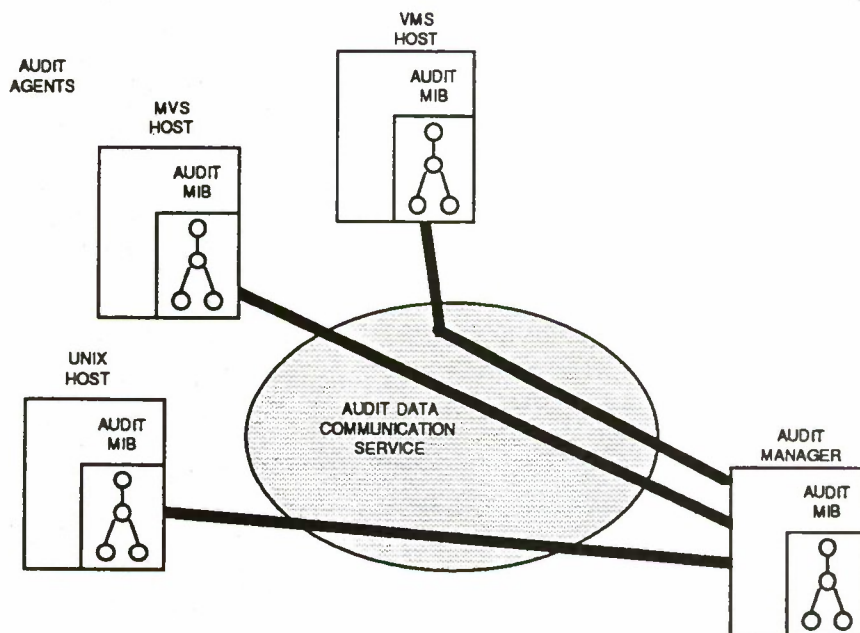


Figure 1. DAS Architecture

Audit Agent

The Audit Agent (AA) application resides on all host systems being monitored in a network. The AA application provides the ability to retrieve collected audit data and transfer this audit data to the Audit Manager (AM).

At the AA it is assumed that audit data is collected from operating system and network interfaces and stored in some local form. The audit data storage is usually in the form of an audit trail but may also encompass administrative and special purpose local logs. The information maintained in the audit trail and local logs may actually consist of more information than is used by the managed objects. It is the responsibility of the AA application to retrieve the necessary attributes about a managed object from the stored data.

Most operating systems provide utilities to record accounting and some security-related information. The ultimate desire for the DAS would be to use systems that maintain C2 audit logs. However, this may be currently unrealistic though the Government has a goal of making all of their systems C2 compliant in the future.

For the prototype DAS the UNIX systems to be used do not have C2 audit logs. The only log maintained by the UNIX System V operating system is for accounting purposes. Security-related data will be obtained utilizing several UNIX utilities and from the accounting log.

For the prototype, the /var/adm/wtmp file is used to obtain information concerning user login and logout. Changes in a user's effective ID is obtained from the /var/adm/messages which logs an entry every time a user does a "set user", "su", command. Information concerning the process a user is running is obtained from the /var/adm/pacct file.

To save overhead, the DAS prototype does not actually formulate a "security log." Instead user information is obtained dynamically from the available administrative logs, /var/adm/wtmp, /var/adm/messages, and /var/adm/pacct, and stored in temporary structures for response to user requests.

Once audit data is retrieved the AA is responsible for translating the local representation of the audit data (e.g., "C" structures, audit records) into an external form (i.e., managed object representation). As stated, earlier, managed objects are in ASN.1 definition that is understood by both the AAs and AMs.

In addition, the AA is capable of responding to AM commands to obtain additional audit information or modify a threshold that would trigger event reporting. The AA application is capable of responding to commands issued by the AM. The command set used is defined by the CMIS services provided under the ADCS.

Though in most cases the AM will request information about a managed object, it is desirable for information on some events (e.g., user login) to be sent asynchronously to the AM. This capability is called event reporting. In the prototype, AAs are capable of sending event reports on a defined set of events.

Audit Manager

The Audit Manager (AM) application resides on a centralized system that would act as a controller for the network. The AM is responsible for collecting and controlling audit data residing on all AAs under its jurisdiction. The audit data is subjected to an analysis technique which could be performed by a security officer, an automated audit analysis tool or, most likely, a combination of the two. A large network may be divided into subnetworks with an AM for each subnetwork.

The AM is responsible for receiving audit managed object definitions from the AA and presenting them for analysis. The analysis process may be either an automated analysis technique (i.e., an Intrusion Detection System (IDS)) or a security officer.

Audit data presented to the security officer will be via a Graphical User Interface (GUI). The audit data presented to the security officer may be raw data coming directly from the AA if an IDS is not used or it may be the results of the IDS after it has analyzed the AA's audit data.

The AM will be capable of controlling AA audit data. Control is necessary to increase granularity of auditing based upon results of analysis (e.g., obtain additional information about a suspicious user).

Issuance of control commands may be triggered either automatically by an audit analysis tool or by security officer interaction via the Security Officer Interface. The command structure to be used by the AM is provided by the ADCS as described in the next section.

Audit Data Communication Service

The Audit Data Communication Service (ADCS) provides the communication services necessary to transport messages between the Audit Agent (AA) and the Audit Manager (AM). The ADCS provides the ability to monitor and control network resources by transmitting command sequences between the AA and AM.

The communication component of the DAS will be provided by the International Standard (IS) Common Information Management Services and Protocol (CMIS/CMIP) [2]. Transfer of audit data and control commands will be accomplished using CMOT (Common Management Information Protocol Over TCP/IP) [4]. The objective of CMOT is to map the OSI management protocol architecture into the TCP/IP environment. CMOT follows the OSI model at the application layer, while using Internet protocols at the transport layer.

There are four primary commands provided by the CMIS services that the AM can use to control auditing at an AA. These commands are:

1. CREATE
2. DELETE
3. GET
4. SET

The AM uses Create and Delete commands to affect instantiations of managed object classes on the AA. Both of these services can only be invoked in a confirmed mode in which case a reply is expected. The AM would issue a Create command to cause an instantiation of a managed object class which would allow information to be obtained about a managed object's attributes. The AM would issue a Delete command to cause an instantiation of a managed object class to be removed when no further information concerning the managed object is needed at the time. Since these commands do not affect the definition of the managed object in the Audit MIB, deleted instantiations of managed objects can be recreated when needed.

The AM would issue a Get command when additional information about a managed object's attributes is needed.

The AM would issue a Set command when it is necessary to associate a threshold with a managed object.

In addition to the above commands, the CMIS services provides the AM the ability to receive Event Reports from the AA which are used to describe an event about a managed object. These reports will be sent asynchronously (i.e., when some threshold has been reached).

Audit Management Information Base

The Management Information Base (Audit MIB) is a "conceptual repository of management information." It is an abstract view of all the objects in the network that can be managed. The Audit MIB is conceptual in that it does not carry any implications about the physical storage (main memory, files, databases, etc.) of management information. Therefore, the Audit MIB can consist of the administrative log, audit log, system files, and information gathered from the network and operating system interfaces.

For the initial prototype, two Intrusion Detection Systems [5,6] were reviewed to determine a common set of audited events that would be used to define a basic Audit MIB. Since both of the intrusion detection systems used were UNIX-based and the prototype DAS is UNIX-based, the classes for the Audit MIB are given in familiar UNIX terminology. The information defined in the Audit MIB can be obtained from other types of operating systems and the terminology will be modified to be more generic for future versions of the DAS.

The basic Audit MIB is shown in Figure 2. As can be seen in this figure, there are four classes defined under the audit class, i.e., the "Audit Control Class", the "Audit Process Table Class", the "Audit Record Table Class" and the "Suspicious Users Class".

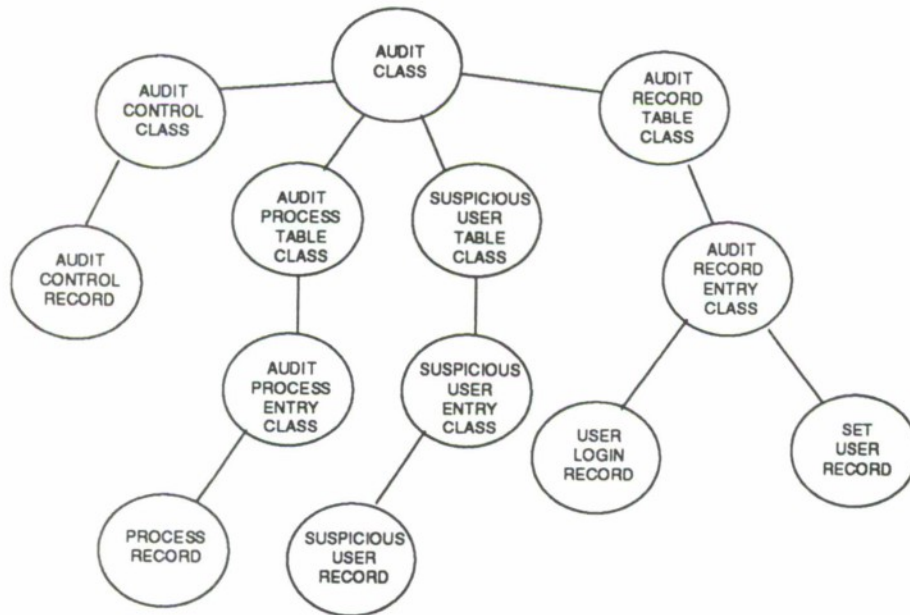


Figure 2. Audit MIB

The "Audit Control Class" manages the asynchronous Event Reporting capability. This class specifies what events or User IDs (UIDs) will trigger an event report. The current mode for the prototype is for Event Reports to be sent on all events and UIDs.

The "Audit Process Table Class" has the subclass "Audit Process Entry Class" defined which manages process information for individual users in the "process" record. The "Audit Process Table Class" manages a table of all "process" records. The "process" record defines attributes based on information obtained from the `/var/adm/pacct` file. This information is as follows:

- User ID (UID)
- Process start time
- TTY name
- Process ID (PID)
- Process name

The "Audit Record Table Class" has the subclass "Audit Record Entry Class" defined which manages individual "user login" and "set user" records. The "Audit Record Table Class" manages a table of all "user login" and "set user" records.

The "user login" and "set user" records define attributes based on information obtained from the files /var/adm/wtmp and /var/adm/messages, respectively. This information is as follows:

- User ID (UID)
- Start/end time
- TTY name
- Host address

The record contains an additional "type" attribute which defines information specific to the particular record. For the "user login" record this attribute states whether the operation was a user "login" or "logout". For the "set user" record this attribute states whether the "set user" operation was "successful" or "unsuccessful".

The "Suspicious Users Class" is used to specify those users that should be monitored more closely. By defining this class the user is able to create records on suspicious users which by specifying "selectedUids" in the "Audit Control Class" would enable the user to receive Event Reports on only this group of users. On a large system where many Event Reports may come in every minute as users log on/off the system this granularity allows a user of the DAS to isolate the activity of a group of users.

SECURING THE DISTRIBUTED AUDIT CAPABILITY

In order for the distributed audit capability to be effective, it must be possible to ensure the security of the audit data being analyzed. Since audit data is intrinsically sensitive, it must be possible to ensure that it cannot be read as it crosses the network. In addition, since audit managing procedures must initialize audit recording and request retrieval of sensitive data, it must be possible to verify that only validated procedures are making the requests. Also, the data being sent to an Audit Manager, either in response to a request or asynchronously, must be validated in order to develop reliable audit reports.

Several methods are being considered for providing the necessary security for the DAS. One method is to use the services to be provided by the Government Network Manage Profile (GNMP). The GNMP services are: authentication, access control, data confidentiality, data integrity, and non-repudiation of messages. Standards are currently being developed for these services with authentication and access control being given the highest priority.

A second method under consideration is to use the set of security protocols defined under an effort led by the National Security Agency (NSA) which could be used in conjunction with the ISO protocol suite. These security protocols would be conceptually "inserted" between various ISO protocol layers and would provide the security functions deemed to be required at certain layers in order to create a secure ISO stack. The SP4 security protocol would be inserted between the Network Layer (layer 3) and the Transport Layer (layer 4) [7]. The SP3 security protocol would be inserted between the Link Layer (layer 2) and the Network Layer (layer 3) [8]. One of these protocols would be incorporated into the current CMIP stack.

A third alternative being considered is to utilize CMIP as an application protocol and substitute an alternative transport protocol such as the Versatile Message Transport Protocol (VMTP) [9]. The ISO Development Environment (ISODE) currently supports CMIP over TCP/IP, which allows exploration of other transport services. VMTP uses a Public Key Encryption scheme and provides message authentication through a two step process.

Security was not included in the initial prototype of the DAS. These methods and others will be further researched and one method, or a combination of methods, will be incorporated in a future version of the DAS.

PROTOTYPE OPERATION

In 1992, a prototype DAS was developed which served as a proof-of-concept that the network management protocols could be used to provide a standard definition of audited events and to control audit data on remote hosts in a distributed environment from a central system. The DAS prototype provides a distributed auditing capability

between two Audit Agents and one Audit Manager. Both the Agents and the Manager are UNIX systems for the prototype.

From the Audit Manager the user is able to receive and control audit data at either Audit Agent via a user-friendly Graphical User Interface (GUI). The GUI uses XWINDOWS for a windows display. The prototype provides several windows for the user as shown in Figure 3. An "Agent Management" window will show which Audit Agent is currently being managed. A "Text Input" window allows the user to issue textual commands to the Audit Agent currently under management. Results to requests made by the DAS user are displayed in the "Text Input" window. One window for each agent is provided to display asynchronous Event Reports from the respective Audit Agent.

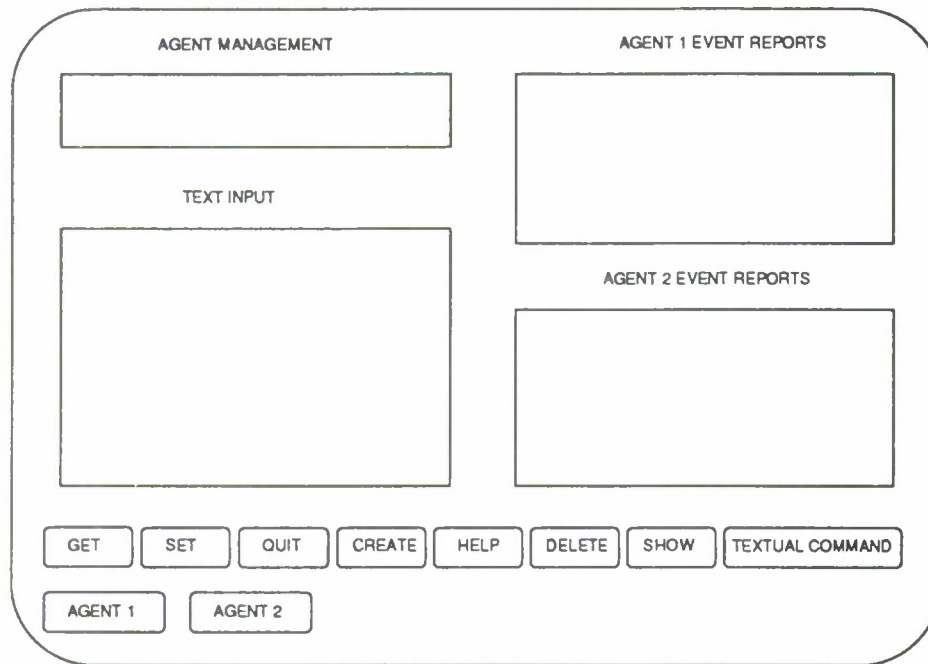


Figure 3. Graphical User Interface for the DAS Prototype

The commands provided in the prototype can be divided into two categories: (1) control commands issued to the Audit Agent, and (2) commands that assist in the operation of the prototype. All commands are provided via mouse buttons. The control commands consist of:

GET	obtain information about a managed object class
SET	change the value of an attribute
CREATE	create an instantiation of an object class
DELETE	delete an instantiation of an object class

The commands used to assist in the operation of the prototype include:

SHOW	display information about a managed object
HELP	display information about command syntax
QUIT	exit the prototype

The security officer will use the above commands to obtain information about the managed objects defined in the baseline standard Audit MIB described earlier in this paper.

SUMMARY

To date there exists a few other projects that are attempting to address security auditing in a heterogeneous distributed network. DAS is unique in that it provides the solution by: (1) defining a standard definition of audited events, and (2) using existing network management protocols for the transfer and control of audit data. Further enhancement of the initial DAS prototype is currently planned.

The prototype DAS was incorporated into SPARTA's LAN. In most cases the audit data used by the DAS was obtained either dynamically or from system files normally maintained on the systems. One exception was in the collecting of accounting information. An alternative method for collecting information derived from the accounting utility is being explored since many UNIX systems do not use this utility due to the additional overhead it causes. This change would not affect the overall DAS design, only the local implementation on UNIX systems. Additional analysis of the effect on system performance will be performed during continued DAS development.

Future modifications will also include the addition of different types of audit agents, including VMS and SUN OS C2 audit agents. The addition of a SUN OS C2 audit agent would be in line with the needs of several well-known intrusion detection systems.

The initial prototype presents the audit data to the security officer for analysis. Enhancement of the prototype would include integration with an existing intrusion detection system. The Distributed Intrusion Detection System (DIDS), being developed for the Air Force by the Lawrence Livermore National Laboratory, Haystack Laboratories, and UC Davis, is a likely candidate for integration.

REFERENCES

1. Banning, Debra L., "A Distributed Audit System (DAS) Using Network Management Protocols," Masters Thesis, California State University--Long Beach, December 1992.
2. Information Technology--Open Systems Interconnection--Common Management Information Protocol--Part 1: Specification for CCITT Applications, November 24, 1990.
3. Specification of Abstract Syntax Notation 1 (ASN.1)--IS-8824, International Organization for Standardization, 1987.
4. Internet RFC 1155, The Common Management Information Services and Protocol over TCP/IP (CMOT), April 1989.
5. T. Lunt, A. Tamaru, F. Gilham, R. Jagannathan, C. Jallai, H. Javitz, P. Neumann, A. Valdes and T. Garvey, "A Real-Time Intrusion Detection Expert System (IDES), Final Technical Report, February 28, 1992.
6. S. Smaha, "Distributed Intrusion Detection System (DIDS)--Preliminary Design," December 1990.
7. SDNS, Secure Data Network Systems, Security Protocol 4, SDNS Protocol and Signaling Working Group SP4 Sub-Group, SDN.401, 1988.
8. SDNS, Secure Data Network Systems, Security Protocol 3, SDNS Protocol and Signaling Working Group SP3 Sub-Group, SDN.301, 1988.
9. D. Cheriton, VMTP: Versatile Message Transaction Protocol, Preliminary Version, January 1988.

TANDEM THREAT SCENARIOS: A RISK ASSESSMENT APPROACH

Lisa M. Jaworski
Trusted Information Systems, Inc.
3060 Washington Road (Rt. 97)
Glenwood, MD 21738

ABSTRACT

In this paper, I will describe the approach taken in a design-level risk assessment for a communications system, focusing on the use of tandem threat scenarios. This methodology has been cited by the National Security Agency as a good example of a comprehensive risk assessment in the *Risk Expectation Management* document [5] of the ISS Engineering document series.

Tandem threat scenarios take the traditional risk assessment methodology one step further by analyzing the perpetration of successions of threats at multiple vulnerability points. This paper will also discuss the real world issues involved in developing a useful and accurate design-level risk assessment for a highly complex system.

Keywords are: risk assessment, threat, vulnerability, vulnerability point, tandem threat scenario, residual risk, and countermeasure.

METHODOLOGY

The risk assessment methodology, which consists of the following steps, is illustrated in Figure 1. Each step is amplified in the following paragraphs.

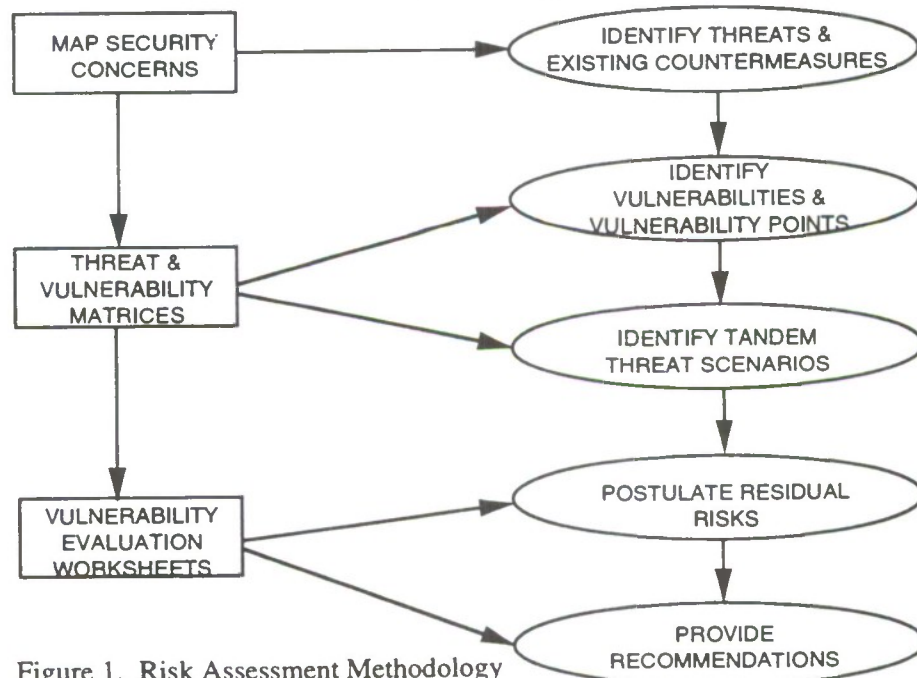


Figure 1. Risk Assessment Methodology

Copyright 1993 Lisa M. Jaworski.

- a. Identify threats.
- b. Identify and analyze the effectiveness of existing countermeasures.
- c. Hypothesize vulnerabilities and identify vulnerability points.
- d. Identify and assess tandem threat scenarios.
- e. Postulate residual risk.
- f. Correlate the risk assessment to the system's security policy and security requirements.
- g. Provide recommendations for reducing residual risks to a level acceptable to the Designated Approving Authority (DAA).

Identify Threats

A threat is any circumstance or event with the potential to cause harm to the system or activity in the form of destruction, disclosure, and modification of data, or denial of service. A threat is a potential for harm. The presence of a threat does not mean that it will necessarily cause actual harm. Threats exist because of the very existence of the system or activity and not because of any specific weakness. Although the threats to a system are identified based on the system's mission and the characteristics of its architecture, generally, threats can be categorized in terms of the system's functions as follows:

- a. Information processing
- b. Information transfer
- c. Emanations
- d. Administrative procedures
- e. Personnel
- f. Physical areas
- g. Operations (mission).

Network security considerations can be considered a combination of information processing and information transfer, or established as a separate category. For each of these categories, security concerns may be represented graphically. These security concern mappings form the basis of the hypothetical vulnerabilities postulated for the system. These mappings are not strictly limited to threats. They serve to identify capabilities within the system for which vulnerabilities must be hypothesized and subsequently analyzed. A sample map is presented in Figure 2.

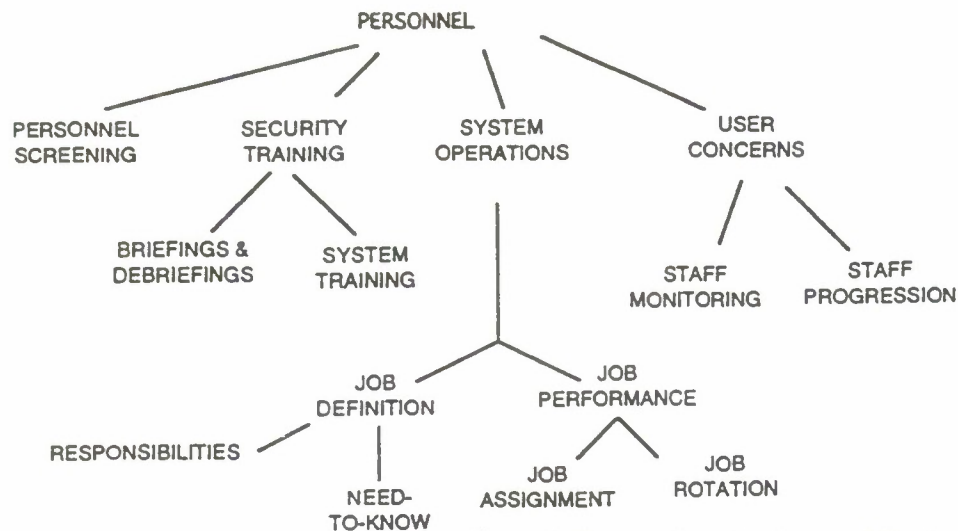


Figure 2. Sample Security Concern Mapping

Assess Existing Countermeasures

A countermeasure is any action, device, procedure, technique, or other measure that reduces the vulnerability of a system or activity to that threat. These countermeasures are categorized in a one-to-one fashion to system threats as follows:

- a. Computer security (COMPUSEC)
- b. Communications security (COMSEC)
- c. TEMPEST
- d. Administrative security
- e. Personnel security
- f. Physical security
- g. Operations security (OPSEC).

Hypothesize Vulnerabilities and Identify Vulnerability Points

A vulnerability is a weakness in the physical layout, organization, procedures, personnel, management, administration, hardware, firmware, or software that may be exploited to cause harm to the system or activity. The presence of a vulnerability does not in itself cause harm. A vulnerability is merely a condition or set of conditions that may allow the system or activity to be harmed by an attack. A vulnerability point is the system component wherein a vulnerability exists. A vulnerability point could be hardware, firmware, software, an administrative or personnel procedure, or a physical area pertinent to the system.

System vulnerabilities are hypothesized for each threat category. The easiest way to do this is to develop a threat/vulnerability matrix for each threat category. Threats are depicted in the rows and hypothesized vulnerabilities in the columns. Individual cells represent hypothesized vulnerabilities which, if left uncountered, could be used to actualize the corresponding threat. Note that several threats may be associated with a particular vulnerability and more than one vulnerability may be associated with a threat. A sample threat/vulnerability matrix is

provided in Table 1. Table 1. Sample Threat/Vulnerability Matrix - Personnel

VULNERABILITIES THREATS	INADEQUATE SCREENING OF PERSONNEL	INADEQUATE POLICY FOR ISSUING ACCESS	INADEQUATE SECURITY BRIEFINGS
DAMAGE	X	X	X
SUBSTITUTION	X	X	X
THEFT	X	X	X
BROWSING	X	X	X

Identifying the set of vulnerability points is easily accomplished by analyzing the system hardware/firmware and software specifications and the system architecture. Hardware, firmware, and software components comprise the minimum set of vulnerability points; however, administrative and personnel procedures, etc., may also constitute vulnerability points. Once the set is defined, it must be applied consistently (i.e., the same set must be applied when analyzing vulnerability points against the hypothesized vulnerabilities).

A vulnerability/vulnerability point matrix is used to hypothesize where a particular vulnerability could exist in the system. A sample matrix is shown in Table 2. Vulnerabilities are depicted in the rows and vulnerability points in the columns. Individual cells represent areas within the system where a hypothesized vulnerability could be actualized. Note that there must be a one-to-one correlation between the vulnerabilities identified via the threat/vulnerability matrices and those presented in the vulnerability/vulnerability point matrices. As with the threat/vulnerability matrix, vulnerabilities may be associated with more than one vulnerability point and vice versa.

Table 2. Sample Vulnerability/Vulnerability Point Matrix - Personnel

VULNERABILITY POINTS VULNERABILITIES	HARDWARE	FIRMWARE	SOFTWARE
INADEQUATE SCREENING OF PERSONNEL	X	X	X
INADEQUATE POLICY FOR ISSUING ACCESS	X	X	X
INADEQUATE SECURITY BRIEFINGS	X	X	X

Vulnerability evaluation worksheets, which are a modified version of the

worksheets specified in OPNAVINST 5239.1A [6], are also used in this step. The front of the worksheet is illustrated in Figure 3. The information contained in the worksheet is presented as follows:

VULNERABILITY EVALUATION WORKSHEET		1. WORKSHEET #
2. PROGRAM COMPONENT		
3. THREAT		
4. HYPOTHEZED VULNERABILITY		
5. DESCRIPTION AND JUSTIFICATION, BASED ON EXISTING COUNTERMEASURES		
6. IMPACT AREA		
<input type="checkbox"/> MODIFICATION	<input type="checkbox"/> DESTRUCTION	<input type="checkbox"/> DISCLOSURE <input type="checkbox"/> DENIAL OF SERVICE

Figure 3. Vulnerability Evaluation Worksheet

- a. Worksheet Number. Box 1 contains a unique worksheet identification number that is used to correlate the risk assessment to the system's security policy and security requirements.
- b. Program Component. Box 2 identifies the particular program component (e.g., subsystem) with which the vulnerability is associated.
- c. Threat Name. Box 3 identifies the particular threat with which the vulnerability is associated.
- d. Possible Vulnerability. Box 4 identifies the hypothesized vulnerability.
- e. Description and Justification. Box 5 contains a description of the vulnerability, the existing countermeasures identified, and an evaluation of the risk resulting from an attempt to exploit the vulnerability.
- f. Impact Areas. Box 6 identifies the areas of impact (risks), should the vulnerability be successfully exploited.

These worksheets are used to evaluate the threat/vulnerability pairs identified via the threat/vulnerability matrices in view of all available system information. In a design-level risk assessment, such information consists of system specifications, the Software Development Plan, the Configuration Management Plan, etc. Note that a worksheet must be developed for each threat/vulnerability pair identified in the matrices.

The vulnerability/vulnerability point matrices may be correlated with information in the Description and Justification box of the vulnerability evaluation worksheets. References (page and paragraph numbers) to system documents should be included when discussing existing countermeasures. These references can be used to derive the set of actual system vulnerability points.

Postulate Residual Risk

Residual risk is that portion of risk which remains after security measures have been applied; that is, residual risk is calculated for each threat/vulnerability pair based on the system's existing countermeasures. According to OPNAVINST 5239.1A [6] and AR 380-19 [2], the risks associated with any system are unauthorized disclosure, modification, and destruction of system components and information and denial of system services. The degree of risk associated with threat/vulnerability pairs is the likelihood of actualization of a threat via the exploitation of a particular vulnerability. Degree of risk, documented in Box 5 of the worksheets, can be divided into three categories:

- a. Low. The threat scenario is considered to be very unlikely to occur, to have a low impact if it does occur, or to be completely controlled by existing countermeasures.
- b. Medium. The threat scenario is considered to have a moderate likelihood of occurrence, to have a moderate impact if it does occur, or to be partially controlled by existing countermeasures.
- c. High. The threat scenario is considered to have a high likelihood of occurrence, a significant impact if it does occur, or not to be controlled by existing countermeasures.

This definition of risk was derived from DoD Directive 5200.28, *Security Requirements for Automated Information Systems (AISs)* [4]. Many risk assessments define risk only in terms of the probability of actualization. Although it may be a bit more complicated to determine residual risk in this fashion, it is a more complete and useful measurement since the the resulting impact of actualization is considered, as well as the probability of occurrence. Thus, if an event was not considered to be very likely to occur but would, in fact, have a very high impact, its residual risk would be high and so, it is very likely that such an issue would "fall through the cracks" or be deliberately ignored is low.

Correlate to Security Policy and Security Requirements

This step is necessary to ensure that the design is consistent with the system's security policy and that all security requirements have been implemented. A high-level correlation of each hypothesized vulnerability may be documented on the back of the vulnerability evaluation worksheet, as follows:

- a. Confidentiality
- b. Access control
- c. Accountability
- d. Integrity
- e. Availability.

Identify and Assess Tandem Threat Scenarios

A tandem threat scenario examines the perpetration of successions of threats (identified in Step 1 of the methodology) at multiple vulnerability points that could result in unauthorized modification, disclosure, and destruction of system components and information and denial of service conditions. A sample tandem threat scenario is presented in Figure 4. As shown, if one chooses to include threat agents, these can be depicted.

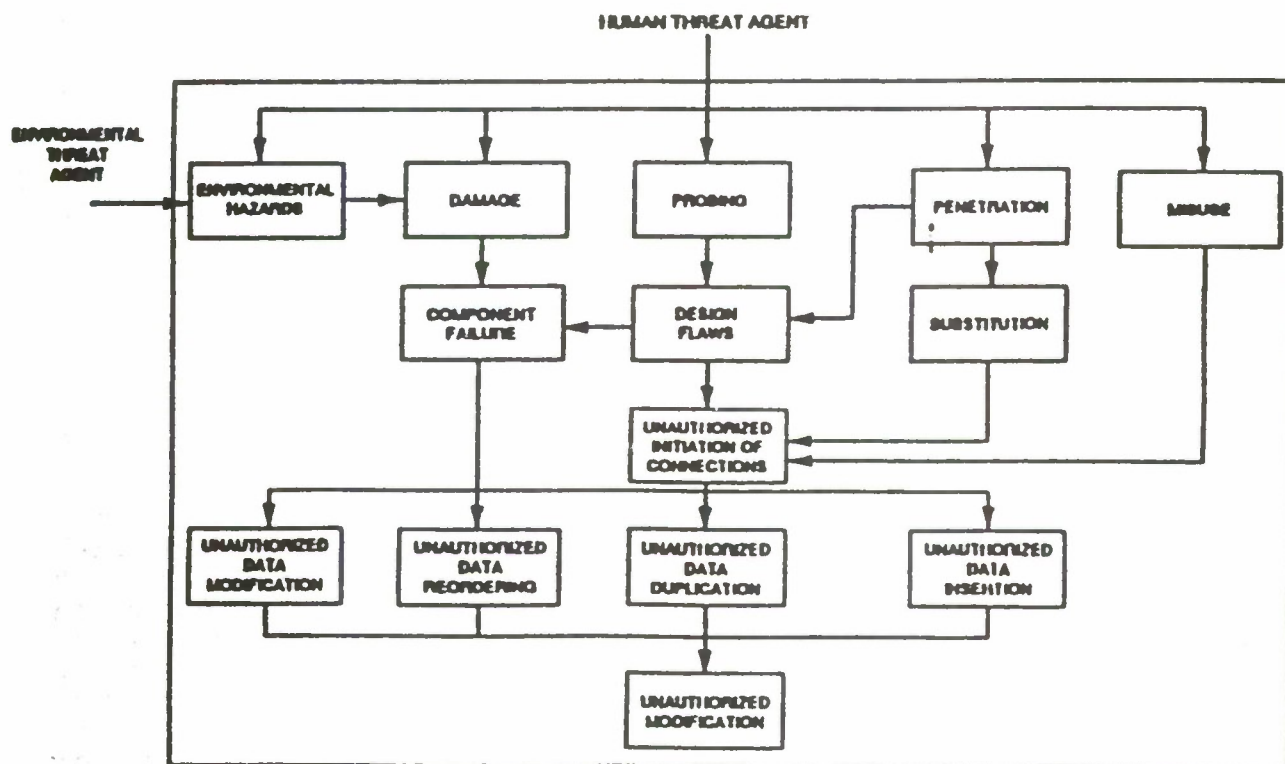


Figure 4. Sample Tandem Threat Scenario - Unauthorized Modification

Several steps must be completed during the analysis of the tandem threat scenarios. First, for each of the threat categories, tables must be generated which depict the associated vulnerabilities, their impact areas, and their evaluated risk. A sample is presented in Table 3.

Table 3. Sample Vulnerability/Risk Matrix

VULNERABILITIES	IMPACT AREA				EVALUATED RISK
	MODIFICATION	DESTRUCTION	DISCLOSURE	DENIAL OF SERVICE	
INADEQUATE SCREENING OF PERSONNEL	X	X	X	X	HIGH
INADEQUATE POLICY FOR ISSUING ACCESS	X	X	X	X	MEDIUM
INADEQUATE SECURITY BRIEFINGS	X	X	X	X	LOW

Next, attacks to the system are hypothesized and studied, in relationship to threats and the system's vulnerabilities. The tandem threat scenarios are used to guide the order of the attacks. Generally, attacks consist of gaining unauthorized access to a system component and misusing that component [1]. Associated with each impact area are numerous methods of attack (i.e., threats). Any attack on a system resulting in unrestricted access will generally permit the actualization of other threats [1].

Finally, results of the analysis of the tandem threat scenarios are documented.

Provide Recommendations

Countermeasures must be recommended to reduce residual risks to a level acceptable to the DAA. Also, if a gap exists between the system's security policy or security requirements and the design of the system, it must be closed. No further action need be taken for those vulnerabilities sufficiently countered to ensure minimal or non-existent risk to the security of the system. Recommendations should be made on the basis of an informal cost-benefit analysis. Thus, a countermeasure that counters several vulnerabilities or that costs less than an equally effective alternative, should be recommended.

Recommendations are categorized in the same manner as existing countermeasures, thus, providing consistency within the risk assessment document.

BENEFITS DERIVED FROM METHODOLOGY

The most important benefit derived from this methodology is that the use of tandem threat scenarios presents a more comprehensive examination of system threats than a more standard risk assessment approach. While all

threats to a system are assessed via the standard approach, they are not assessed in succession nor applied at multiple vulnerability points. The tandem threat scenarios enable system threats to be prioritized according to the number of threads connecting to it. Obviously, the threat with the most threads connected to it could prove to be the most catastrophic and so, should be assessed most intensively.

In addition, categorization of threats and their corresponding hypothesized vulnerabilities enables early identification of problems whose solution or countermeasure requires a long lead time. For example, some personnel security issues such as obtaining appropriate clearances or billets requires a one year lead time. Product procurements can also require an extended amount of time, especially if a certain contingency such as Desert Storm erupts. During Desert Storm, several types of computer products required for a system integration effort were not available, since the Pentagon had ordered all available units shipped directly overseas.

Although not directly attributable to the use of tandem threat scenarios, there are a number of secondary benefits derived from this methodology. First, a concise presentation of each step of the assessment of the security of the system can be presented to the DAA. Many security engineers feel that risk assessment is an art rather than a science. Although it is, to an extent, it is important to be able to present the risk assessment methodology and findings clearly to the certification authority and, subsequently, to an accreditor. The use of matrices and tables ensures consistency throughout the document.

Second, the methodology did not become bottlenecked in trying to derive dollar figures for financially abstract system components (e.g., messages). The compilation of qualitative risk probabilities was preferred by project officials who reviewed the document, most notably the DAA.

Finally, the risk assessment document was structured so that the summary data (e.g., threat/vulnerability matrix) was presented in the front of the document and modularized according to system component. This approach is most useful when assessing the risks associated with a network, especially if viewed as a single entity (as opposed to the interconnected accredited AIS view).

LIMITATIONS OF METHODOLOGY

As with any new approach or methodology, there will be some time spent on the learning curve (i.e., the current state of the art may sometimes need to be extended to incorporate new ideas). For instance, at the time of this writing, there is no automated risk assessment tool known to the author that could be used to apply this methodology. Thus, the process is very time intensive (four to five technical man-months of analysis for a large system plus intensive graphics and word processing support).

In addition, due to time and budget constraints, residual risks for tandem threat scenarios was not correlated with the residual risks of individual threat/vulnerability pairs. Research into how such

probabilities could be derived reliably is encouraged.

CONCLUSIONS

The use of this risk assessment methodology clarifies and enhances the standard risk assessment approach. Tandem threat scenarios provide many benefits which simply cannot be derived from the standard approach. In addition, the secondary benefits of this approach, specifically, document layout, may be obtained even if tandem threat scenarios are not utilized. The document layout described herein is very useful when assessing the risks associated with a network, especially if viewed as a single entity (as opposed to the interconnected accredited AIS view). As with any risk assessment, tandem threat scenarios must be viewed as a management tool. The risk assessment only identifies problem areas (vulnerabilities) and possible corrective actions (countermeasures). Risks are not actually reduced unless management implements the recommended countermeasures.

ACKNOWLEDGEMENTS

The author wishes to thank Richard Hale, David Mihelcic, and Michael Harrison for their support in the development of this risk assessment methodology.

REFERENCES

- [1] AFR 205-16, Computer Security Policy, For Official Use Only, Headquarters, Department of the Air Force, Washington, DC, 28 April 1989.
- [2] AR 380-19, Information Systems Security, Headquarters, Department of the Army, Washington, DC, 1 August 1990.
- [3] DoD 5200.28-STD, Department of Defense Trusted Computer System Evaluation Criteria, National Computer Security Center, December 1985.
- [4] DoD Directive 5200.28, Security Requirements for Automated Information Systems (AISs), Department of Defense, 21 March 1988.
- [5] Information Systems Security Engineering: Part 1 Risk Expectation Management, S9 Technical Report 1-92, National Security Agency, Ft. George G. Meade, MD, 3 September 1992.
- [6] OPNAVINST 5239.1A, Department of the Navy Automated Data Processing Security Program, Department of the Navy, April 1985.
- [7] SECNAVINST 5239.2, Department of the Navy Automated Information Systems (AIS) Security Program, Department of the Navy, 15 November 1989.

TOWARD A COMPREHENSIVE INFOSEC CERTIFICATION METHODOLOGY

Charles N. Payne, Judith N. Froscher and Carl E. Landwehr
Center for High Assurance Computing Systems
Naval Research Laboratory
Washington, D.C. 20375-5337

Abstract

Accreditors want to know what vulnerabilities will exist if they decide to turn on a *system*. TCSEC evaluations address *products*, not systems. Not only the hardware and software of a system are of concern; the accreditor needs to view these components in relation to the environment in which they operate and in relation to the system's mission and the threats to it. This paper proposes an informal but comprehensive certification approach that can provide the accreditor with the necessary information. First, we discuss the identification of *assumptions* and *assertions* that reflect system INFOSEC requirements. Second, we propose the definition of an *assurance strategy* to integrate security engineering and system engineering. The assurance strategy initially documents the set of assumptions and assertions derived from the requirements. It is elaborated and refined throughout the development, yielding the *assurance argument*, delivered with the system, which provides the primary technical basis for the certification decision. With the assurance strategy in place, certification of the trusted system can become an audit of the development process.

Keywords: Certification, Trusted Systems, INFOSEC, Software Engineering

INTRODUCTION

Computer security certification is the assessment that the computer hardware and software is trustworthy.¹ It supports the accreditation decision to allow the computer to

process classified information in an operational environment.

Trusted product evaluation is the computer security certification of the product against the criteria of the *Trusted Computer System Evaluation Criteria* (TCSEC) [1]. Trusted system certification², on the other hand, comprises several technical and procedural certifications, including a technical computer security certification. The outcome of the trusted system certification influences the criteria for other certifications, such as administrative security and TEMPEST requirements. If the protection features of the system are deficient in any way, other protection measures must be used to protect the information maintained by the system.

While the security feature requirements of the TCSEC are targeted primarily at "information processing systems employing general-purpose operating systems that are distinct from the application programs being supported" (i.e., trusted products), the assurance requirements extend "to the full range of computing environments"[1, p. 2].

According to the TCSEC, both trusted products can be evaluated and trusted systems can be certified against its criteria. However, the evaluation approach that is used for trusted products does not satisfy the needs of an accreditor for a trusted system. In particular, the approach does not identify the risks of using the system in its operational environment. Accreditors want to know what vulnerabilities will exist if they turn on the system. A better approach is needed for certifying trusted systems. This paper proposes an informal but comprehensive approach that can be used by project managers, designers, and implementors of a system and can provide the accreditor with the risks of using the system.

We want to clarify our use of the terms *trusted product* and *trusted system*. We have adopted the definitions of

¹The computer is *trusted* if we rely on it for security enforcement. It is *trustworthy* if that reliance is justified technically. A computer may be trusted even though it is not trustworthy, because the Designated Approving Authority (DAA) may permit its use despite known weaknesses.

²In the TCSEC[1], this is called a *certification evaluation*.

product and *system* from the European community's *Information Technology Security Evaluation Criteria* (ITSEC) [2]. According to the ITSEC, a *system* is a specific installation "with a particular purpose and a known operational environment". A *product*, on the other hand, is "a hardware and/or software package that can be bought off the shelf and incorporated into a variety of systems". A product or system is *trusted* if we rely on it for some critical purpose, such as (in the present context) security enforcement.

The characteristics and requirements of a trusted system's end-users and the threats to a trusted system's security can be determined with some certainty. The security requirements that it must enforce derive from national security policy. If a trusted system is based on an evaluated product, the person deploying the trusted system must ensure that the assumptions of the product are valid for the operating environment. It may be necessary to develop additional trusted code to enforce environment-specific security requirements.

The computer security certification of a trusted system can be based on the same criteria as the evaluation of a trusted product. But the system's certification may require more resources than the product's evaluation, because the system's security requirements are based on the known operational environment and so are more comprehensive than those of the product.

In the following sections, we discuss why the evaluation approach for trusted products does not scale up for trusted systems. Then we identify our objectives for trusted system certification and describe how the development process must be improved to support these objectives:

- adopt the accreditor's perspective and identifying the system INFOSEC policy,
- base the development (and consequently the certification) on trade-offs between assumptions and assertions, and
- define an *assurance strategy* to motivate the development process.

The assurance strategy documents the assertions that must be true as the result of design decisions, and identifies the methods used to demonstrate the validity of each assertion for the system. Finally, we propose that with the assurance strategy in place, certification of the trusted system can become an audit of the development process.

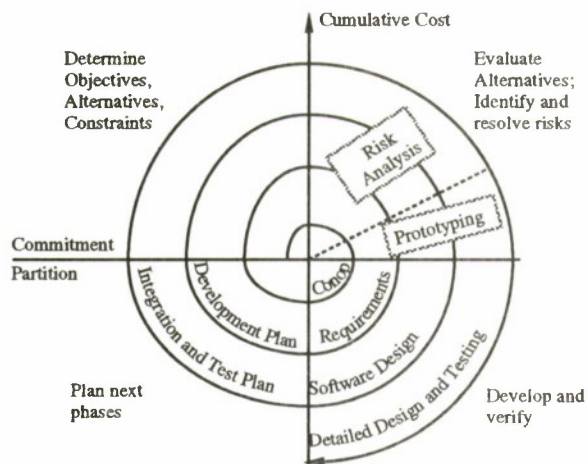


Figure 1: Boehm's spiral model

WHY THE PRODUCT EVALUATION APPROACH DOES NOT WORK FOR SYSTEMS

Boehm [3] defined a software process model that views the software process as an iteration through four phases: planning, risk identification, risk resolution and development. In this model (see Fig. 1), a project begins its life in the center and follows a spiral trajectory outwards: distance from the origin represents cumulative project cost, and angular displacement corresponds to the current project phase. We will use this model to illustrate the two ways described previously ([4]) for how the product evaluation approach can be applied to systems. Later, we will use it to illustrate our proposed system certification approach.

The NCSC's product evaluation approach provides security engineering expertise to the developer during the product's construction (Vendor Assistance and Design Analysis Phases) and then assigns an evaluation team to assess the completed product and documentation (Formal Evaluation). If we apply this approach to a system, as illustrated in Figure 2, security engineering support would be provided throughout the spiral, but certification would begin only after system delivery. The certification team must be isolated from the security engineering support effort in order to preserve the independence of the certification.

Although this process is likely to deliver a system, its certification may be quite protracted, because disagree-

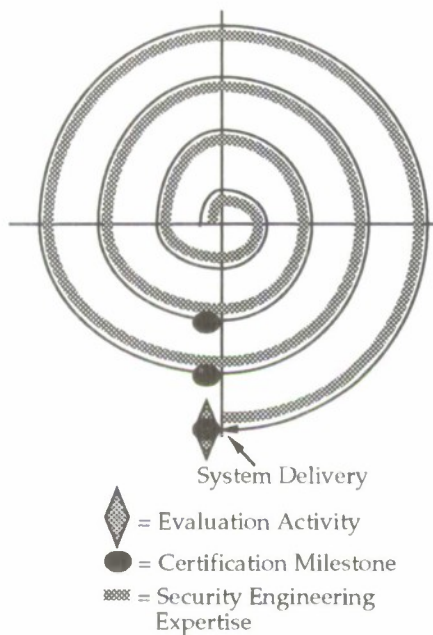


Figure 2: Certification after delivery

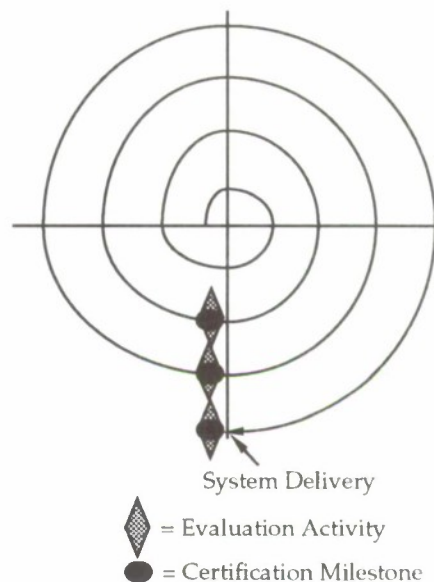


Figure 3: Milestone-based certification

ments between the security engineers and the certification team may require redevelopment of portions of the system. A product developer can accept this lengthy process, because the product can eventually be sold to many customers, but a system is often built for a single customer. The customer cannot tolerate lengthy delays between the delivery and operational use of the system, nor can the developer redevelop the system without further funding. The accreditor is left in the uncomfortable position of permitting the uncertified system to operate, accepting unevaluated risks, or denying the user a needed capability. For these reasons, delaying the start of system certification activities until after the system is completed seems impractical.

An alternative approach is illustrated in Figure 3. Here, security certification is structured as an independent verification and validation process that reviews the development at each contractual milestone. The security engineering support is eliminated, but the certification team's review of the deliverable provides feedback to the developer and the customer. If certification evidence provided at a particular milestone is inadequate, the developer must remedy the deficiencies before proceeding. Certification still does not occur until after the system is delivered, but the delay between system delivery and system certification should be much reduced from the previous approach. If the system is delivered, it should be certified

promptly.

While this approach eliminates some of the risks of the previous one, it introduces new ones. Since the TCSEC describe assurance evidence only for a completed system, the certification team is challenged to identify what certification evidence is needed at each milestone in order to assure that the resulting system can be certified – and this activity may recur each time a new system, TCSEC class, or milestone structure is addressed. The need to revise deliverables to meet certification requirements at each milestone may substantially increase system development time. Finally, the necessary interaction between the certification team and system developer can threaten the independence of the certifiers.

CERTIFICATION OBJECTIVES

We have identified problems with two certification approaches. These difficulties are related to how trusted systems are procured and developed. While results from the certification process should influence programmatic decisions, programmatic constraints alone should not define the assessment process. The ideal certification process should be applicable to any project, regardless of lifecycle model and programmatic milestone definition.

Certifiers should view the development as it progresses. Certification decisions should be made throughout the system lifecycle — not just when particular pieces

of assurance evidence have been completed. Also, certification decisions should be based on more than just the form of the assurance evidence. The certification team and the developer need a systematic way to reason about countermeasures, their effectiveness, and the design, development, and assurance argument of the protection mechanisms. In fact, if the developer made trade-off/design decisions based on the same concerns and in the same context as the certifier, the certification process could become an audit of the development process. The development process should produce assurance that countermeasures are effective and correct. The assurance argument should drive the development of a trusted system and should begin at system concept with threat identification. We have now identified objectives for the development process as well as the certification process.

STEPS TO A SOLUTION

Consider the Accreditor's View

We believe that to develop a better approach to system certification, we must consider the accreditor's perspective on the system. The accreditor's primary concern is protecting classified information with the most cost-effective controls available. Given a particular mission, the threats to the successful execution of the mission, and a system intended to help accomplish it, the accreditor must first understand the risks of operating the system and whether there are countermeasures outside the system for reducing those risks to an acceptable level. When all means for reducing risk have been considered, the residual risk must be weighed against the contribution of the system to its intended mission, and the decision to operate it or not must be taken.

The accreditor's view thus includes not only system hardware and software components but also the people who use the system and the administrative measures that regulate their use. Indeed, physical and personnel security measures often are instituted to compensate for uncertainties or weaknesses in automated systems; by bringing different security disciplines into a common framework, we hope to make it easier to formulate rational trade-offs among them and to clarify, if not quantify, the risks a particular set of choices presents when the system is operated.

Identify the System INFOSEC Policy

Past work in computer security has frequently identified "security policy" (as in "Formal Security Policy Model") with mandatory and discretionary access control. But in the context of systems rather than products, this focus is clearly too narrow.

Sterne [7] recognized this problem and identified three levels of security policy: security policy objectives, organizational security policy, and automated security policy. One of Sterne's goals was to separate policies applied to people from those applied to machines. Although we agree that people must know what's expected of them and what they can expect of their machines, we want to provide a framework that accommodates trade-offs between human and machine policies. None of the policies Sterne identified seems to correspond directly to the view of the accreditor, who often is concerned with the mission of the system as well as the policies Sterne identified.

For this purpose, we identify the *information security (INFOSEC) policy*. As illustrated in Figure 4, the INFOSEC policy is derived from the organizational security policy and from other constraints. The automated security policy, i.e., the trusted system's security requirements, is then derived from the INFOSEC policy and the operational requirements.

The trade-off analysis begins with the system requirements analysis phase of the lifecycle and continues informally throughout system development. The resulting decisions drive the development, and consequently the certification, of the trusted system. So that the developer and the certifier can understand these decisions, we need a framework in which to capture and document them. With such a framework, if a trade-off decision leads to an unworkable design, the developer can better assess other alternatives, and the developer, customer and certifier can determine the impact of any changes on the development and certification of the system.

Think in Terms of Assumptions and Assertions

The framework that we propose for documenting the trade-off decisions is based on identifying the assumptions and assertions that must be true for the system as a whole to be secure. The notion of identifying assumptions and assertions is not new; it derives from earlier work documented by Landwehr [5] and Froscher [6]. Its use to identify relationships among various security disciplines, however, has not been advocated previously, nor has it been suggested as a tool for documenting trade-

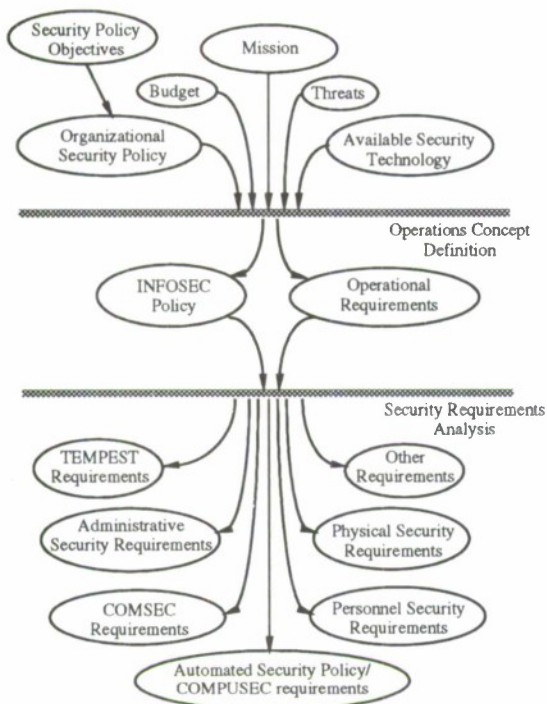


Figure 4: Introducing the INFOSEC Policy

off decisions taken during system development.

For a given system, *assertions* are predicates that are enforced by the system, and *assumptions* are predicates that are enforced in the system's environment. The system is unable to enforce the assumptions, but relies upon them. Together assumptions and assertions represent what must be true of the system in its environment to satisfy the security policy. If an assumption or an assertion is false, a security violation may occur.

An example assertion is

A user can only view on a workstation screen information for which (s)he is cleared.

This is a relatively high level assertion about system behavior, and it might lead to several other, lower level assertions concerning the access controls implemented by the operating system, user authentication, and so on. Taken together, the collection of lower level assertions should support the argument that the higher level assertion will be enforced.

But suppose that the system developer can only use an operating system that lacks effective access controls, or that users cannot be required to authenticate themselves? In this case, other ways of supporting this assertion must be found. For example, one might require that:

- All individuals able to view workstation screens are cleared to the SECRET level, and
- No sources that produce information at a level higher than SECRET are connected to the system.

From the standpoint of computer security, these last two predicates are assumptions, for they cannot be enforced by computer software and hardware. From the standpoint of physical and personnel security, however, these are assertions: measures within those security disciplines should be sufficient to enforce them.³

The purpose of stating the assertions and assumptions explicitly is to facilitate a systematic analysis that demonstrates that all stated security predicates are true for the system and its environment. The systematic analysis compares the system's computer security assumptions to the assertions of other security disciplines. This exercise continues for all defined security disciplines. If assumptions for any discipline are identified that do not correspond to assertions for some other entity, then these assumptions represent *vulnerabilities* in using the system. If the vulnerabilities result in a risk that is too great, the trade-off analysis is revisited. This analysis is illustrated for administrative security, a physical security officer and computer security in the simple example of Figure 5.

The trade-off continues throughout the design of a system that must enforce the COMPUSEC requirements. Assertions and assumptions are determined for individual hardware and software components, then for modules, etc. At each new level of specification, assertions are derived from the previous level. Assumptions are propagated from higher levels or they may become assertions for this level. In addition, new assumptions may be introduced.

The assumptions and assertions approach can represent clearly the effects of decisions made during the development cycle and can make explicit the risk of using the system. It also makes explicit the risk of interconnectivity and allows protection measures to be developed that promote secure interoperability. Because it captures the role each part of a system plays in the development of a countermeasure, this approach allows the maintainer and the accreditor to reason about the effects of any proposed change to the accredited system. The maintainer and the certifier can identify what parts of the system assurance argument must be reexamined to reaccredit the system as a result of any change. We also believe that

³ Naturally, this example is incomplete! For example, we have said nothing about the system's initial state. Establishing the completeness of a set of assumptions and assertions, relative to some set of security objectives, is not something we expect to be able to do algorithmically.

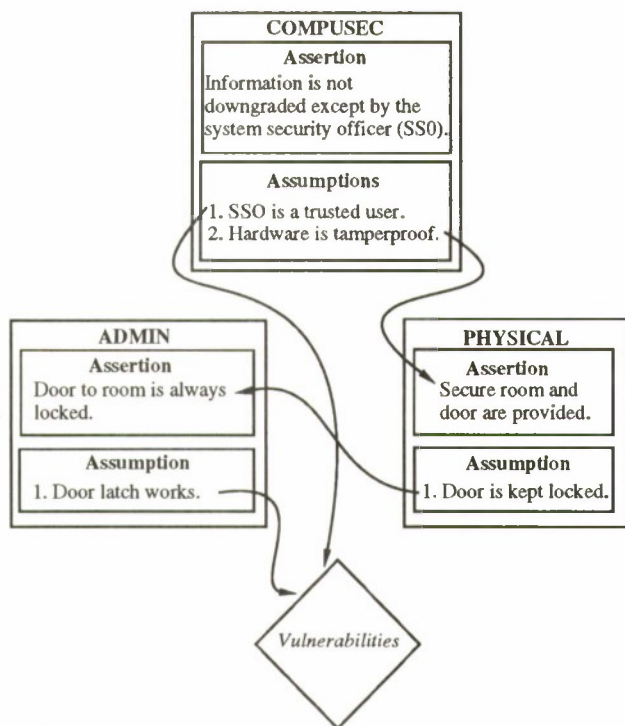


Figure 5: Using assumptions and assertions to find vulnerabilities

the assumptions and assertions approach can support trusted system technology's migration to the open systems goals for insertion of new technology into existing systems.

Define an Assurance Strategy

Too often in trusted system developments the security engineering team is isolated from the system engineering team. Many of the TCSEC's assurance requirements are satisfied by documentation that can be produced independently of the system engineering process. There is little incentive to integrate the security engineering process and the system engineering process. If a security flaw is detected late in the design, it may be very expensive to fix. Developers and certifiers need a strategy for assessing how the system's assurance requirements will be satisfied, so that they can identify vulnerabilities *before* significant development resources have been expended. Ideally, the development process is structured so that this assessment can be completed with little additional effort.

We propose that an *assurance strategy* be defined and maintained by the developer to record the assurance trade-off decisions — in the form of assumptions and

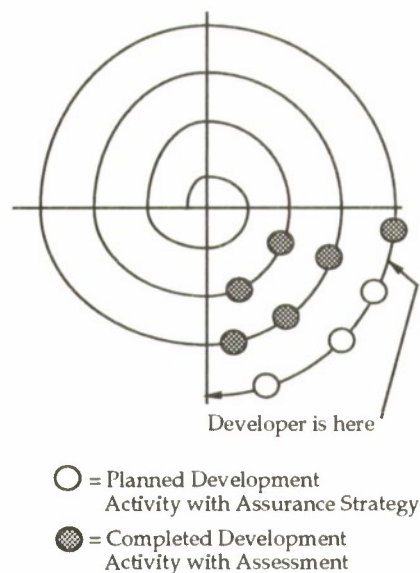


Figure 6: The development process — in progress

assertions — and to identify the techniques and methods that will be used to satisfy the assurance requirements. A strategy for demonstrating assurance is defined and assessed for each development activity and reassessed when that activity is completed. Figure 6 illustrates the relationship of the assurance strategy to the development activities.

The assurance strategy initially documents the set of assumptions and assertions derived from the requirements. It is elaborated and refined throughout the development, yielding the *assurance argument*, delivered with the system, which provides the primary technical basis for the certification decision. The assurance strategy streamlines the certification effort and makes the satisfaction of the assurance requirements a major force in the development process. In ITSEC terms, it addresses two facets of assurance: effectiveness and correctness. Evaluation of effectiveness assesses suitability of functionality, binding of functionality, vulnerabilities, ease of use and strength of mechanisms. Evaluation of correctness includes the construction and operation of the trusted system.

The assurance strategy allows certifiers to assess the role a design decision plays in the overall assurance argument and to determine whether the proposed assurance techniques are effective for demonstrating the validity of the decision. This determination, which can occur early in the system's lifecycle, also facilitates recertification and accreditation when the system is modified or when the

operational configuration changes. Development of the assurance strategy allows the developer, the accreditor, and the user to decide how much assurance is needed for different countermeasures. Every countermeasure may not need to satisfy the same assurance requirements.

The assurance strategy describes how evaluated products will be used and what role they will play in enforcing the system security policy. If evaluated products are not used, the strategy should convince the certifier why no products will suffice. Since a goal of trusted system development is to minimize the assurance effort, the certification team should be involved at the outset in assessing correctness and effectiveness.

A COMPREHENSIVE CERTIFICATION APPROACH

By completing the steps outlined in the previous section, we reduce the certification task significantly. Most of the certification process would evolve into an audit of the development activities (as illustrated in Figure 7) that contribute to the construction of the assurance argument, e.g., the formal modeling effort, the specification of the interface requirements, the design refinement, the requirements decomposition, and so on. The certification team would assess the effectiveness of the assurance strategy for each development activity and would audit the activity's contribution to the overall trusted system assurance argument.

This certification approach satisfies several important objectives. It maintains the security certification as an Independent Validation and Verification activity that proceeds along with the development. It increases the frequency of checks on the development so that they are not limited to reviews of major documents produced at a relatively small number of major procurement milestones. Thus it is flexible enough to be applied to any development lifecycle. It has ongoing visibility into the development process, but it does not threaten the independence of the certification team. Finally, the assertions and assumptions framework provide a systematic way for the developer, certifier and accreditor to reason about countermeasures and their effectiveness.

SUMMARY AND CONCLUSIONS

The trusted product evaluation approach cannot be easily applied to certifying trusted systems because it does not

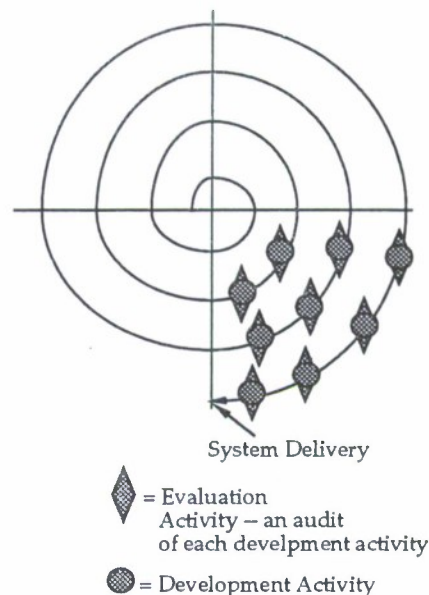


Figure 7: Certification as an audit of development

address the risks of operating the system in its environment. We have proposed a comprehensive certification approach that eliminates many of the shortcomings of the product evaluation approach. It combines the concepts of an INFOSEC policy, assumptions and assertions, a development-motivating assurance argument and an assurance strategy to move toward reducing certification to an audit of a trusted system development process.

We believe the system certification approach proposed here promises to improve upon previous methods in four ways:

1. it is based on a rigorous but flexible framework that makes the risk of using the system explicit,
2. it addresses the trusted system in its environment,
3. it includes a rigorous strategy for avoiding pitfalls in the certification and development process, and
4. it makes the assurance argument an explicit and useful part of the development process.

The next step is to evaluate this approach in more detail by applying it to a significant example.

Acknowledgements

The authors wish to thank H.O. Lubbes and the anonymous reviewers for their helpful comments.

References

- [1] National Computer Security Center, Ft. Meade, MD, *DoD 5200.28-STD, Trusted Computer System Evaluation Criteria*, December 1985.
- [2] Commission of the European Communities, Luxembourg, *Information Technology Security Evaluation Criteria (ITSEC)*, June 1991.
- [3] B. W. Boehm, "A spiral model of software development and enhancement," *Software Engineering Notes*, vol. 11, no. 4, Aug. 1986, pp. 22-42.
- [4] J.N. Froscher, J.P. McDermott, C.N. Payne, and H.O. Lubbes, "Successful acquisition of certifiable application systems (or: How not to shake hands with the tar baby)," *Proc. Sixth Annual Computer Security Applications Conf.*, Dec., 1990, IEEE CS Press, pp.414-422.
- [5] C. Landwehr, C. Heitmeyer, and J. McLean, "A security model for military message systems," *ACM Transactions on Computer Systems*, vol. 2, pp. 198-222, August 1984.
- [6] J. Froscher and J. Carroll, "Security requirements of navy embedded computers," NRL Memorandum Report 5425, Naval Research Laboratory, September 1984.
- [7] D. F. Sterne, "On the buzzword 'security policy'," in *Proc. Symposium on Research in Security and Privacy*, IEEE, June 1991.

MODULARITY OF ASSEMBLY-LANGUAGE IMPLEMENTATIONS OF TRUSTED SYSTEMS

E. John Sebes

Trusted Information Systems Inc.

444 Castro Street, Suite 800

Mountain View, CA 94041

415/962-8885

ejs@ba.tis.com

Terry C. Vickers-Benzel

Trusted Information Systems Inc.

11340 W. Olympic Blvd. Suite 265

Los Angeles, CA 90064

310/477-5828

tcvb@la.tis.com

Abstract

This paper presents an approach to assessing the modularity of trusted systems that are implemented in low-level languages. The approach presented is based on experience gained from in-depth analyses of the security features of such systems. The methods described here are centered around defining analogs of high-level language (HLL) constructs in low-level language (LLL) implementations, so that similar modularity interpretations can be applied.

This paper can serve to extend the findings of the NSA System Architecture Working Group (SAWG) to the critical class of complex trusted systems which require the use of lower level implementation languages. Such LLL modularity interpretations are essential to the application of evaluation criteria to a broad range of systems, including trusted systems for use in embedded or real-time military applications, and mainframe trusted system products such as OSs or Virtual Machine Monitors. Many such systems must use a carefully balanced approach to meeting requirements such as reconfigurability, fault tolerance, and isolation-based access control, while still satisfying modularity requirements.

Keywords: trusted systems, evaluation criteria, modularity, assembly-language, software analysis techniques, software development techniques.

Introduction

This report describes an approach to assessing the modularity of trusted systems that are implemented in low-level languages. Our work is complementary to the recent report¹ by the NSA System Architecture Working Group (SAWG), which has served to clarify the evaluation criteria for assessing modularity in Trusted Computing Bases. However, that report focuses primarily on TCBs implemented in High Level Languages, and leaves left unspecified the applicability of these "modularity interpretations" to TCBs implemented in assembler language, or to the firmware portions of TCBs.

TIS has conducted several modularity studies for system vendors and military system developers, in which we have analyzed systems implemented in assembler languages and in architecture-specific vertical microcode. Based on this experience, we have established methods for assessing modularity in trusted computing bases implemented in low level languages. These methods are based on defining analogs of high-level language (HLL) constructs in low-level

¹Summarized in "Assessing Modularity in Trusted Computing Bases" published in the Proceedings of the 15th National Computer Security Conference.

language (LLL) implementations, so that similar modularity interpretations can be applied to LLL implementations.

Thus, this report can serve to extend the findings of the SAWG report to the critical class of complex trusted systems which require the use of lower level implementation languages. Such LLL modularity interpretations are essential to the application of evaluation criteria to a broad range of systems including trusted systems for use in embedded or real-time military applications, and mainframe trusted system products such as OSs or Virtual Machine Monitors. Many such systems must use a carefully balanced approach to meeting requirements such as reconfigurability, fault tolerance, and isolation-based access control, while still satisfying modularity requirements.

Assessing the modularity of a candidate trusted system is necessary not only as part of a commercial product evaluation, but also for certification and accreditation of government systems.

In the application of criteria to higher-assurance trusted systems, a major issue is the demonstration of the compliance of the system with system architecture requirements, of which the modularity requirement is central.² Modularity is important not only because it is a pervasive characteristic of systems, but also because it is a pervasive concept of system architecture requirements. That is, if a system does not have a modular internal structure, then it would be very difficult to perform such trust-relevant analyses as: identification of the TCB modules that implement the reference validation mechanism; separation of system elements that are protection-critical from those that are not; and applying the principle of least privilege to discrete portions of the system.

For all these reasons, modularity is critical to higher-assurance trusted systems. Even with HLL-implemented systems, assessing modularity is often far from straightforward. But LLL-implemented systems lack the structuring that HLLs can provide, to constrain the interactions of code and data. Lacking this language-implemented structuring, assessing the modularity of LLL-implemented systems is not only potentially more difficult, but also more critical, due to the security ramifications of undisciplined use of protection-critical data and code. Therefore, it is especially important that LLL-implemented systems provide a strong demonstration of modularity.

Therefore, the challenge of a modular implementation in a low-level language is to:

- define LLL analogs of HLL structuring features;
- use these definitions to impose HLL-like constraints by convention;
- document the system's adherence to such constraints, both with external documentation and with comments in the code itself;
- demonstrate that the system is implemented as documented.

If this is done, then the system's modularity can be assessed by determining the pieces of which the TCB is constituted, the partitions between those pieces, and the permitted forms of interaction across those partitions.

It is our belief that this challenge can be met, such that an LLL-implemented systems can demonstrate modularity in much the same manner as systems implemented in an HLL. In fact, most of the guidelines which we present here are derived from defining the way that LLL code can mimic the structure of HLL code. These mimicing techniques can allow HLL-oriented notions of modularity to be applied to LLL-coded systems, while preserving the ability of the

²By higher-assurance trusted systems, we mean systems for which there is applicability of criteria such TCSEC B2 and higher, and ITSEC E4 and higher.

LLL to provide the hardware-specific functionality and low-level optimizations that make LLL useful.

Before describing our approach to LLL-implemented TCBs, however, we should explain what kind of modularity we are looking for. Although "modularity" and "good" software engineering practices are subjects of extensive discussion, we can focus on some implementation attributes that have been identified by the SAWG as being central to the modularity of trusted systems: *complexity*, *code cohesion*, *data cohesion*, and *coupling*.³

In order to assess these attributes of an LLL implementation, our methods focus efforts on activities which we call the packaging and presentation of both code and data. By packaging we mean documenting the organization of the code and data of the implementation to illustrate its modular structure. By presentation, we mean internal documentation in the form of comments; these comments should follow coding and commenting standards designed with the intent that the code should demonstrate the inter-relationships between modules.

The subsequent sections address the packaging of code and data, and the presentation of code and data. The main thrust of code packaging is to define LLL analogs for procedures and modules, in order to enable assessment of them on the basis of *code cohesion* and *coupling*. Likewise, data packaging defines groups of data and data structures which can be assessed for *data cohesion*. In addition, the separation of data into discrete pieces allows the statement of shared data dependencies of procedures and modules, which also contributes to the assessment of *coupling*. Together, code and data packaging provide the information necessary to state the *contract* of each module. The role of presentation is to illustrate that the code and data do in fact match the structuring definitions of the packaging, and to make the code and data definitions more readily interpretable in light of the packaging documentation.

Packaging of Code

LLL implementations typically consist of a large number of blocks of code, and the assessment of the modularity of this mass of code depends in significant part on guidance from documentation of the structure of the code. Therefore, the LLL code must be packaged into units that are similar to procedures of HLLs, and these procedures (together with data) must be packaged into modules. The inter-relationships between these modules then define the higher-level structure of the system, and also form the basis for assessments of *coupling* and *cohesion*.

Functional Decomposition

The obvious way for this state of affairs to come about is for the system to be designed using some kind of functional decomposition design approach, and for this approach to be well-documented. However, many LLL-implemented systems will not and need not follow this approach to an ideal degree. Some systems will have some amount of reused code, and this code will have to fit into the structure of the system and be documented as such. Also, some systems may be partly implemented by re-engineering existing code, i.e. not merely re-using code, but altering it for use in the system. Such re-engineering must be done in a modular manner as well, and documented. The most extreme case would be a system that is already largely completed, but not adequately documented. In such a case, the existing documentation must be expanded to document the modular design of the system; the implementation must be checked for conformance to the design; and non-conforming flaws must be corrected.

³For easy reference, each use of these SAWG terms will be italicized.

In any case, our discussion assumes that the system can be described in terms of some kind of functional decomposition, whether or not the documentation of that decomposition was done before any code was written. For the purposes of discussion, we use the term component to refer to a part of a high-level decomposition of the system, a part which is implemented by a significant portion of the code. Components are composed of subsystems, and subsystems of modules containing some collection of code and state data.⁴

An essential feature of the documentation of this decomposition is that it map onto the actual code that implements the system. In order for the implementation to be modular, each code element must belong to exactly one module, each module to one subsystem, and each subsystem to one component. The modularity of the system depends on these relationships in order for each procedure and module to be *cohesive*, and in order for the *coupling* between modules to be assessed.

Procedure Definition

However, before such modularity assessments can be attempted for an LLL implementation, one needs to define analogs for the procedures and modules of HLLs. Doing this is the first key step in assessing the modularity of LLL implementations. Typically, an LLL implementation will consist of a large number of sections of code which “call” one another, often using macros to transfer and return execution control. Such calling behavior is the foundation of LLL procedure definition. Without this, an LLL system cannot be distinguished from a large collection of unstructured code.

Therefore, procedure definition rests on two equally important features:

- regular and rigorous use of “calling” macros, and
- identification of well-defined points to which execution control is transferred, and points from which execution control is returned.

Code that lies between such points is a candidate for being a procedure. However, there are restrictions which must be taken into account for such a code block to be considered a procedure.

First of all, the entry to and exit from such a block must be well-defined. All transfers to the code block must be to the same point. Having one exit from a code block is also desirable, but it may be acceptable to have a small number of exits, if every one of them satisfies the same exit conditions, such as leaving state data in a consistent state, and assigning return values correctly. However, it may be more trouble than it is worth, to demonstrate this about all the exit points, and in many cases it would be easier to include control flow logic to arrange for one exit point. Therefore, it is highly recommended that a single exit point be used for all procedures, except in unusual and well-documented cases where there is an important reason for not doing so. An example of such a case would be a critical code section which is frequently executed, deals with many possible alternatives, has few or no exit conditions, returns no value or a simple value that can be easily identified, and for which the multiple exit points save a lot of control flow logic (and execution time) which would be used to have one exit point.

A similar argument can be made about multiple entry points, but it is even less advisable to have multiple entry points. One reason for this is that procedures in modern HLLs have only one entry point, though multiple exit points are allowed. Therefore, multiple entry points undermine the main goal of LLL modularity, namely mimicking by convention the behavior of modular HLL programs. Also, the entry requirements of procedures (such as the validation and

⁴Complex systems may of course have a deeper hierarchy of functional decomposition.

use of input parameters) can often be complex, and multiple entry points can cause significant doubt as to whether such entry processing is handled correctly in every case.

Of course, the other main attribute of procedures is that each accomplish a specific area of functionality, i.e. that it be functionally *cohesive*. To have any hope of assuring this, each procedure must be well-defined in terms of its entry and inputs, exits and outputs. Therefore, the "call" mechanism must also allow for parameters to be passed in some well-defined manner, such that code can be inspected to insure that each call provides the needed parameters. Functional cohesiveness follows from good design, but there also must be a "call" mechanism that provides for well-defined procedures that can be cohesive.

Calling Mechanism

However, this requirement for the use of a "call" mechanism does *not* mean that the full overhead of HLL procedure calls must be mimicked in LLL. Rather, the main goal is to document and regularize the use of a reasonable LLL calling discipline. Actually, a set of various calling disciplines is more likely to be useful; in some cases, more calling activity (e.g. saving registers) will be needed than in others. The full range of such needs can be accommodated, but it is essential that the resulting range of calling behavior be done in a uniform manner, using a set of "call" macros. Such macros should be used instead of inline code to perform the calling activity. Strict avoidance of such inline code is necessary for the code to clearly show each call, because these calls are the basis of the module inter-relationships that are essential for modularity assessment.

Therefore, the goal of LLL procedure definition is to provide some low-level partition of the code into well-behaved sections, so that these procedures can be used in modularity assessment. The design documentation must provide the decomposition of functionality down to the procedure level, so that one can examine the implementation to determine if the procedures actually meet their functional requirements *cohesively*.

Assessment of code *complexity* is another area that is affected by this LLL procedure definition. With well-defined procedures, one can examine the code to determine whether it has overly complex control structures or other elements of excessive complexity. Without well-defined procedures, one could not be sure that the appearance of a tidy control structure is actually undermined by undisciplined branching into the middle of a procedure. This is another reason why *all* control transfers, or "calls" must be orderly and regular, since even one undisciplined transfer (e.g. using inline code rather than a call macro) could cause some other procedure to behave abnormally. Note, however, that the use of call macros alone doesn't guarantee orderly control transfer. It is still the case that for each call, the correct amount of register saving, etc. is done for the caller and callee to work properly; in other words, the correct call macro must be used.

Once a system has well-defined procedures grouped into modules, the assessment of the *cohesion*, *complexity*, and *coupling* of the modules can proceed largely as in an HLL-implemented system. Unlike HLLs, however, the documentation must define what collection of procedures constitute a module. With this information in lieu of HLL module definitions, one can state the *software module contract* of the modules, assess the modularity of the module definitions, and then determine whether the module implementations accurately reflect the module definitions.

Data and Macro Usage

One important aspect of modularity that we have so far omitted is the scope and handling of data. That is, it does little good to say "module" and point to a collection of procedures,

if one hasn't identified the non-local data that these procedures share, or the common kinds of data that they manipulate. Certainly, the documentation of procedures must include this information. The specifics of how to document data usage are the subject of the next section on packaging of data.

However, one feature common to the packaging of both code and data is the use in the documentation of some clear structured notation (such as some form of PDL) to document what the code does and what the data means. This is particularly important because LLL implementation will contain far less of this information in the code, because of the lack of typing mechanisms and control structures. In order to assess the modularity of a system, code inspections are necessary, and in order for the code to be comprehensible it is even more important in LLLs than HLLs that the documentation provide at least a high level of guidance to the interpretation of the code.

One other note about code packaging pertains to macros. LLL implementations typically make significant use of macros, and in many cases these act as procedure templates or inline procedures. Therefore, macros must be treated as procedures as well, and subject to the same documentation and scrutiny. However, common LLL coding practice with macros can be extremely troublesome for modularity. In particular, macro definitions can be extremely complex by having many different alternatives to the code that the macro expands to; some "parameters" to macros are not actually data that the macro-expanded text uses, but are rather data that are used to define the macro-expansion itself. In extreme cases, pages of macro definition can expand to only a few instructions. This practice makes it extremely difficult to determine what the source code is really implementing, and how it relates to documentation. Since the same macro-expansion effects can be achieved by more careful macro definitions, the practice also has little justification, and is therefore heavily discouraged.

Finally, there is a class of cautions that are parallel not to HLL language issues as above, but to HLL linking and loading functionality. Typically, the compilation, linking, and loading of HLL programs uses some set of conventions about layout of code in memory, and related issues. In LLL implementations, such conventions may be absent. However, for a modular LLL implementation, there must be disciplined usage of such functionality. For example, overlaying in memory of distinct code blocks is an excellent way to undermine the important code structuring discussed above. Similarly, the use of absolute addresses (as in table-based branching) can also be quite troublesome. This is not to say that these sorts of practices are strictly forbidden, because they are certainly useful at times, and such techniques are used in assembly code generated by compilers. However, a straightforward usage of memory layout should be the rule, and exceptions should be very well documented, so that their effects on structuring can be adequately assessed.

Packaging of Data

Packaging the data of LLL implementation has a similar purpose as packaging the code, namely documenting the structure of the data, so that its *complexity* and *cohesion* can be assessed. In addition, data packaging information is indispensable to assessing the *coupling* of modules that comes about by sharing data.

The overall approach to data packaging is also related to functional decomposition. Unlike most HLLs, LLLs have no means of enforcing data access notions such as scope, access mode, and sharability. Therefore, it is essential that the LLL code use the data by conventions that emulate scope, etc., and that the data be packaged to indicate the expected usage.

In addition to lack of scoping mechanisms, another challenge of LLL data is that in typical LLL implementations there is rather a lot of data which may appear to be widely shared. This

difficultly arises largely because of the lack not only of scoping mechanisms, but also of typing mechanisms. A mass of shared data will be more palatable from a modularity perspective if it comprises one instance of a type of structured data with well-defined access methods, rather than a hodge-podge of relatively unrelated items. Therefore, packaging of data is required to demonstrate that potentially unmodular masses of data really do have structure, and that the data can be accessed as safely as though most accesses had undergone the type-checking and other consistency check that a compiler would have provided.

For LLL implementations of trusted systems, this kind of packaging is indispensable. Without it, one would not be able to find the system data that is critical to the enforcement of the security policy, and other security-relevant data; nor would one be able to state the invariants of these data that guarantee their semantic integrity, upon which depends the secure operation of the system.

Documentary substantiation of these claims will need to define:

- the separation of data, i.e. partitioning the data into distinct groups;
- the structure of data of each group;
- the legitimate discipline for accessing and maintaining the internal consistency (correctness) of the data.

On the basis of this information, the code should demonstrate that the data is accessed and maintained in a manner consistent with the documented grouping, structuring, etc. If so, then one can say that the data is accessed in the same manner as if it were implemented in an HLL. As a result, one can then apply to the data the same *cohesion* and *complexity* criteria as one would with an HLL.

The manner in which the code should make this demonstration is the subject of the subsequent section on code presentation. There are a few issues of LLL data usage practices that must be addressed separately, however.

First of all, there are issues of data layout in memory, such as the practice of overlaying one data block on top of another. A justifiable reason for this practice is to emulate the HLL functionality of unioned, discriminated, or tagged data structure types. While there are justifiable reasons for this practice, there is always a possibility that data will be stored in one semantic form and subsequently retrieved erroneously by other code with the wrong semantic interpretation. Therefore, the use of this technique must be very carefully documented, and restricted to cases where the benefits outweigh the cost of the documentation, the extra programming constraints, and the extra *code complexity*. Also, some explanation of the cost/benefit reasoning would be helpful documentation. In particular, we highly discourage the use of such techniques for critical TCB data, and assume that the potential for incorrect implementation of security-critical code outweighs the benefits of overlaying data.

Data aliasing may also occur by the use of arrays whose elements are accessed by indexed displacements. Because in an LLL these data structures have no checking on index values, there is a possibility for undisciplined data access. Therefore, the documentation must clearly indicate array index ranges, and the constraints that the code must implement to stay in range.

Of course, there may be other ways in which LLL features can undermine data packaging, but we have not identified any other ways that have justifiable utility. We can only say that if there are any exceptions to the rules stated in the the system's data packaging documentation, each exception must be very carefully documented and justified, and should be generally avoided in conjunction with critical TCB data.

Secondly, we recognize that all but the most virtuous LLL implementations will have some form of data definition and usage that is not normative in an HLL paradigm. There are invariably hardware-related or other low-level concerns that necessitate these. By comparison,

compiler-generated code and data also has such low-level features, though of course they are not visible in HLL TCB source code. Therefore, we view these as expected and acceptable, though subject to rigorous documentation of the effects on the documented usage of more "normal" data.

Finally, there is the issue of the amount of data which is shared throughout large parts of the system. We share the view that there should be only a small amount of system global data. Data packaging requirements should be most stringently applied in these cases. However, we must stress that an interpretation of the question "How little global data is acceptable?" must be based on the data packaging documentation, in order to make a parallel with HLL implementations. Even if there appears to be several instances of shared data, one must also look at the way that it is accessed. For example, if a data block is always accessed by shared code (or macros), then the data and the shared code together can be considered a module that hides the data. In other cases, this data-hiding kind of access may not apply, but documentation can demonstrate how the mass of shared data is one structure of related data, rather than a large number of unrelated data. In other words, the amount of shared data should be assessed not only by the perhaps daunting LLL text necessary to define the data, but also by the documentation that describes the data in terms of parallels with HLL data structuring concepts.

Presentation of Code and Data

The presentation of both the data and the code consists of comments made in accordance with commenting guidelines. The purpose of these comments is to explain the LLL code in terms of the packaging. The code should make it straightforward to identify procedure calls, data accesses, and control structure analogs to HLL control structures. The code should distinguish between different kinds of data access, different data scopes (e.g. working, in-component, cross-component), and different kinds of control transfers (e.g. subroutine calls vs. calls to another module).

The purpose of presentation is to enhance the readability of the code so that it gives some evidence of the correspondence to external documentation. Although the aim of presentation is to be helpful, it cannot be construed as being strictly required. It is conceivable that a HLL implementation could be deemed to be sufficiently modular even though it had few or no comments. However, for a realistic modularity assessment, good commenting is practically indispensable, and this is even more the case for LLL code than HLL code. Although the presentation will not improve the modularity of an implementation, it can be extremely helpful in tying the LLL code to structuring concepts in the packaging documentation.

Presentation of Data Item Access

One important function of presentation is to indicate the scope and usage of various data blocks, based on the distinctions between different kinds of data, scope of data, and mode of access. For instance, some procedures may have working storage that is essentially local in scope. Other data may be shared by procedures in one module, but not outside it. Still other data will be shared between modules within one subsystem, while other data will be shared throughout an entire component, or even several components.

Therefore, each data access should identify which kind of data is being accessed: local, module-private, subsystem-shared, component-shared, or multiple-component shared. In a modular system taken as a whole, there should be relatively few occurrences of the latter

kinds of access. To enable an assessment of this general rule, the code must make plain which kind of access is taking place.

Naming conventions are one simple approach to making these indications. Data should be accessed by symbolic names, and the names should indicate whether the data is procedure-local, module-local, or module-external. Each module-external reference is related to the degree to which the modules are *coupled* by sharing the data. Therefore, for each module-external references, the name of the data should indicate which module the data was exported from. Naming or commenting should also indicate whether the exporting module is part of the same subsystem or component as the importing module. Procedure-local and module-local data should have names that indicate this, and indicate which module the data is private to. Needless to say, there should be no references of module-private data for which the module name isn't the name of module in which the reference is made.

In many cases, however, the data access should not be inline, especially for data which is external to the module. This does not mean, however, that the data has to be accessed via a procedure "call"; in most cases, this is the sort of overhead that needs to be avoided, prompting LLL implementation in the first place. However, there must be some mechanism by which the data is accessed correctly via some method of access specific to that data or type of data. The best approach is to use a macro to access the data, very much in the manner of a macros or inline procedures of HLLs.

From a modularity viewpoint, these access macros should be viewed as procedures, and the macros and data together comprise a module. As a result, this form of access mimics data hiding which minimizes direct access of data which would otherwise be considered global. With structured data, the usage of access macros is especially important (whether the data is local or not) because of the concern over correct usage of structured data. However, not all data will admit of this structured-access interpretation, but will be related in some other way which also requires common access code; in these cases also, the common access should be implemented as macros or procedures, rather than by explicitly repeating code for each individual access. Even so, there may be some data which doesn't admit even of this interpretation, and which might be accessed inline, even across module boundaries. It is these cases that must be few in number, easy to find in the code, and well documented and justified.

Finally, a global comment relevant to all these data accesses, pertains to the mode of data access. In most HLLs, it is fairly clear when data is being written or read (i.e. assigned to, or used in an expression). However, in LLLs this is not always easy. Therefore, presentation of each data access should also indicate whether the data is being written or read. This is particularly important when data is being accessed by indirection. It should be clear that indirection is being used, which data are used for the indirection, and which data are the actual data that is being accessed via indirection.

Presentation of Data Segment Usage

Another concern about structured data access is the manner in which procedures of one modules use the data of other modules, not in the sense of individual accesses, but in the sense of the source code constructs necessary to make the data accessible by symbolic name.

Typically, there is some source construct that defines data names and addresses, and similar constructs that define macros. Such constructs can be used in a manner similar to the include-file mechanism of the "C" language, which is itself often used to mimic the module-import mechanisms of languages such as Modula or Ada.

However, it is important that each unit of source code contain only the "include" statements that are strictly necessary, and that these inclusions not be nested so that source code

units don't inadvertently include more than necessary. Furthermore, there should be some commenting convention for indications of which other module is being "imported" from, and why. This is critical because it can be the only centralized and regular means of indicating data dependencies between different modules.

It is these dependencies that are the basis of modularity assessments of *coupling* of modules. External documentation should describe these dependencies, but some indication of this should be visible in the code as well. "Over-inclusion" should be avoided so as not to give the appearance that the code shows more dependencies than the documentation claims.

Also, "over-inclusion" is a problem for assessment of the adherence to the principle of least privilege, since gaining access to some data would give a block of code the ability to change the data, possibly affecting some change in the secure state of the system.

Presentation of Procedure Calls

Another important aspect of presentation is to indicate "calls" to other blocks of code which constitute LLL analogs of procedures. As mentioned above, *all* control transfers must take place via some call mechanism, and there must be no instances of undisciplined branching into other code blocks. Therefore, all transfers of execution control should take place via some common macros that encapsulate the call mechanism. Therefore all inline branches (i.e. those not part of a call) should be branches that do local control structuring. For each of these, there should be comments describing the control structure (e.g. if, while, etc.). Thus, there should be no instances of undocumented branching. As a result, it should be possible to assess the complexity of the code of each procedure or macro.

Just as important as identifying calls is the differentiation of different kinds of calls, e.g. local subroutine calls vs. calls to another module. These distinctions are necessary to assess *coupling*. For example, one procedure X may be a subroutine of another procedure Y because X is called by Y, and by no other procedure. This relationship should be documented externally, but comments should also indicate that the call is a call to a private subroutine. In general, a call can be to another procedure in the same module, or to a module-external procedure. Each call should indicate which is the case, and in the latter case whether the other module is in the same subsystem or component.

As with data presentation, naming conventions can be of some assistance here. An additional approach is to have differently-named call macros, with a convention that one is used for intra module calling, and others for calling procedures of other modules in the same subsystem, or in different subsystems of the same component, or different components altogether.

This sort of information is critical for assessing the coupling of modules, and the determination of whether the actual call structure matches the functional decomposition. For example, if two modules have procedures which do a lot of calling to procedures of the other module, then one might suspect that the module definitions aren't quite right. Or, if there is real module separation but the modules are in two different subsystems, then perhaps the subsystem boundaries aren't sufficiently defined.

Automated Analysis of Presentation

Much of the foregoing discussion has been oriented toward establishing and using various documentation and coding practices that allow LLL code to behave similarly enough to HLL code, so that HLL-oriented modularity assessment techniques can be applied to LLL code. However, the assurances provided by the HLL-mimicing presentational conventions are only as good as the extent to which they are followed. In some cases, a departure from the conventions

merely raises a question to be followed up by further code inspection, and the consequences may or may not be important. In other cases, such as undisciplined control transfers, even one departure from the conventions can have potentially wide-reaching implications.

For these reasons, the modularity assurances of a LLL-implemented system can be greatly enhanced by the usage of automated code analysis tools that can detect departures from the conventions. It is hardly likely that all the code of an LLL-implemented system can be scrutinized to detect instances of poor coding practice with potentially security-relevant consequences. Therefore, automated analysis of the source code can provide valuable assurances as to the absence of such instances, beyond what is feasible for humans to accomplish.

The nature of the analysis that is feasible depends on the specific conventions, and nature of the LLL itself, but at the very minimum cross-referencing can detect some problems. Beyond this, simple special-purpose text processing programs may also be feasibly developed to search source code for other problems.

There are many examples of code features that could be searched for. Cross-reference data can indicate where specific named data are used. This information can be matched both with the information in the conventional name of the data, and also with the procedure and module definitions; these matches can be used to verify that the scope of the data is as indicated by the name, and that the users of the data are only those that should be. This information can be used not only to document the extent of such accesses, but also to compare instances of such accesses with a previously established list of such instances that are allowed for and documented. Other things to look for might include branches to code not in the same block of code, which are also not part of a normal "call"; or checking each call which appears to be an intra-module call with the procedure and module definitions, to see if it is in fact intra-module.

Clearly, there is a broad range of possibilities, and a range of technical challenges to more sophisticated analysis. However, even relatively simple practices of this sort can be a great help. For example, it could simply be part of the software development techniques for the system that cross-reference information be periodically scrutinized for conformance with expectations. Even if the amount of information and level of manual analysis is beyond the scope of those individuals assessing the system's modularity, there is still greater assurance from the use of such techniques in software development. And of course if some such automated analysis is practically useful to modularity assessors, there can be even stronger assurances that the system is as modular as it appears to be from the documentation and from examination of some of the code.

Conclusion

This paper has presented an approach to assessing modularity in trusted systems that are implemented in low-level languages. The approach presented is based on experience gained from in-depth analyses of the security features of such systems. As the evaluation community has stated, understanding the structure and inter-relationships of a system are key to analyzing the trust properties of systems. Over the years the modularity requirement has taken on a form of mystique in some circles. The SAWG report has served to remove that mystique. The authors of the SAWG report indicated that the primary motivations for requiring modular TCB implementation are understandability, maintainability, and testability. Our experience has shown that these motivations play an even greater role in LLL systems.

While many of the systems developed for evaluation as commercial products are implemented in HLLs, there are nonetheless a significant number of systems implemented in LLL. These systems are especially prevalent in the area of critical systems developed for use in government applications. The security properties of these systems are analyzed as part of a

certification and accreditation process rather than commercial evaluation. Because modularity is so central to the analysis of a system, one need not be overly concerned as to which particular evaluation criteria is being used. In fact, we believe that the techniques presented in this paper and in the SAWG report are applicable to the evaluation of systems against a wide class of criteria including such diverse requirements as flight safety, human criticality, integrity and security.

This report has described both an approach to assessing modularity and some of the steps necessary to implement modularity in LLL systems. Because our basic approach consists of defining analogs to HLL modularity constructs, these are complementary with the processes described in the SAWG report.

Our assessment has focused on the software structure and the internal and external documentation of LLL TCBs. One primary area to be addressed is the definition of a system structuring concept which can be shown to be functionally cohesive. In an LLL system, this is needed to bridge from the high-level design components to the numerous pieces of code in the implementation. Another primary aim is to use coding standards, and standards of data separation and structuring, to provide for data cohesion and to document the amount of global data and the manner in which it is accessed. We believe that these sorts of practices can codify and extend existing LLL implementation practice, rather than imposing unreasonable extra burdens on implementations, which would remove some of the benefits of using an LLL.

The eventual result of this approach is documentation of the LLL system that provides the same modularity-related information that a HLL system can provide. This information, both in external documents and in internal code comments, provides the basis for assessments of the cohesion, coupling, and complexity of the system.

We believe that the use of LLL in the implementation of high assurance trusted systems need not be considered an obstacle in meeting modularity requirements. Our experience has shown that by employing a few carefully chosen techniques in the areas of coding standards and design documentation, it can be quite feasible to implement LLL TCBs and to apply modularity criteria to them.

Acknowledgments

The authors would like to acknowledge the valuable input of Marv Schaefer, who was a key contributor to the modularity studies that led to the approach reported here. Thanks also go to Rich Feiertag for review and assistance with the preparation of this report.

Security Considerations in the Design of Multi-level Secure (MLS) Database Applications

Frank Kramer, Doug Nelson, Steve Heffern and James Studt
The Federated Software Group, Inc.
342 W. Madison, Kirkwood, MO 63122-4105

Abstract: The design of applications with requirements for multi-level security is too often conducted using standard and accepted design principles without consideration to the peculiar demands made by the MLS requirement. This report addresses some of the difficulties encountered using this approach and the strategies developed to overcome them.

Introduction

With the increase in Commercial Off The Shelf (COTS) products that are evaluated, or in evaluation, for Multi-level Security (MLS) and the emergence of MLS as a viable technology for information management systems, more and more client agencies are appending MLS requirements for new data systems. Engineering teams responsible for the design and implementation of these systems are generally inexperienced in the design of MLS applications. In addition, they are finding that they do not have a range of trusted products sufficient to assure a complete Trusted Data Distribution Path (TDDP). Finally, many of the trusted COTS products that do exist are in an immature state and do not provide all of the capabilities to perform the tasks required of the new systems. The engineering team faced with this situation typically proceeds with task phases that focus primarily on functional issues, relegating security issues to a secondary (or even lower) status. The general solution is to approach the engineering effort in three stages:

- Proceed with application engineering based on functionality requirements
- Use commercial (untrusted) versions of COTS products.
- Replace commercial products with trusted versions when they become available or sufficiently robust.

Based on experiences gained and lessons learned in the MLS retro-fit of the Air Mobility Command's (AMC) Global Decision Support System (GDSS), it may be claimed with a high degree of confidence that the success of this approach, when applied to a database oriented application, is very highly unlikely. The simple act of replacing the untrusted versions of COTS products with their trusted counterparts *will not* result in a fully functional MLS system. It is predicted that any application that is required to have MLS capabilities and is not designed with MLS security concepts at the forefront of the design

effort will require extensive reworking, to the point of a complete overhaul, in order to satisfy both the functional and security requirements demanded.

The MLS/GDSS Project

Engineering Background

The AMC used the GDSS system for planning, allocating, scheduling, and controlling the nation's airlift capabilities to support Department of Defense (DoD) requirements. The Command and Control Multi-level Security Program (C2MLSP) was initiated by the DoD to demonstrate the operational capabilities produced when applying MLS technology to the existing GDSS system.

The DoD security community showed wisdom in choosing an existing application, rather than a new one, for the MLS testbed. There are several reasons for this assertion, among them being:

- The functional requirements of the system were already determined, thus enabling focus to remain on security issues.
- The operational system had a large user community that was both knowledgeable and available to the engineering team.

The process of retro-fitting MLS capabilities to the operational system is nearing completion and has brought to light several shortcomings of accepted design principles. These shortcomings were not immediately obvious at the inception of the project and, on several occasions, their discoveries necessitated the disposal of many lines of new code in order to correct them. For these reasons, the engineering of MLS GDSS was carried out in stages, each stage refining the proof of concept for design strategies. It was often the case that the majority of code for a particular phase was discarded in anticipation of a complete redesign.

Security Requirements

In applying security technology to GDSS, the engineering team was presented with several security-specific requirements for the new system. Among these were:

- All information at sensitivity levels from Unclassified to Secret¹ will be stored in a single database. Access to information contained in this database must be determined by the user's clearance level. Control of this access is enforced by the Trusted Computing Base (TCB) and its extensions (TCBE)..
- Though not required, the inclusion of Compartments in the sensitivity levels of data should not be precluded by design.
- The concept of cover stories for classified information must be supported.

1. In the context of this report, the term "Unclassified" is synonymous with Sensitive Unclassified. The Term "uncleared user" is a user who has no clearance and is authorized only to Sensitive Unclassified information. A "cleared user" is one who is cleared to view information classified up to the Secret level.

- The system should retain the “look and feel” of the single-level GDSS system in order to retain current functionality and to minimize the retraining of personnel.
- Database fields in the single-level GDSS database should be examined to determine if they should be: a) always unclassified such that the uncleared user will always have access to the data in that field, or b) capable of being classified in the MLS database. The former are referred to as “Fixed Fields”, the latter as “Swing Fields”.
- Trusted sensitivity labels must be applied at the data element (field) level. Current MLS COTS DBMSs label data at the tuple level and not at the field level. Therefore, the TCB was extended to include the means to label individual data items in a composite tuple presented to the user.
- Sensitivity labels of displayed data must be available for display. Users must be able to view readily and trust the sensitivity labels of all displayed data items in order to determine whether that data may or may not be disclosed.

Standard Engineering Methods

Because MLS GDSS is an application heavily tied to relational database operations, most of the concepts discussed in this and subsequent sections deal with engineering practices associated with DBMSs and their client applications. The design methodologies and concepts are considered to be good and standard for modern application engineering.

In querying a relational database, using SQL, to obtain information from multiple tables, the standard practice is to use the sub-query or join to select the data. These procedures greatly reduce the processing that must be performed in the 3GL application code and is, in general, more efficient than querying each table separately, and combining the resultant tuple information within the application. In addition, these data may be sorted, or ordered, by the DBMS, thus precluding the need to sort the information in the application code.

In order to operate efficiently and maintain relational integrity, the database schema should be normalized [1] to at least the third normal form. Database tables should contain key fields that make a particular tuple unique, and non-key, attribute fields that describe, delimit, or modify the key fields. Those fields that then describe the attributes should be placed in separate tables with the described attributes as keys. Those fields that are simply entities produced by a calculational process on other fields should not be stored in the database and should be calculated separately within the application.

Application programs generally place embedded SQL commands within the application code in order to have the returned data stored in the application process space and immediately available to the application. This is mainly a performance issue designed to eliminate the overhead of multiple steps used in obtaining and manipulating data in a more distributed application.

Often, user-input data must be validated before leaving a particular form and updating any changes made. It is far more efficient to place the form enabling commands and form validation procedures within some sort of loop, such that the loop is terminated only when all modified data fields have been validated. This is often necessitated when the forms management system does not have the capabilities to perform the validations from within the form itself.

Required MLS Modifications

The Polyinstantiation Model

As previously stated, the above principles are generally considered to be good engineering practices in application design. However, in the process of designing and implementing MLS capabilities into the GDSS system, it was determined that these concepts were unworkable for reasons either of security, functionality or performance.

Before launching into the modifications made to the engineering standards, it is necessary to provide some background regarding the specific MLS requirements of the system and the design strategies employed to meet them. The requirement to provide cover story capability, when combined with the need for element-level sensitivity labeling mandated the use of polyinstantiated database tables. A polyinstantiated database table is one that can contain tuples containing the same primary key information, but holding different versions of non-key data at different sensitivity levels. The use of polyinstantiation is recommended whenever a *logical* tuple must contain data with differing sensitivities. Figure 1 provides an example of the construction and use of polyinstantiation. Consider a tuple containing Unclassified data that yields schedule information for a particular mission:

Fig. 1: Single Level Tuple

	Key	Fixed	Fixed	Swing	Swing
SL	Mission ID	Sched Dest	Sched DTG	Actual Dest	Actual DTG
U	123A	TEL AVIV	0830	TEL AVIV	0900

In this figure, Mission 123A was shown to be scheduled to arrive in Tel AVIV at 0900. The Key, Sched Dest, and Sched DTG (Date Time Group) fields are fixed fields, i.e. they are always unclassified. The sensitivity level (SL) of the tuple is shown to be Unclassified. The Actual Dest and Actual DTG fields are swing fields, i.e. they may be classified. In Figure 2, a user in a Secret session has overwritten the actual destination location and time. When the transaction is committed to the database, the existing tuple is polyinstantiated with a new tuple containing the original key and values for the Secret data. The sensitivity level for the polyinstantiated tuple is Secret. Any unchanged data in the tuple, including the fixed fields, are assigned the NULL value.

Fig. 2: Polyinstantiated Tuple

	Key	Fixed	Fixed	Swing	Swing
SL	Mission ID	Sched Dest	Sched DTG	Actual Dest	Actual DTG
U	123A	TEL AVIV	0830	TEL AVIV	0900
S	123A	NULL	NULL	BAGHDAD	0900

In conjunction with the polyinstantiation, the system, through a TCB Extension, performs a row collapse on the polyinstantiated rows such that any higher sensitivity data that is not NULL overwrites the lower sensitivity data producing an MLS tuple structure containing a sensitivity label for each data item, designated a Labeled Data Object (LDO) that is used internally and not stored in the database. A simplified LDO structure is shown in Figure 3, below:

Fig. 3: Labeled Data Object

Field	Data Value	SL
Mission_ID	123A	U
Sched Dest	Tel Aviv	U
Sched DTG	0830	U
Actual Dest	Baghdad	S
Actual DTG	9000	S

The details that went into the decision to use the polyinstantiated database tables are documented elsewhere [2], as is further information concerning the collapse mechanisms used to form polyinstantiated tuples [3].

Database Table Joins

One of the first casualties of the polyinstantiated model was the use of data table joins. It turns out that joins across database tables do not work unless each tuple in each table is polyinstantiated to a degree equal to the total range of sensitivity levels contained in the joined tables. This is untenable for two reasons: First, as most of the information in the MLS GDSS database is Unclassified, the result of providing Secret tuples would be accomplished at extreme cost both to storage facilities, and to performance; adding new sensitivity levels, *e.g.* Top Secret, or compartments would further complicate the situation in that the database would have to be completely rebuilt, along with some of the application code whenever a new level or compartment was added. Second, any join thus produced would result in the selection of many Secret-Secret tuples that contain

nothing but useless key field and NULL data, greatly increasing the overhead of both the query and the data filtering that the application would be required to perform.

Sub-Queries

A related, but less troublesome problem exists with the use of sub-queries in SQL commands. The use of sub-queries, unlike the situation for joins, does not require universal polyinstantiation of database tuples. However, when the subject field of a sub-query is not unique within a table, *i.e.* it is not a key field, there is a high likelihood that the number of tuples returned by the query will be greater than that desired. Therefore, there still exists the increased overhead involved in filtering the unwanted tuples in the application. The use of such sub-queries, though more expensive than those in which the subject fields contain unique values, is certainly to be preferred to queries performed without them.

Sorting

The use of record sorting, through use of the ORDER BY clause in a SQL command was discovered to be quite dangerous when used without a comprehensive understanding of the database schema. Its misuse is most noticeable when the table column to be sorted on is not a key field. In this case, it is highly likely that any tuples which contain NULL data will be selectively placed either at the beginning of the returned list, or at the end, depending on whether the sort is to be ascending or descending. Obviously, this effect would be manifested most greatly in Classified tuples, as they are by design prone to NULL data. The net result is the general failure of the row collapse mechanism and the release of uncollapsed Classified tuples. Even if the sort column cannot contain NULL data, as in a key field, great caution must be practiced in the design of such clauses in order that the data so returned is in the order appropriate to successful and correct collapsing of polyinstantiated tuples.

Database Normalization

As previously mentioned, one of the goals of relational database design is that the schema be in third normal form, or better. We had, in fact, redesigned the GDSS database to meet this criteria early in the design effort. As the implementation phases progressed, we found that we needed to de-normalize many of the tables that had been so meticulously crafted. One of the more compelling reasons for denormalization involves the calculation of displayed fields by the application. With the requirement that all displayed fields be labeled as to data sensitivity, the label for a field containing data that has been calculated from labeled information is, to first order, undetermined. The easiest solution is to apply a sensitivity label equal to the session label for a calculated item. But this is likely to result in the overclassification of the displayed data. Another solution is to apply a high-water-mark label, based upon the sensitivities of the data items that participated in the calculation. This, however, would require trusted code to determine, and it is not clear that simply applying a high-water-mark label to the result would represent the true sensitivity of the information. The solution chosen was to include all calculated fields in database tables, and to recalculate those values whenever a participating field is

altered. Fortunately, most of the calculated fields in the MLS GDSS application are determined from values contained in a single tuple, thus easing the performance and maintenance burdens associated with the method.

With the loss of join capabilities, some tables had to be denormalized in order to make up for the degradation in performance associated with this loss. Database queries that would normally utilize a two or three table join to fill a displayed form, would then have to perform three separate queries and include costly sorting and filtering by the application in order to produce the same information. For those procedures that were simply too costly in terms of performance, the tables were combined so that a single query would produce the desired result in reasonable time.

Embedded SQL Commands

As mentioned earlier, the use of embedded SQL commands in an application program placed retrieved data in the application process space, resulting in faster, more efficient access by the application. In this environment, however, where data labels are closely associated with the data, this would result in the need to evaluate the application code (*all* of the application code) to ensure that it does not contain malicious constructs. This is better handled through the Database Interface portion of the TCB Extensions wherein commands are composed from clauses supplied by the application, and the returned data is properly labeled and stored as LDOs in protected memory areas. The TCBE then passes pointers to the LDOs back to the application requiring the data.

Trusted Call-outs

When data are displayed to and modified by the user, it is often desirable to perform validation procedures against entered data, and allow the user to correct those data items that fail the validation. One solution is to disable the form, go through each data item, check for modification and validating those modifications, setting flags for those items that failed validation, re-enable the form, populate the form with the former data and notify the user of the invalid data. This obviously carries a high overhead and is a potential channel back into the form for malicious code or intruding processes. A more secure solution, and the one chosen for this system, is to validate the data within the form through a call-out mechanism to trusted validation procedures. These procedures must be trusted due to the access to both data and labels managed by the user interface. This closes the security breach and reduces the overhead involved with leaving and re-entering the form.

Monoinstantiated Data

The use of polyinstantiation makes a tacit assumption that the integrity of information is proportional to the sensitivity of that information; that Secret data has a higher integrity than Unclassified information. This is not the case, however, in all instances. In the case of user remarks within the application, it cannot be assumed that a Secret remark should overwrite an Unclassified remark, as the remarks may be entirely independent. For these data, the row collapse mechanism must be bypassed, and remarks at all levels for which

the process has access should be returned to the application for display to the user. Further details concerning the handling of monoinstantiated data are discussed in a previous report [4].

Conclusions

The design of applications requiring MLS capabilities should not be completed without a thorough understanding of the security implications. Some design techniques and accepted engineering standards that are considered to be *de riguer* in a normal engineering environment, may have to be abandoned, and new methodologies developed to fill the void left by their absence. In the case of an application tied to a relational database, careful attention must be paid to the database schema, and to the particular application demands made on that schema when viewed from an MLS perspective.

References

- [1] Date, C.J. (1983). Database: A primer. Addison Wesley. New York
- [2] Doncaster, S., M. Endsley, and G. Factor (1990). Rehosting existing Command and control systems into a multilevel secure environment. Proc. Sixth Ann. Comp. Security App. Conf. pp 434-445.
- [3] Nelson, D., and C. Paradise (1991). Using polyinstantiation to develop an MLS application. Proc. Seventh Ann. Comp. Security App. Conf. pp. 330-342.
- [4] Kramer, F., and S. Heffern (1992) Implications of monoinstantiation in a normally polyinstantiated multi-level secure database application. Proc. 15th Nat. Comp. Security Conf. pp. 236-242.

TRUSTED DATA DISTRIBUTION AND DATA LABELING

Doug Nelson, Steve Heffern, Frank Kramer, and Jim Studt
The Federated Software Group, Inc.
342 West Madison
St. Louis, MO 63122-4105

Abstract: This paper discusses design and developmental issues associated with the development of the Multi-Level Secure Global Decision Support System (MLS/GDSS). The MLS/GDSS is an MLS B1 version of the primary command and control system of the US Air Force Air Mobility Command slated for deployment in 1993. The discussion focuses on the implementation of a Trusted Data Distribution Pathway (TDDP) to support trusted labeling at the granularity of single data elements. To implement a TDDP using available trusted products, an architecture based on an Application Security Kernel (ASK) was developed. The ASK provides a number of security functions such as a trusted memory management module that are used to compose the commercial trusted products into a secure system. The ASK consists of trusted code that extends and melds the TCBs of the individual trusted products that form the system. The ASK represents an approach to providing a TDDP to support trusted labeling that may be highly portable to other command and control systems requiring similar capabilities.

Introduction

In the Spring of 1989, a Multi-Level Security (MLS) testbed was initiated by the United States Transportation Command (USTRANSCOM) in conjunction with the Military Airlift Command, now the Air Mobility Command (AMC). The testbed's primary mission is to demonstrate the feasibility of developing MLS military systems by re-engineering an existing Command and Control system, the Global Decision Support System (GDSS) into a multi-level secure GDSS (MLS/GDSS). The first phase of operational deployment of MLS/GDSS is scheduled for April 1993.

The GDSS is the primary command and control system of the AMC and provides flight following, mission planning and scheduling, logistics, transportation, and personnel support capabilities. In addition, the GDSS communicates to sixteen other external systems and is distributed at several sites around the world. The MLS/GDSS system, which will replace the existing system, is targeted to provide a B1 level of security assurance. It will be capable of handling sensitive unclassified through secret data and will provide all of the functional capabilities of the currently deployed system. Because it is a C² system, the MLS/GDSS not only must perform the fundamental information management operations found in most information processing systems, but it also must ensure a very high level of integrity and be capable of handling widely varying levels of transaction processing in a timely fashion.

MLS/GDSS System Design Goals

Some of the overall software architecture goals of the MLS/GDSS system include:

- Build a system that performs the same operations as the currently deployed system in a secure fashion and is capable of processing, storing and protecting Sensitive Unclassified through Secret information when the user community consists of both cleared and uncleared individuals.
- Use Commercial-Off-The-Shelf (COTS) and Government-Off-The-Shelf (GOTS) products that conform to accepted standards.
- Design and build a system that provides trusted labeling of single data elements on the user screen.

This paper contains a discussion of our efforts to design a system architecture that meets these requirements and focuses, in particular, on attempts to provide a means of protecting data and associated labels throughout the system. The discussion introduces the concept of a Trusted Data Distribution Pathway (TDDP) that is composed from the Trusted Computing Bases (TCBs) of COTS products and additional trusted coded that was developed to integrate these components. This pathway is used, in part, to deliver trusted data element-level sensitivity labels to the user

interface. The system COTS components include a trusted operating system, trusted relational database management system (RDBMS), trusted network, and trusted user interface.

Trusted vs. Advisory Labeling

We believe that there is currently a great deal of debate and at the same time confusion over the need for the display of trusted sensitivity labels to the user. In the MLS/GDSS system, it is possible for both disclosure and command decisions to be made based on the classification of single data elements. A significant amount of command and control activity involves computer-based applications in conjunction with secure and unsecure phone and HF radio communications. Users of the currently deployed GDSS system identified this need very early in the process of developing system requirements for the replacement MLS/GDSS. Our analysis of several other command and control systems has led us to believe that the need for data element level labeling is a common long-term requirement of command and control systems.

As systems integrators developing the MLS/GDSS system, we have had difficulties in understanding exactly what advantage is provided by displaying advisory labels. Advisory labels remain at the original level so that the viewer knows the actual level of the data. This is most applicable to screen-level labeling capabilities of Compartmented Mode Workstations. In the MLS/GDSS system where labeling granularity is maintained at the data element level. It is not necessary to also include advisory labels since the true sensitivity label of each element is always available to the user. If the label of a data element can not be trusted to be accurate, the display of any label other than that of the session level is confusing to the point of danger in command and control systems. It is clear, based on discussions with system users that if any sort of decision, command or otherwise, is to be made based on the label of a data element displayed to the user then the label must be trusted. Otherwise the user must treat all data as if it is labeled at his session level. The display of advisory labels to the user could have serious consequences, particularly if the user is inadequately trained in the use of advisory labels. From our perspective, in rapidly reacting command and control systems like the GDSS, display of untrustable data, i.e. advisory labels, to the user worsens his work load and may complicate his decision making. We realize that this is an extreme position and may not be applicable in general to all systems. In particular, advisory labels could play an important role in information analysis systems. Our experience with C2 systems, however, points out the absolute requirement for trusted labeling of data elements in command and control systems.

Trusted Data Distribution Pathway (TDDP)

A major obstacle in fulfilling the goals of the program, to utilize COTS products and provide trusted labels on the user's screen, was the design and implementation of a complete protected pathway for the movement of data and labels to and from the RDBMS and the user interface. This component of the system we term the Trusted Data Distribution Pathway (TDDP). The TDDP encompasses all of the TCBs of the COTS components that comprise the system and all trusted code that was developed to compose these products into a functional system. Thus, the TDDP is a union of all the TCB components that form the system.

TCB Composability

In designing the system using COTS products, a number of shortcomings in the ability of these products to be composed to form a TDDP were identified. These shortcomings were overcome by the development of additional trusted code that was designed to extend the TCBs of the COTS products and to complete the TDDP.

To provide a TDDP that permits the user to trust data sensitivity labels displayed on the screen, all of the individual TCBs of the COTS products must be connected to each other in a secure fashion. Each product must also provide within its own TCB a complete and trusted pathway to manage data and associated labels. As an example, consider an MLS RDBMS product. Most of the products that are currently available include a trusted relational database engine that forms the core of the RDBMS, and associated Application Programming Interfaces (APIs). Many of these products also support a client-server model, with the RDBMS engine capable of residing on one machine and an API component resident on a different machine. Presumably the two machines are connected with a trusted MLS network.

Much of the focus of the vendor's development, and indeed the evaluation process of these products, is on the relational engine, its security policy and TCB. In order for these products to be used to develop systems that provide trusted data labeling, each component of the product having write access to the data and labels must be evaluated for trust. Not only must the relational engine be secure, but its interface on the server and the API components on the client machine must also be trusted. Thus, the TCB of the RDBMS must encompass all of these components. Unless these requirements are met, the MLS RDBMS can at best only ever provide advisory labels, forcing the classification of all data displayed to users in an MLS environment to be treated as session level. It does little in terms of developing systems that provide trusted labeling if only the relational engine forms the TCB of the RDBMS products. Unless there is a trusted means of exchanging data and associated labels between the trusted application code and the API of the RDBMS, the labels are only advisory.

We experienced this problem with Version 1.0 of the Sybase Secure Server whose relational engine was developed to meet B1 assurance levels. Unfortunately, the database API used to communicate with the relational engine had no trust associated with it. As a result, the Sybase product could not be used to develop systems requiring trusted labels.

This requirement imposes difficult constraints on the product vendors and on the evaluation process. RDBMS products are currently treated as layered system components and evaluated against the Trusted Database Interpretation (TDI) using a particular combination of RDBMS and underlying operating system. In a client server architecture, or even when all components of the RDBMS are resident on the same system, the MLS network must be considered in the evaluation process if these products are ever intended to be used to develop systems providing trusted data labeling.

Protected Data and Labels Storage

One of the obvious requirements that must be met by the TDDP is the protection of data and associated labels as they traverse the TDDP. At no point must untrusted components of the system be permitted access to the labels in such a way that the labels might be modified in a manner that violates the security policy enforced by the system.

We assume that the TCB of the evaluated COTS products provide adequate protection of labels. A major shortcoming in using available COTS products to build systems similar to the MLS/GDSS is in the area of the user interfaces. This is discussed in more detail later in the paper. In composing the system from various TCBs, it was necessary to develop a trusted repository for data and labels available to the applications that make up the system. Once the data and associated labels leave the RDBMS and are delivered to the application requesting the data, they must be protected. This trusted repository provides secure memory management capabilities to the applications. Secure means that the data and labels are readable within a single process but can only be written by trusted, privileged code. Secure memory management is accomplished using kernel mode memory pages that are only accessible from trusted code operating with privileges. The memory repository operates in much the same way as the operating systems memory manager. Once the data and labels enter the address space of the application, they are accessible to any code executing within the user's process. Without a protected memory manager, all application code would need to be trusted not to violate security policy and accidentally or deliberately modify labels. The executables that make up the MLS/GDSS applications are built from sources including more than 500,000 lines of Ada code. Requiring trust of this much code is not feasible. Use of a segregated and protected memory manager helps permit the development of applications in a secure MLS environment without requiring them to be trusted. Thus, the secure memory manager provides intra-process protection of labeled data.

Security Policy Enforcement

Another significant issue in composing COTS products into a secure systems capable of displaying trusted labels is that of consistent security policy enforcement. When incorporating multiple TCBs into an integrated TDDP, it is obviously important to ensure that all of the components enforce the same security policy. Fortunately, the security policies of currently available products all support a modified Bell-LaPadula model. Most of the products do not permit write-up of data, although this is allowed under the original model.

In composing these COTS components into an integrated system it was necessary to support a similar model within

the trusted code components that were developed to support applications development. The overall security policy that was developed for the MLS/GDSS systems also incorporates a number of requirements such as data ownership rules, downgrading rules, and auditing rules that are not inherent in the TCB components. The additional trusted code that was developed to support a TDDP is responsible for enforcing these components of the security policy within the process space of the user. Through the programming interfaces to the COTS products, trusted code developed to compose the system also ensure that these requirements are supported by the TCB of the appropriate COTS product. The trusted code interface to the COTS products works in concert with the TCBs of the COTS products to extend their capabilities to meet the security policy.

One example of the enforcement of the security policy by the trusted code that was developed to compose the system concerns the generation of data created in the user's process. Any data that is extracted from the RDBMS is labeled by the RDBMS when first entered into the relational engine. However, data created by the user by direct entry into the user interface forms or by the application code in processing this new data must be properly labeled. The MLS/GDSS security policy stipulates that any data created within the user's process must be labeled at the level of the user's process. Enforcement of this rule is performed by the trusted memory management component of the system when memory is allocated for the storage of new data.

It is possible, for example, for a user to enter new data into a screen form and then request the system to display the sensitivity of the data before the data is actually stored in the RDBMS and properly labeled by the TCB of the RDBMS. The same is true of data that is created to build the appropriate tuples to insert into the RDBMS. New data entered into the system by the user through the user interface may cause the application to generate several tuples of data to support the relational model.

Addition of trusted code to complete the TDDP was necessary in this case because the TCBs of the COTS products used in the system do not extend into the application domain. The TCB of the RDBMS ends when data is transferred out of the API of the RDBMS. The TCB of the operating system enforces labeling of data entering or leaving the user process and enforces inter-process communication labeling. There is no support for intra-process security policy enforcement or protection by the TCBs of any of the COTS products. For this reason, it was necessary to develop additional trusted code. We were aware of other approaches to the problem but these other solutions did not provide adequate domain separation of data and associated sensitivity labels within the context of a single user process. The granularity of data labelling and protection provided by the COTS products and other approaches was insufficient to meet the needs of the program.

Data Classification Constraint Enforcement

Another requirement that was encountered in designing the MLS/GDSS is the need to support data classification constraints. Classification constraints are rules that stipulate what levels of classification are permitted for data contained within specified fields. While most of the COTS database products permit the enforcement of such constraints within the RDBMS, there is no generalized support outside the RDBMS TCB. In composing the system, it was necessary to add trusted code that supported these rules throughout the TDDP.

For example, if the user operating at a SECRET session attempts to enter data into a field that is constrained to hold only UNCLASSIFIED data, the resulting classification constraints must be immediately enforced. Operationally, it would not be feasible to allow the user process at the user interface and application level to violate security classification constraints but enforce these rules inside the TCB of the RDBMS. If the user were allowed to violate the classification constraints rules at the user interface, they could inadvertently enter classified data into the screen and be shown a label for those data elements that violated the security policy and could lead to inappropriate command decisions. Only when the user attempts to commit the data to the RDBMS would the constraint violations be detected and reported. This is another example of ensuring that the TDDP enforces the same security policy throughout the pathway, and not just within the TCB of a single component.

Data Element Level Labeling

As stated before, one of the requirements of the MLS/GDSS is to provide labeling at the granularity of single data

elements. This granularity of labeling, however, is not supported by the currently emerging MLS RDBMS products. The RDBMS products provide tuple level labeling which is translated to single data element level labeling by trusted code that interfaces to the RDBMS API and uses the protected memory manager of the system. This code uses a process called row-collapsing and polyinstantiation to support data element level labeling and cover stories. The design of this component has been described elsewhere[1]. Since this component has the capability to modify labels directly, it must be considered trusted. As data and associated labels are received from the RDBMS API labeling granularity is translated from a tuple-level to data element level, and the resultant data set transferred to the protected memory manager. This interface is constructed from trusted code that was developed to compose the system using COTS MLS RDBMS products. Its translation of tuple-level labeling to data element level labeling assumes that the underlying RDBMS securely transfers data and associated labels from the data structures within the RDBMS's API and those allocated by the database interface code to receive them. The database interface provides a number of other functions that are beyond the scope of this paper.

The Application Security Kernel (ASK)

In order to provide a TDDP and provide isolation of the TDDP from the command unique applications, we developed an architecture that provides a layer of TCB extensions that reside between the applications and underlying COTS products. The core of the system is called the Application Security Kernel (ASK) which fulfills two very important, basic functions. First, the ASK provides an isolation buffer between the Command Unique Application Code and the Trusted Products which form the majority of the Trusted Computing Base (TCB). Second, it provides extensions to the TCB that enforce the security policy of the system, as well as extensions to the capabilities of the COTS products to meet the functional requirements of the system. The relationship of the ASK to the other major components of the system is illustrated in Figure 1.

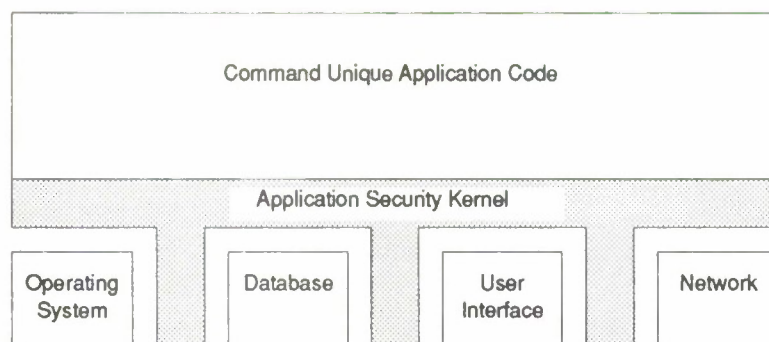


Figure 1: The MLS/GDSS Software Architecture

The ASK consists of a body of procedures, written in C, that serve as an interface between the Applications and the Trusted COTS Products. Its primary purpose is to provide a complete set of application programming services which unconditionally implements the security policy. If the application code uses the ASK to access the underlying COTS products, then no sequence of operations which the applications can perform will cause the security policy of the system to be violated. As such, the application code does not have to be treated as trusted code during the security evaluation process. This is key to achieving accreditable security in the system, since the security kernel represents just 1% of the total code in the system. Thus, the focus of the accreditation process may be *significantly* narrowed. Obviously, if applications that are not constrained to use the ASK are permitted on the system then there is the potential to violate the security policy. Through configuration management of software and proper configuration of the COTS products, the presence of non-compliant software is eliminated.

The ASK provides several critical capabilities to the system. Included in these capabilities are the following:

- A complete TDDP that includes a protected memory management component that is available to the applications and other ASK components. This repository protects the data and associated labels within the context of the user's

process and in conjunction with the TCBs of the COTS products provides a trusted pathway for transfer of trusted labels to the user interface.

- Trusted computing base extensions that enforce the security policy of the system and provide security functions to the applications through an open library containing numerous functions. These functions include support for such items as label translation between the various COTS components that comprise the system, trusted memory management of data structures used by the untrusted applications, and trusted access to various services provided by the COTS products.

- Extension of the capabilities of the COTS products to meet both security requirements and functional requirements of the system. The security extensions include such items as extension of auditing capabilities of the COTS products that make up the system to meet the requirements associated with a B1 trusted computing systems and those additional requirements needed for command and control systems such as GDSS. An example of a functional extension is the ability of the system to provide data element level labeling to the user. Current MLS database management systems provide, at best, tuple level labeling capabilities. Because of the need to make disclosure decisions regarding the values of individual data elements associated with a mission (such as landing time) to unclassified users of the system, it was necessary to extend the labeling capabilities of the system to the granularity of single data elements.

Processing of Application Commands by the ASK

Figure 2 shows an expanded view of the COTS components and interfaces provided by the ASK. The stippled sections of the diagram indicate those components that have been composed to form the TDDP. Three of the major components of the ASK are illustrated: label manager (LM), database interface (DBI), and trusted user interface (TUI). The command unique application code comprises the majority on the developed code of the system and is not trusted to enforce security policy.

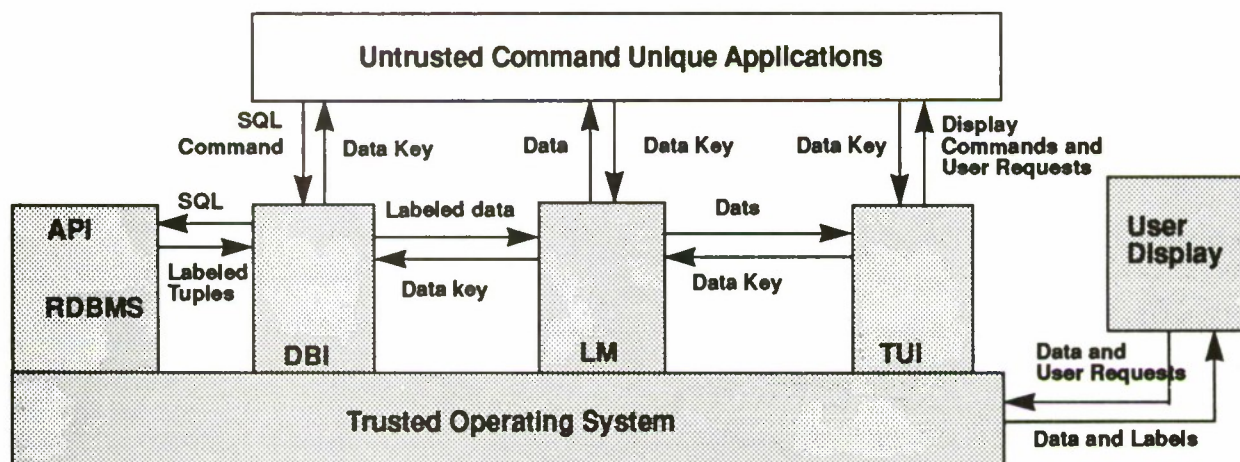


Figure 2: Process and Data Flow in the TDDP and ASK

When the user, through keyboard commands, requests data residing in the database to be displayed, this request is passed to the application code which issues multiple query strings to the database interface component of the ASK. The ASK composes these strings into SQL commands and issues these commands to the RDBMS, processes the returning rows to provide data element level labeling and passes the resulting data set to the label manager. The label manager allocates protected memory to hold the data and labels and returns a data key to the database interface. This key is in turn passed back to the application. The application is free to request copies of the data, make decisions based on the data, or generate new data requests in preparation for displaying data to the user. The application then instructs the trusted user interface component of the ASK to display the data by passing the appropriate data keys. The user interface then assumes control, requesting the data and associated labels for display directly from the label

manager using the data keys. The user interface then processes commands from the user. When the user modifies existing data or enters new data and requests processing of the form, the user interface transfers the data to the label manager which enforces the proper labeling of the new or modified data elements and returns a data key to the user interface. This new set of data keys is then transferred back to the application. At no point does the untrusted application code have the ability to modify the labels associated with the data. With the exception of the RDBMS server, all of the components operate within the context of a single process.

Trusted User Interface COTS Products

The MLS GDSS system is primarily a forms-based application; users are presented with forms containing information that has been read from the relational database. Users have the ability to modify the information and store the modifications in the database. The user interface in a MLS system has to either enforce the security policy of the system or be guaranteed not to violate it. In the case of MLS GDSS, the user interface has to:

- Show the user the security label of every piece of information on the screen
- Ensure that data entered by a classified user is marked as classified even before the data is entered into the database
- Provide a capability of covering up all of the classified data on the screen
- Provide a trusted path between the user and the remainder of the system
- Support character cell and X-based terminals
- Provide a migration path to windowed workstations.

At the present time, there are no commercially available user interface products that meet these minimum requirements. However, the FIMS standard is available as a forms management standard to which any future trusted forms product is likely to adhere. At the present time, there is an emerging B2-compliant product that is being evaluated by the MLS/GDSS project.

Summary

This paper has presented some of the issues of providing trusted labels in the MLS/GDSS that is being built using Commercial-Off-The-Shelf (COTS) products and accepted government and commercial standards to produce a portable system design and architecture that provides a B1 level of security for command and control (C²) systems. As a result of the effort, a highly portable system architecture and Application Security Kernel (ASK) that provides security extensions and seamlessly integrates a number of secure COTS products into a functional information management system have been produced. The ASK is designed to support a Trusted Data Distribution Pathway to permit protected management of data and associated labels and ensure trust of sensitivity labels displayed to the user. The AMC is currently using this architecture and software to develop several secure C² systems.

References

- [1] Nelson, D., and C. Paradise (1991). Using polyinstantiation to develop an MLS application. Proc. Seventh. Ann. Comp. Security App. Conf. pp. 330-342.

COMPOSING TRUSTED SYSTEMS USING EVALUATED PRODUCTS

Daniel Gambel
Deputy Director

Joan Fowler
Security Architect

Grumman Data Systems
2411 Dulles Corner Park, Suite 500
Herndon, VA 22071

Abstract

This paper discusses a method for composing trusted systems using evaluated products. This method assumes modification to the Trusted Computing Base (TCB) of the composing products will be necessary in some manner to achieve successful adaption. Unless justification and review is performed on a product's modified TCB, and control exercised over how the adaption is accomplished, the original product rating could be invalidated. This paper also addresses how system-level assurances can compensate for the product assurances invalidated due to such adaption.

Introduction

Over the past several years, the Evaluated Products List (EPL) is beginning to contain more Commercial Off-the-Shelf (COTS) products. In today's COTS-oriented economic environment, the proclivity is towards using evaluated products when designing trusted systems. A trusted system in the context of this paper, is a system composed of multiple products which, at the interface to the resulting TCB, conforms to the Department of Defense (DoD) "Trusted System Evaluation Criteria" (TCSEC) (DoD 5200.28-STD) and the forthcoming TCSEC-derived protection profiles of the Federal Criteria.

The difficulty in using multiple evaluated products in a complex trusted system solution is that generally each product performs all functions required by the level of trust without adequate mechanisms for distribution or remote centralization of functions across product lines. Product vendor's are reluctant to support radical adaptation, and the interfaces within a product are usually not well defined which is a condition necessary to achieve interfacing between products.

Classic system engineering approaches based on requirement decomposition have been used to remedy the problem in non-trusted system designs. We show in this paper how the same techniques can be used to achieve a well-organized trusted solution.

Typical System Design Methodology

The methodology used to design a trusted system with evaluated products is much the same as designing any system using COTS products. The design of a trusted system may take an additional step which is not completely necessary when designing a non-trusted system. The additional effort is required to ensure that the resultant system TCB retains behavior as an intact entity.

Methodology for Designing Typical Systems

The methodology for designing a system of COTS products is to decompose the system functional requirements, allocate the requirements to a set of products, recompose the system of integrated products to ensure cooperative functionality, and assess the results ensuring completeness of the solution compared to the original requirements. This is true of any system based on COTS products.

During decomposition, the system requirements are decomposed to a level at which the requirements can be allocated to a single or multiple COTS products. Since the task of engineering the system can be very large, the overall requirements are typically divided into high level groups of requirements approachable as a combined set of requirements with a hopefully common solution. In identifying COTS products, the process is to decompose the requirements until the products become visible. After a set of products are identified, the identification of the subset of requirements that the product fulfills is defined. A satisfactory fit of product to requirement results in a tentative allocation of the product to the solution set. This positive identification also results in a residue found by comparing the original set of requirements with the set of satisfied requirements. The continuing process is to decompose the remainder of the requirements until additional COTS products are identified and allocated to the solution. At this point, the iterative process of decomposition/allocation will continue as long as unfulfilled requirements remain.

In general, the decomposition to two contiguous levels without the emergence of a suitable product would mean that the function must be uniquely designed and developed for the system. An alternative to determining a design and development entity is to choose a different mix of products at a higher level of decomposition (recursive and regressive) in an attempt to identify a solution where the need for a unique design is further reduced. This decomposition/allocation process with the system requirements can continue until the full range of products within the technology is examined and the development effort for the system is truly minimized.

The above allocation identifies the product mix best suited to performing the decomposed set of functions. The next stage, recomposition, assesses the integrity of the system allocation and identified interrelationships between components. During recomposition, the interactions of one product with the others are assessed and resolved to meet the operational requirements. The effect of the operations of one product can produce a negative impact on another product. For example, the mail

service typically depends on a more primitive service to perform communications. This is generally achieved using well-defined interface specifications. Similarly, a Data Base Management System (DBMS) typically relies on a file system abstraction provided by an operating system. If either the mail service or the DBMS are allocated into incompatible environments, then the allocation is negated. Again, a new product may be chosen, or a design or modification effort undertaken. If the optimum allocation is precluded by unacceptable product modification levels, then the process of allocation must be redone.

Assessment is conducted by an independent engineering organization and is based on the evidence developed during the design process. This evidence, in untrusted scenarios, can be provided simply through testing during unit-level, integration, and operational testing. This level of assurance that the system works correctly is primarily supported by the assessment provided by the test program. Normally little documentation, of how the various products were integrated into the whole system, is necessary.

The outcome of this system engineering approach is a definition of system requirement groups and the allocation of these groups to COTS products. Each group selected for the system should be internally cohesive in that it performs a single high-level task and requires little interaction with the other groups in the system. [7] Another objective in determining the groups is to minimize coupling between the groups to make them as independent as possible. [6] Of course, no system can exist without some coupling to preserve system cohesiveness.

Typically, each product chosen during the initial decomposition process provides a surplus of functionality beyond that needed to satisfy the system requirements. System engineers view this surplus as additional value, enhancing the solution. Only where one product's surplus interferes with another desirable products operation are attempts made to minimize the capability provided by any single product. In a like manner, many products are dependant upon other products to function at their fullest. No single product meets all of the system requirements, any given set of application products provides a surplus of functionality, and in many cases, provide redundant features. An example is a redundant mail service provided by both a DBMS and an office automation application. Another example is the communication features provided by both the operating system and the communications application vendors. The net effect is that the vendor provides a surplus of features, alternative answers exceeding the specified requirements exist, and the user enjoys alternatives resulting from this surplus embedded in the product allocation.

This methodology (decomposition, allocation, recomposition, and assessment) can be followed designing any system using primarily COTS products. Of course, a system usually can not be built using only a combination of COTS products, some "glue" must be available to tie the products into a cohesive system. Some of this "glue" may be another COTS product, a set of "switch selections" for selected products, or it may be developed uniquely for the system.

Methodology Differences for Trusted Systems

The design methodology described above can also be used when designing a trusted system. The difference is that the process must be taken a step further. The security functionality of each TCB in the identified trusted products may not satisfy all of the security requirements of the system. Surplus security functionality must also be eliminated to achieve a minimized TCB, eliminate conflict between overlapping components, and provide for a single policy across the system environments. A minimized TCB is important to permit examination of the security element and to minimize the cost of documentation. Overlapping components should be eliminated because of the potential impact on operational performance, user acceptance, and ease of administration. A single system policy is critical since, if there is more than one policy, the weakest policy is the only one that can be assured for the total system.

The modification, adaptation, and elimination of security functionality in evaluated products can take many forms. The easiest and most trusted form is to tune the product using the product's switches and mechanisms. For example, in many products it is possible to audit on all or no activity for a user. Another form is to use the product as it was not necessarily intended to be used. If a product with Mandatory Access Control (MAC) labels and controls is used in a system high environment, the MAC processing actually occurs in the execution of the software, but it does not have any relevancy in the system. Another form of eliminating security functionality is to actually modify the code of the product. This form is the least desirable and should only be done when the system requirements dictate that product code modification is the only solution.

The TCB composed of the products and the system must be addressed for insufficiencies and redundancies. There may be system security requirements that are not fully addressed by a combination of all of the product TCBs. Or, there may be security requirements that are addressed in each of the product TCBs selected. This is a frequent occurrence (e.g., audit or identification and authentication (I&A)). It is especially a problem when there are inconsistencies between the handling of the security requirements between the various products of a system. These inconsistencies can irritate a user, but can also defeat an explicit countermeasure in the redundant feature of one of the other product TCBs (e.g., different identifications, discretionary access control (DAC)).

During the decomposition, the process of trusted system engineering is basically the same as in system engineering. The primary difference is that the vocabulary/grammar used in trusted systems is distinct from that used in system engineering. This distinction can lead to confusion between the two communities. Thus, a set of requirements contained in the DBMS set may appear to meet requirements in the TCB set. However, since the TCB is based on a different vocabulary, there is actually no commonality. This is particularly true for constraints placed on DBMS views which restrict a user operationally, but are insufficient to meet requirements for the trusted system constraints.

During the allocation, the need to minimize the TCB dominates the trusted system problem. While overlap and redundancy is probably desirable in application functionality (i.e., an extra capability in a word processing package may provide an additional resources for a user), the TCB cannot afford the restrictions and contradictions invoked as a result of such overlap of security functionality. This abstraction is singularly attributed to the TCB, and should not preclude overlap within application layers of the system. Since the TCB is generally central to the system operation, it applies constraints to the system solution. A determination to allocate a subset of a specific security requirement to a component necessitates an appreciation for the impact of that allocation on the set of documentation, potential interface requirements, functionality, and user acceptability as a result of that decision.

During the recomposition, the TCB is integrated to identify potential redundancies and conflict between the components. Typical is the need for elimination of "login" processing in favor of a unitary login. Also typical is the need for a centralized authentication and user/terminal profile data base to minimize administration and eliminate desynchronization of the user permission sets across components. This process invariably leads to a need to adapt the product TCB to perform a lesser set of functions within the system TCB. This adaptation also necessitates two documentation impacts. First, the product documentation needs adaptation to define the operationally retained functionality. Second, a system-level documentation set is required to define the interfaces between the implemented components of the TCB.

During the assessment of the TCB, it is not generally adequate to simply test the features. Testing is usually inadequate to demonstrate layered constraint features of the system. It is impossible to determine through testing whether the application layer, the operating system layer, the communication layer, the TCB layer or the reference monitor of the TCB is actually doing the constraining of the user. Thus, an asserted set of features may be functionally inoperable without detection using testing. Verification of documentation, along with penetration testing, is the current technological solution to this dilemma.

The search for a product evaluated at a specific level produces candidate processor/operating systems. All of the candidate products may meet the level of trust requirement. However, these products must also fulfill the remaining non-security related performance and functionality requirements of the system as a whole.

The difficulty in using multiple evaluated products in a complex trusted system solution is that generally each product performs all required security functions for its evaluated TCSEC class without adequate mechanisms for distribution or remote centralization of functions across product lines. This precludes using multiple trusted products in bringing a candidate system toward a secure solution using trusted approaches. As a result of the vendors providing comprehensive solutions within the single product, there is a serious design difficulty in cooperative processing between products.

Example Distributed System

An example distributed system, used for this paper, has a guard, application servers, and an I&A component. Figure 1 illustrates the overall architecture in this example system.

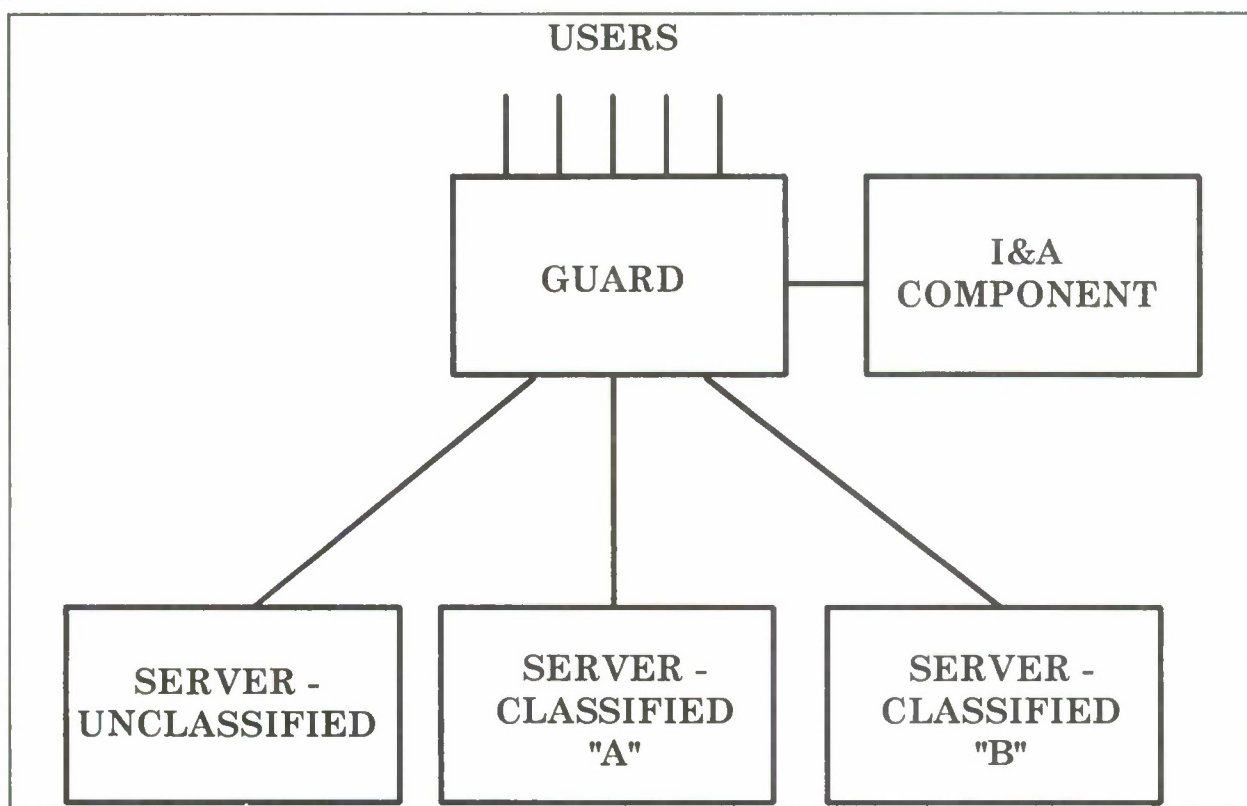


Figure 1. Example Overview System Architecture for Distributed Systems

High Level TCB Definition

A third-level decomposition, in our example system, leads to dividing the TCB functions and the application functions through a distributed system. In this distributed system, the TCB is distributed by mandatory access control (MAC), DAC, I&A, and Audit components. Further decomposition of the requirements is not necessary for the non-TCB elements. All of these functional applications reside on each of the single-level servers.

This level of decomposition does not suffice for the description of the TCB since the relationship between the TCB components requires further illustration. Figure 2 illustrates the actual distribution of each of the TCB elements in this example system.

At the chosen level of decomposition, the I&A function is decomposed into (1) system approach, which is composed of the trusted path, the user ID, and the

password; (2) the authentication element, which is the password and/or a token based authentication device; and (3) the registration element, which registers users on the single-level servers. The audit component is satisfied internally for each component.

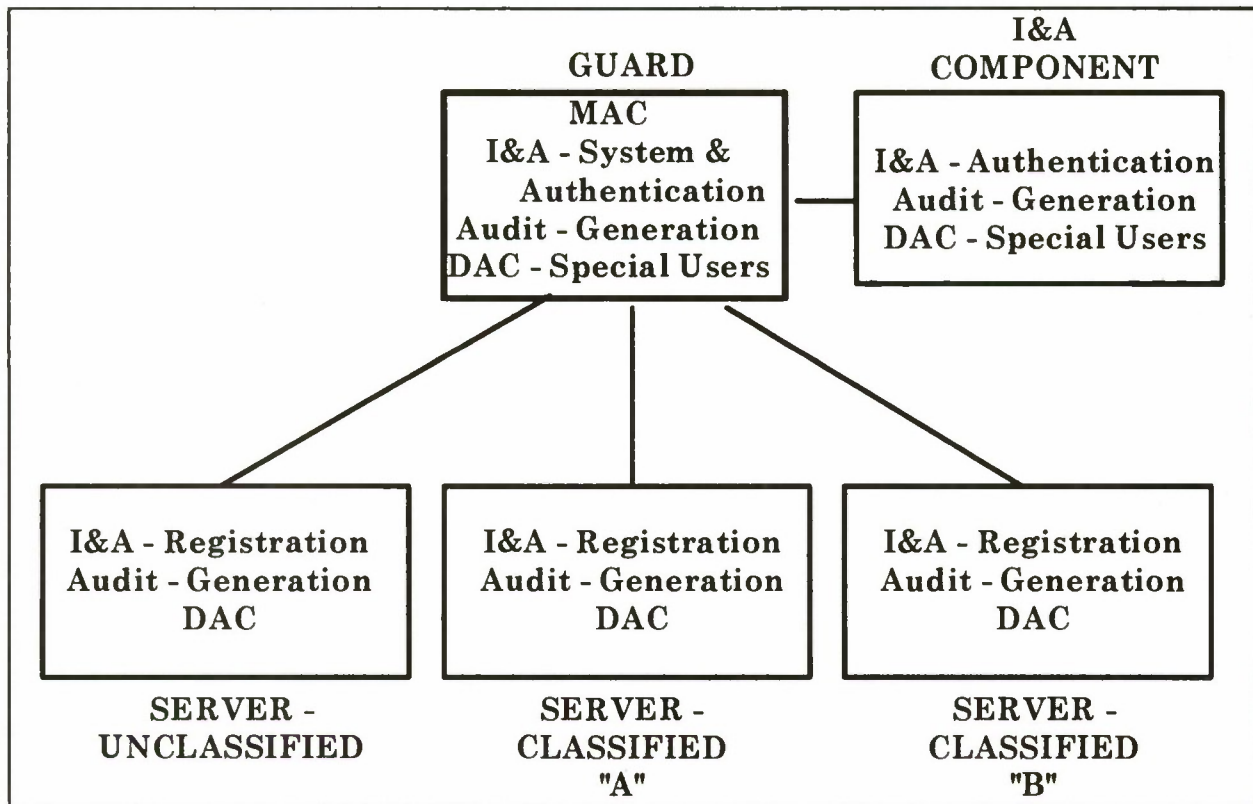


Figure 2. Distribute the Elements of the TCB throughout the Architecture

The MAC component is performed on the guard for the system. The DAC component requires no decomposition and is allocated to the servers. The exception to the previous statement is the allocation of role based special users (e.g., security administrators) that requires access/protection for all elements of the system. The normal external user has no requirement for DAC on the guard.

TCB Mechanism and Interface Analysis

At the lowest level of practical decomposition, the coherency necessary to ensure that the system meets the required level of trust is not readily apparent. In order to ensure that the attributes of the desired level are provided, recomposition is required. As a result of recomposition, significant elements of the product TCBs are found not to be required in the combination of the active elements (e.g., the full I&A capabilities of the single-level servers). These elements are redundant capabilities among the product TCBs. In order to address a single system policy and provide cohesive protection for the system, the elements of product TCBs that are redundant must be

eliminated. For example, in the system described above, the guard performs the I&A for the system. However, each of the server operating systems were developed and evaluated as stand alone entities. Therefore, each of the servers has an I&A capability that is redundant for this system. These redundant I&A capabilities must be eliminated to give the system a unitary logon capability. All of the security functionality (mechanisms) on each of the evaluated product TCBs must be similarly dealt with to determine their applicability within the constraints of the overall system TCB.

The mechanisms which perform the TCB protection are not the only part of the product that should be addressed. Each of these mechanisms has interfaces to other TCB and non-TCB mechanisms within the product. These interfaces must all be dealt with to define the system TCB using the product TCBs as components. Some of these interfaces must be eliminated, others must be redirected to interface between the products or with new elements and mechanisms unique to the system. In the case of the unitary logon described above, the server I&A capability must be modified to accept the registration from the guard, and the guard I&A capability must be modified to register a user with the added server registration capability. The interfaces to the server I&A capability from the evaluated product must be eliminated, the interface between the guard and server must be introduced for I&A registration, and the interfaces between the modified server I&A registration capability and the remaining server operating system must be modified. All of this must be accomplished in a secure manner so as not to jeopardize the system I&A as a whole.

Example Conclusion

This methodology is based on the approved methodologies of both the Trusted Network Interpretation (TNI) [4] and the Trusted Data Base Interpretation (TDI) [5]. Specifically, the TNI concept of partitioning, and constraining the partition, was combined with the TDI concept of TCB subsets wherein the less primitive subset is constrained by the more primitive layer. The partitioning concept of the TNI leads to the simple top-level concept of decomposition in which the system-level TCB is extended with each server partitioned from each of the others by the guard. In addition, the TDI concept of constrained subsets is used in the relationships between the guard and the servers, in that each server is constrained by the hierarchical aspects of the guard M-component. The MAC partition is invoked by the guard between each server. Within each partition is a constrained subset of the M-component. The result of this combination is a partition of the system within which a subset is constrained. This architecture is then translated to functional and physical components through decomposition and allocation techniques as described above.

Trusted System Assurance

The modification of EPL products into a unique system may invalidate the assurance documentation developed for the evaluation of each product. The system-

level assurance documentation must compensate for this product assurance documentation.

The TCSEC describes the assurance documentation required for trusted systems. The required documentation that is directly applicable to the actual implementation of the system falls into two categories. The first is the requirement that "if the TCB is composed of distinct modules, the interfaces between these modules shall be described." [1] This interface description is required by the TCSEC for all levels above Division D, Minimal Protection. At Class B1, Labelled Security Protection, the TCSEC requires "the specific TCB protection mechanisms shall be identified...". [1] Since the COTS product documentation already exists, it is prudent to discuss these interfaces and mechanisms, as they have been modified, in the system-level documentation.

System-level Assurance Compensates for Product Modifications

In a composed system, the system-level assurance must describe the modifications to each of the products, as well as the system additions necessary to combine the products. During the design process, the product-level documentation must be assessed to determine the modifications and additions that are necessary. Most of the product-level documentation should be applicable to the system. If it is not, then perhaps the functionality of the product is not consistent with the system requirements as originally observed, and a new product should be selected that does not require such extensive modification.

In order to limit the cost of system integration the changes to products should be minimized in the integration of the system. In our example, we have described a methodology to eliminate software development risks, we have also coincidentally minimized the scope of the design effort necessary for the system. This is an approach to be considered to minimize system risk: to minimize the design such that the assurance requirements are primarily at the system level. The effect of the minimized design effort is that focus can be spent on the remaining issues required to generate system-level assurance as opposed to implementation specification documentation.

Interface and Mechanism Descriptions

The system-level documentation must address all of the interfaces and mechanisms that have been modified in the process of composing the system. This includes not only the "glue" that is used to provide a cohesive system, but it must also include the new interfaces and mechanisms that are used to modify or extend the EPL COTS products.

In addition, those interfaces and mechanisms that are eliminated in the COTS product for use of that product in the system must be addressed in the system-level documentation. The impact of the elimination of the interface or mechanism on the overall working of the product and the product's role in the system needs to be

addressed. This can prove that the remaining product TCB is still intact and performing as expected in the new configuration.

Conclusion

In conclusion, evaluated products can successfully be used in the composition of trusted systems. The methodology described, and commonly used for typical systems, can be used for the design of trusted systems using evaluated COTS products with some modification. The TCB of the products must be analyzed to determine the redundancies and insufficiencies for the TCB of the entire system. The modifications, additions, and eliminations of both the TCB interfaces and TCB protection mechanisms must be documented in the system-level assurance documentation to compensate for any invalidation of the assurance evidence of the product.

References

- [1] Department of Defense, "Trusted Computer System Evaluation Criteria" (TCSEC), DoD 5200.28-STD, December 1985.
- [2] Modell, Martin E., A Professional's Guide to Systems Analysis, McGraw-Hill Software Engineering Series, McGraw-Hill Book Company, New York, 1988.
- [3] National Computer Security Center, "The Design and Evaluation of INFOSEC Systems: The Computer Security Contribution to the Composition Discussion", C Technical Report 32-92, June 1992.
- [4] National Computer Security Center, "Trusted Network Interpretation" (TNI), NCSC-TG-005, 31 July 1987.
- [5] National Computer Security Center, "Trusted Database Management System Interpretation of the Trusted Computer System Evaluation Criteria" (TDI), NCSC-TG-021, April 1991.
- [6] Page-Jones, Meilir, The Practical Guide to Structured Systems Design, Yourdon Press, Englewood Cliffs, New Jersey, 1988.
- [7] Pressman Roger S., Software Engineering, A Practitioner's Approach, McGraw-Hill Series in Software Engineering and Technology, McGraw-Hill Book Company, New York, 1987.

A PRACTICAL APPLICATION OF COMMERCIAL-OFF-THE SHELF PRODUCTS TO THE AUTOMATED INFORMATION SYSTEMS SECURITY OF THE NASA JOHNSON SPACE CENTER CONTROL CENTER COMPLEX

By
James W. Coyne
Senior Systems Engineer - AIS Security
Loral Space Information Systems
1322 Space Park Drive
Houston, TX 77058
(713) 335-6531

Abstract

From July through December 1992, the Mission Operations Directorate at Johnson Space Center engaged in a proof-of-concept activity intended to demonstrate the viability of constructing a control center comprised predominantly of off-the-shelf products/capabilities. As a part of this effort, satisfaction of the security requirements associated with a control center, through the use of native platform operating system and off-the-shelf products, was assessed.

Introduction

The Control Center Complex (CCC), is currently under development at the National Aeronautics and Space Administration (NASA) Lyndon B. Johnson Space Center (JSC)¹. It represents the consolidation of two previously existing development efforts, the Space Shuttle Mission Control Center (MCC) Equipment Replacement Project, and the Space Station Control Center (SSCC) Project. This consolidation was undertaken in order to reduce both the development and operations costs of these facilities. The CCC is envisioned to replace the current MCC, in the performance of launch, orbit, and re-entry operations for the Space Shuttle Program. It also replaces the planned SSCC, in the performance of the day-to-day ground-based operations for Space Station Freedom. The CCC implementation is a cooperative

effort of the NASA government, and development/operations contractor communities.

System Description

The Control Center Complex will consist of three Flight Control Rooms (FCR's), and several Multi-purpose Support Rooms (MPSR's). The FCR's, are referred-to as "Red", "White", and "Blue". The Red FCR, to be located in the present Space Shuttle Program MCC, will be responsible for the launch and re-entry phases of Space Shuttle missions. Extremely stringent availability requirements are associated with these highly-dynamic operations. The White and Blue FCR's will be responsible for day-to-day operations of Space Station Freedom and on-orbit shuttle operations. As these activities are considerably less dynamic than those conducted in the Red FCR, their availability requirements are correspondingly reduced. (An additional FCR, referred-to as "Gold", will be used for Independent Verification and Test (IV&T) activities. It could also be configured to support other programs.) This separation of activities will result in substantial savings, in development and operations costs.

Requirements Definition

The end of Department of Defense (DOD) involvement with the Space Shuttle Program, eliminated the need for the MCC to process DOD classified information. As a result, the requirement to comply with DOD automated information systems (AIS) security regulations and guidance was likewise eliminated. Relieved of the requirement to comply with DOD regula-

¹This project is being performed under contract number NAS 9-18300 for the National Aeronautics and Space Administration.

tions, expressions such as "C-2", "B-1", "System High", or "Multi-level Security" are rarely applied at JSC. Recognizing the vacuum this created, as well as the opportunities it presented, the JSC Mission Operations Directorate developed the MOD AIS Security Manual (or "Pink Book") [2]. This document advocates an AIS security policy based upon risk management, as opposed to absolute compliance with regulations.

The current MCC, and the CCC under development, are planned to process only Sensitive Unclassified information. NASA Headquarters and JSC security documents, as well as the MOD "Pink Book" divide this information into three "Sensitivity Levels" (SL's). They are defined as follows:

1. SL 1 - Automated information, applications, or systems which if altered, inaccurate, disclosed, or unavailable would have a minimal impact on NASA missions, functions, image, or reputation or could result in the loss of some tangible asset or resource.
2. SL 2 - Automated information, applications or systems which if altered inaccurate, disclosed, or unavailable would have a significant impact on NASA missions, functions, image, or reputation or would result in the loss of a major tangible asset or resource or severely impair the Agency's ability to fulfill a statutory obligation. Personnel or Privacy Act-protected information resources, to include crew biomedical data, fall into this category.
3. SL 3 - Automated information, applications or systems which if altered inaccurate, disclosed, or unavailable would have an irreparable impact on NASA missions, functions, image, or reputation or would result in the loss of a major tangible asset or resource or pose a threat to human life.

The mission-critical nature of the CCC resources have resulted in a designation of Sensitivity Level 3, for the facility.

The creation and maintenance of custom hardware and software, to implement AIS security measures is exceptionally expensive. The SSCC development baseline originally included several

tens of thousands of source lines of code to deliver the SSCC Security Control Subsystem. Several thousand additional lines of software was included in pending Change Requests, at the time the SSCC project was rolled-into the CCC project. This software was to have provided an enhanced security capability, over and above that present on the computational platforms. (The MCC Equipment Replacement Project had not yet baselined a line of code estimate, at the time of consolidation.)

One of the underlying objectives, of the "Pink Book's" authors, was that the security measures for MOD AIS, should be available as commercial-off-the-shelf (COTS) products. Vendor independence was a further goal. Through the acquisition of COTS products, which generally have vendor-provided maintenance available, capabilities can be acquired and sustained for a fraction of the cost of custom code.

In the interests of cost-effectiveness, during the consolidation, that resulted in the CCC Project, NASA challenged the CCC development contractor to implement as much of the CCC architecture as possible, with commercial-off-the-shelf (COTS) products. This challenge extended to the security architecture. Development of the security-related custom software was placed on-hold for a year, in order for the development contractor to evaluate products and perform proof-of-concept activities. As a result of this challenge, the CCC Project is providing the first real opportunity to engineer a completely COTS control center security architecture at JSC.

The first step in the proof-of-concept process was to define the specific requirements that the COTS products would have to meet. The information resources of the CCC fall predominantly into the SL 3 category, although some CCC information (e.g., crew biomedical and weather data) is at lower sensitivity levels. As such, security measures consistent with the protection of SL 3 resources are required. Instead of approaching this activity from the perspective of rigidly defined requirements, the following system-level functionalities, consistent with the policies contained in the MOD AIS Security Manual [2] for SL 3 systems, were identified:

1. Individual identification of system users via User ID's.
2. A user authentication, or password, function which provides the following capabilities:
 - a. The owner must be able to change the password at will.
 - b. The system(s) must force the owner to periodically change his/her password.
 - c. The passwords must have a minimum length of six characters.
 - d. The owner is prevented from re-using an expired password.
 - e. The system(s) should not display the password when it's entered.
 - f. Passwords should be encrypted during storage and during transmission across networks.
 - g. Passwords must be non-trivial
3. A means of limiting access to specific files, applications, and the operating system, to those individuals requiring it.
4. A "tunable" auditing capability, which provides the capability to selectively capture at least the following information for user sessions:
 - a. All successful and unsuccessful system accesses (login/logout).
 - b. All successful and unsuccessful writes, modifications, and deletions, at the file level.
 - c. All successful and unsuccessful reads of selected files.
 - d. All successful and unsuccessful actions of privileged (root) users.
5. An automated means of performing audit data analysis.
6. The capability to generate meaningful reports from the audit data.
7. Source and destination checking, for information transiting networks.
8. The ability to identify viruses, worms, and other malicious code.
9. The capability to "time-out" a user, following an established period of workstation inactivity. (Due to the operational nature of the CCC's activities, this must be accomplished in such a way as to not terminate the user session, and not stop processes executing on the user's behalf.)
10. A means of managing the Data Encryption Standard (DES) cryptographic keys employed by the Space Station Freedom Program.
11. A means of guaranteeing the integrity of data received from external sources (e.g., a cryptographic checksum or cyclic redundancy check).

Capabilities Assessment

The next step in the proof-of-concept process was to identify the products currently commercially available. This process was simplified by the fact that the CCC hardware architecture had already been established. The architecture is a hybrid consisting of Unix workstations and file servers, coupled with IBM ES9000 mainframe computers. These computational platforms will be configured as nodes on a fiber-optic (FDDI) network. The majority of the workstation security requirements were expected to be satisfied by their operating systems. Conversely, the IBM Resource Access Control Facility (RACF), which was bundled with each of the mainframes performs their security functions. While it is possible that other products would provide like capabilities, these products represent "sunk costs", and would have to be proven in some way deficient, to merit replacement. Therefore, our orientation is towards products that supplement these "native" platform capabilities.

Evaluation of these "native" capabilities was performed during a CCC proof-of-concept period referred-to as the Early COTS Platform (ECP) Project (more recently referred-to as CCC Delivery 1, Release 1). During this period numerous disciplines, in addition to AIS security, were evaluating COTS means of fulfilling their design goals, or the rehosting/re-use of existing software products. It was, therefore, an appropriate time to engage in similar evaluations of security functionality.

The IBM Multiple Virtual Storage (MVS) operating system, with RACF, has been evaluated by

the National Computer Security Center. Further, several IBM mainframe computers are currently in-use, in the MCC. For this reason, it was deemed unnecessary to exhaustively evaluate the security capabilities of the mainframe platforms. The capabilities of RACF are well-known and understood among the JSC community. The only requirements, among those previously mentioned, not met by RACF are:

1. The ability to identify viruses, worms, and other malicious code. (This is, to some extent, understandable. Although no computer is totally immune to malicious software, MVS viruses are extremely rare.)
2. The capability to "time-out" a user, following an established period of inactivity, in a manner acceptable to the MOD Operations Community. (IBM does provide a time-out feature. However, their implementation terminates the user's sessions, closes files, and terminates processes.)
3. A means of managing the Data Encryption Standard (DES) cryptographic keys employed by the Space Station Freedom Program. (This requirement was not expected to be met by the "native" platform capabilities.)
4. A means of guaranteeing the integrity of data received from external sources.

Several different manufacturer's workstations were included in the Delivery 1, Release 1 configuration. These included Sun Sparc 2 and IPX workstations and fileserver, running SunOS 4.2.C; Masscomp 6600 workstations, running Concurrent Real-time UNIX (RTU) 5.0; a Silicon Graphics Iris 2 workstation; running IRIX 3.3; and Digital Equipment Corporation DECstation platforms, running ULTRIX 4.2.3. All were exhaustively evaluated, by the development contractor's Independent Validation and Verification organization. The results of this evaluation are as follows:

1. None of the evaluated workstation platform/operating system combinations were able to satisfy the following requirements:
 - a. Prevention of the use of trivial passwords
 2. The DECstation workstations, with the ULTRIX enhanced security option enabled, met all of the security requirements, with the exception of those mentioned above
 3. In addition to the above exceptions, the Sun Sparc 2 and IPX workstations/fileserver, with the SunOS "C2" enhanced security option enabled, also failed to satisfy the following:
 - a. A user authentication, or password, function that enforces a minimum password length of six characters. (SunOS enforces a minimum length of five characters.)
 - b. A user authentication, or password, function that prevents the re-use of an expired password. (Re-use of the immediately previous password was demonstrated.)
 - c. A means of guaranteeing the integrity of data received from external sources.
 4. In addition to the above exceptions, the Masscomp 6600 workstations, running Concurrent RTU 5.0, also failed to satisfy the following:
 - a. A user authentication, or password, function that enforces a minimum password length of six characters. (RTU 5.0 enforces a minimum length of six characters, unless a "sufficiently robust" alphabet (upper and lower case characters) is used. When this occurs, it enforces a minimum length of only four characters.)
- b. An automated means of performing audit data analysis.
 - c. The ability to identify viruses, worms, and other malicious code. (This was not expected to be provided by an operating system.)
 - d. The capability to "time-out" a user, following an established period of workstation inactivity, in a manner acceptable to the MOD Operations Community.
 - e. A means of managing the Data Encryption Standard (DES) cryptographic keys employed by the Space Station Freedom Program. (This requirement was not expected to be met by the "native" platform capabilities.)

- b. A "tunable" auditing capability. Concurrent RTU 5.0 only provides a true audit capability for the root user. (However, during the evaluation, it was determined that sufficient information could be gleaned from the process accounting log to satisfy the auditing requirement.)
 - c. A means of guaranteeing the integrity of data received from external sources.
5. In addition to the above exceptions, the Silicon Graphics Iris 2 workstation, running IRIX 3.2, also failed to satisfy the following:
- a. No auditing capability was included with this operating system. (Unlike the Masscomp workstations, the process accounting log was not usable for this purpose. It is overwritten daily, and does not capture the date of the activity.) Subsequent discussions with Silicon Graphics vendor representatives indicate that an auditing capability is available with their Trusted IRIX/B operating system.
 - b. A means of guaranteeing the integrity of data received from external sources.

Satisfying Remaining Requirements

The results of the evaluation were clear. To varying degrees, all of the evaluated platforms met at least some of the identified Sensitivity Level 3 requirements. However, five of the requirements were not provided by the bundled capabilities of any of the evaluated platforms, including those with operating systems designed to satisfy Trusted Computer System Evaluation Criteria (TCSEC) C2 requirements. Therefore, those requirements will have to be satisfied with other products, or with custom-developed software.

For reasons previously discussed, the obvious preference is to acquire COTS products to satisfy these requirements, if such products exist. An exhaustive industry survey was undertaken, utilizing the resources of both the development contractor, and the MOD AIS Security Engineering Team (ASET), an organization chartered to support the MOD Computer Security Official in the development of security

policy, and to advance the introduction of new security technologies in MOD.

This process was simplified by the determination that the baseline CCC workstation will be the DECstation 5000/240 (augmented by DEC Alpha platforms, for computationally-intensive tasks. Given this knowledge, only products compatible with such an architecture needed to be evaluated. During this period, 55 different products, offered by 33 different vendors, were reviewed.

Several likely candidates for the DES key management system have been identified. As a result, this product will be acquired through competitive procurement. Unfortunately, for most of the remaining requirements, only single candidates were identified:

1. **Digital Equipment Corporation DECinspect** is a Network Security Product that includes:
 - a. The DECinspect Compliance Manager, which establishes and maintains a baseline of values for the security-related settings on ULTRIX and SunOS systems. It periodically checks file and directory permissions and account privileges. It performs password management, checking for minimum length and trivial/expired passwords. It performs network security (TCP/IP and NFS), and augments the operating system audit controls.
 - b. The DECinspect Intrusion Detector lets the security administrator define alarm thresholds for security-relevant system events. Intrusion Detector can be configured to track login failures, file access failures, and break-in audit events. It can also notify the security administrator of hostile activity, stop unauthorized processes and disable malicious accounts, and restore audit alarms if turned-off by a user or it's own processes if they terminate unexpectedly.
 - c. The DECinspect Security Reporting Facility centralizes security audit data analysis and reporting to a single platform. It allows the security administrator to view current and historical security information for a distributed network, and generate customized reports.

Also performs real-time reporting of security alarms. (Note: At the moment, this component is only available for VAX/VMS systems, although it will accept input from Compliance Managers and Intrusion Detectors resident on ULTRIX and SunOS platforms. As of this writing, this component is expected to be available, for Unix platforms, within six to eight months.)

These products are currently being evaluated for inclusion in the CCC architecture.

2. Cybersoft VFIND: The only Unix virus detection tool identified. This product is capable of locating Unix viruses (e.g., the AT&T Attack virus), worms (e.g., the Internet Worm), trojan horses and logic bombs. It also provides a removable media scanning capability that quarantines input media. This product was evaluated quite successfully, in parallel with the workstation evaluations, and will be purchased for the CCC workstation platforms. (No virus scanning products are known to be available for IBM mainframes.)
3. Digital Equipment Corporation DECdetect for ULTRIX V1.0: This product uses a cryptographic checksum, created using the MD4 algorithm by RSA Data Security, as a means for detecting modification of files. This product is being evaluated for inclusion in the CCC architecture, to augment the Kerberos capabilities included under the ULTRIX enhanced security option. (This is largely due to our understanding that the initial releases of the DEC Alpha's OSF-1 operating system does not support Kerberos.)

To date, no product has been identified which implements a workstation time-out, in a manner acceptable to the MOD Operations Community.

Conclusion

With the exception of the workstation time-out requirement, all of the CCC Sensitivity Level 3 requirements are expected to be met, without a single line of custom software having been written. While several of the above products are still under evaluation, if they perform as expected, their inclusion in the CCC architecture will be

pursued. As of this writing, we have every reason to believe that the CCC AIS Security Architecture can be accomplished almost exclusively through the use of off-the-shelf products.

Acknowledgments

The author gratefully acknowledges the review comments of Charlene Curtis, his NASA Task Monitor. This paper is largely the result of work conducted as part of the Control Center Complex Project, sponsored by the National Aeronautics and Space Administration, under Contract No. NAS 9-18300.

References

- [1] National Aeronautics and Space Administration, Lyndon B. Johnson Space Center *Automated Information Systems Security Plan*, JSC-23668, April 1989.
- [2] National Aeronautics and Space Administration, Lyndon B. Johnson Space Center, *Mission Operations Directorate Automated Information Systems Security Manual*, JSC 23982, October 1990.
- [3] National Aeronautics and Space Administration, Lyndon B. Johnson Space Center, *Mission Control Center (MCC) Level A Requirements, Revision C*, JSC-12804, November 1992.
- [4] National Aeronautics and Space Administration, Lyndon B. Johnson Space Center, *Space Station Control Center (SSCC) Level A Requirements, Revision B*, JSC-32069, June 1992.

CHOOSING AMONG STANDARDS FOR LOWER LAYER SECURITY PROTOCOLS

Wayne A. Jansen, Dale L. Walters
The National Institute of Standards and Technology
Technology Building
Gaithersburg, Maryland 20899

ABSTRACT

Within the last year, several security protocols pertaining to the lower layers of the OSI reference model have emerged from the international standardization process. These standards offer similar, but not identical, security services for the transport, network, and data link layers. This paper attempts to distinguish each security protocol from the others in terms of the subtle differences that may be important, but not obvious, to a potential user. It also provides guidelines concerning the environments for which each protocol is best suited. Based on the organizational protection policy and the information provided here, individuals can better select the security protocol most appropriate for their requirements.

1. INTRODUCTION

The Open Systems Interconnection (OSI) reference model [1] defines a seven-layer hierarchical model for communications. Each layer of the OSI reference model provides a service to the layer above and uses the services of the layer below. National and international standardization bodies use the model to develop specific protocols that fulfill the service specifications of a sub-layer or layer within the model. The OSI Security Architecture [2] is an addendum to the reference model that identifies security services suitable for OSI, and specifies where in the model these services may be offered. The Security Architecture identifies the following categories of security services:

- (a) Data Integrity,
- (b) Data Confidentiality,
- (c) Authentication,
- (d) Access Control, and
- (e) Non-repudiation.

Not every service is allowed at each layer of the reference model. Of the services allowed, none are mandated. Underpinning each service are one or more security mechanisms, such as encipherment, digital signature, and traffic padding. The Security Architecture provides the framework from which security protocols that employ specific security mechanisms may be developed to address a well-defined subset of services. The relationship between services and protocol layers given in the OSI Security Architecture is illustrated in Figure 1.

As Figure 1 shows, not all the security service categories are applicable to all layers. Non-repudiation prevents one protocol entity from falsely denying to another that information was transmitted or received. Non-repudiation is exclusive to the application layer where the communications abstraction (e.g., message handling) is meaningful to individuals. Data Integrity and Confidentiality services often provide the initial motivation for employing a security protocol. Data Confidentiality renders non-disclosure of communications, while Data Integrity renders fidelity of communications. The remaining services augment these services. Authentication ensures that claimed identities associated with protocol entities or data units are established and verified. Access Control ensures that only authorized entities gain access to information.

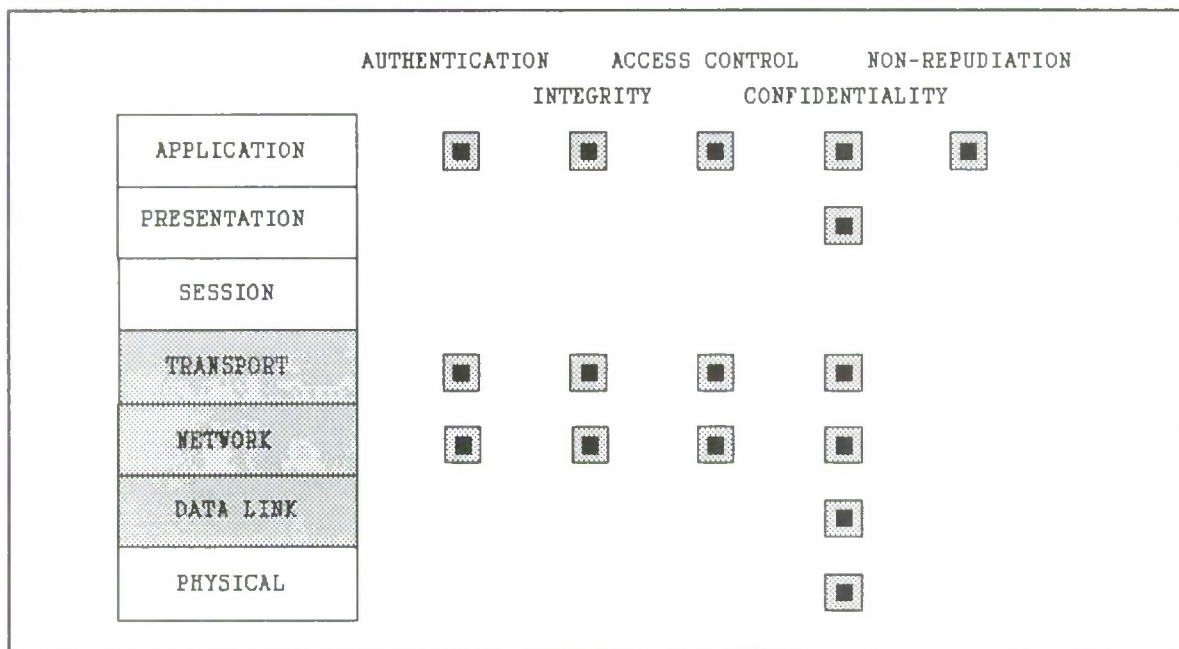


Figure 1: OSI Security Architecture

2. SECURITY PROTOCOLS

The lower layer security protocols considered in this paper occur at the transport, network, and data link layers of the reference model. They are respectively the following:

- (a) Transport Layer Security Protocol (TLSP) [3],
- (b) Network Layer Security Protocol (NLSP) [4], and
- (c) Secure Data Exchange (SDE) Protocol [5].

It is important to recognize that lower layer protocols are only one part of a much broader picture that includes security in the upper layer protocols, security management, existing or planned infrastructure, and organizational security policy. The OSI Security Architecture is aimed toward protection of communications between open systems and nothing more. Within the context of an organizational security policy, a specific security architecture may be designed from the allowable set of security services defined in the OSI Security Architecture. The design would most likely avoid duplication of service, and select from the available services only those required to meet the organizational security policy. An example of an organizational security architecture is the NATO OSI Security Architecture (NOSA) [6].

Clearly from Figure 1, the entire set of security services can be provided at the application layer. An organizational specific security architecture could be designed that would not require any lower layer security protocols. Therefore, the discussion in this paper is relevant to security architectures that have mixed upper and lower layer security service requirements, or rely solely on lower layer security services. Emphasis is also placed on security architectures that employ a single layer security protocol, exclusively of the others, to avoid duplication of service and the associated overhead involved.

A lower layer security protocol may distinctively contribute to the communications security between open systems [14]. One of the benefits of providing protection at a lower layer is that all application layer protocols can make use of the available services. That is, rather than incorporate security features into each application protocol, a common set of services offered through a lower layer security protocol can be managed and used in a consistent fashion. Because communications devices, such as bridges and routers,

are based on lower layer protocols, there is also an opportunity to incorporate companion security protocols within these devices, allowing a highly integrated implementation to be produced.

All three security protocols mentioned have many issues common to them, since they apply similar protection to exchanged data units. The first and perhaps most significant of these common issues is the reliance on a security association for protocol operation. A security association is a relationship established for use between communicating layer entities that identifies an agreed set of attributes required for secure communications. The attributes indicate the security services to be provided, the mechanisms to be used, and may include cryptographic material, algorithm identification, security labels, and other mechanism dependent parameters.

A security association may be established in several ways: by the communicating entities themselves, by another set of entities on behalf of the original communicating entities, or by some external means, such as manual dispatch. Despite how a security association is established, it is a prerequisite for proper operation of the lower layer security protocols. Associated with each lower layer security protocol is a means to establish security associations dynamically between communicating entities. Within the respective standards, it is referred to as either a key management protocol or security association establishment protocol.

Another issue common to the lower layer security protocols is that of algorithm independence. The lower layer security protocol standards are defined to be independent of the algorithms that underlie a security mechanism used in the protocol. This principle applies to both cryptographic and non-cryptographic algorithms. Implementors' groups, such as the Open Systems Environment Implementors' Workshop (OIEW), are expected to define suites of algorithms intended for use in commercial applications involving sensitive information. Ultimately, the determination of whether the algorithms used in an implementation are suitable for the sensitivity of the information being protected must be made by the user organization. For Federal systems, the U.S. Government will decide what algorithms to use, and may mandate unique or classified algorithms for sensitive, but unclassified, information.

A related issue is whether the implementation is correct and effective for the intended environment of use. This second aspect is often identified in the U.S. with the trust requirements contained in the DOD Trusted Computer System Evaluation Criteria [7]. The standards for the lower layer security protocols mentioned are silent on this point. However, since the protocols are simple and involve only security functionality, they are likely candidates for incorporation into the trusted computing base for an implementation. As with the choice of algorithms, this issue is left to organizational policies for resolution.

2.1 The Transport Layer Security Protocol (TLSP)

TLSP is an International Organization for Standardization (ISO) International Standard (IS). TLSP extends the services of either the OSI connection oriented (ISO 8073) [8] or connectionless (ISO 8602) [9] transport layer protocols to support the following security services defined in the OSI Security Architecture:

- (a) Data Integrity,
- (b) Data Confidentiality,
- (c) Data Origination Authentication, and
- (d) Access Control.

In the OSI reference model, the transport layer has the sole responsibility to provide reliable end-to-end communications between peer end-systems. TLSP services, therefore, nicely complement those provided by the transport protocols. In contrast to the other lower layers, a transport layer protocol has end-to-end significance, guaranteeing that operations involving transport protocol entities take place only between end-systems. This feature distinguishes TLSP from other security protocols with regard to security architecture requirements.

TLSP should be viewed as an extension or addendum to the referenced transport protocol standards, than as an independent layer protocol. TLSP functionality is situated near the bottom of the transport layer as illustrated in Figure 2. It consists of a simple encapsulation/decapsulation protocol that protects transport protocol data units (PDUs) within a cryptographically secure envelope.

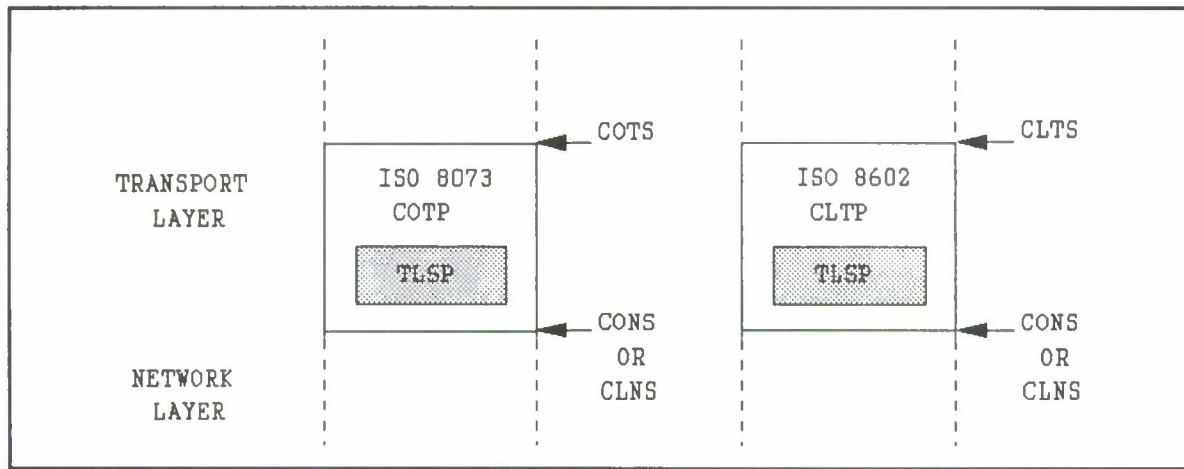


Figure 2: TLSP within the Transport Layer

TLSP used with ISO 8073 can protect PDUs on either a per connection basis or on an end-system oriented basis, depending on the granularity specified by the security association. If a TLSP entity operates on a per connection basis, a user can distinguish a different level of protection for each connection. For example, one connection can use Data Confidentiality and Data Integrity services, another Data Integrity alone, and yet another, no protection services at all. This is a feature unique to TLSP and can be beneficial in many applications. Multiple transport connections of differing levels of protection can be multiplexed over the same instance of communications at the network layer. The end-system oriented alternative applies the same level of protection collectively to all PDUs, for all connections of the same class between two end-systems.

TLSP used with ISO 8602 protects the confidentiality and integrity of individual transport PDUs independently of one another. A significant difference between TLSP used with ISO 8602 or ISO 8073 is that although the integrity mechanism is the same, ISO 8602 supports only a Connectionless Integrity service (without Recovery), while the connection oriented mechanisms of ISO 8073 allow a more robust service, Connection Integrity with Recovery.

The TLSP standard identifies other security functions on which it depends. These functions include key management, security management, and cryptography. Key management is needed to establish and maintain security associations; security management is needed for event reporting and auditing; and cryptography (i.e., employment of specific cryptographic algorithms) is needed to support Data Confidentiality, Data Integrity, and Authentication services, as appropriate. In a distributed application, these security functions may reside within the local end-system or be disbursed among several end-systems, using OSI communications to carry out their tasks.

One form of key management is described in an amendment to the TLSP standard for a security association protocol [10]. The amendment specifies an optional extension to TLSP for establishing a security association and rekeying the association during its life time. Through this protocol an additional security service of Peer Entity Authentication may be provided. The security association protocol is defined for both ISO 8602 and ISO 8073, but it will most likely be used only with ISO 8073 since the other lacks the robustness needed for many environments. The security association protocol is considered by many experts to be an interim step until a broader solution emerges. Both key management and security management are envisaged to operate as distributed applications that use appropriate OSI application layer protocols.

2.2 The Network Layer Security Protocol (NLSP)

NLSP is an ISO Draft International Standard (DIS) for the network layer. NLSP is defined to operate with either the connection oriented (ISO 8208) [11] or connectionless (ISO 8473) [12] network protocol (CONP or CLNP), at or near the top of the network layer. Figure 3 illustrates these protocols. NLSP applies a simple security encapsulation/decapsulation procedure similar to TLSP. NLSP may reside in either end-systems or intermediate systems. The capability of NLSP to be used within intermediate systems, either between two intermediate systems or between an intermediate system and an end-system, distinguishes it from TLSP.

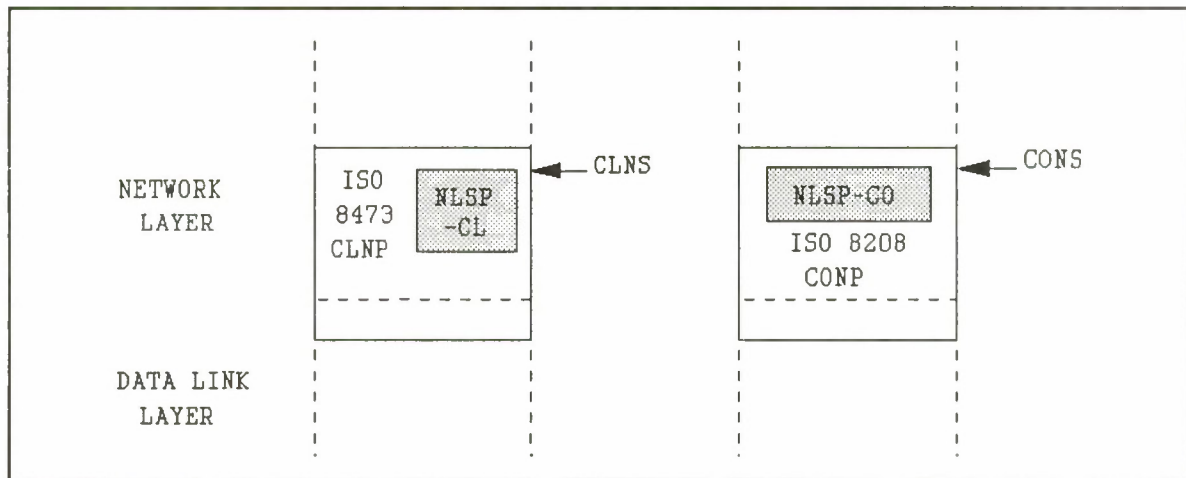


Figure 3: NLSP within the Network Layer

NLSP has two modes of operation corresponding to either a connectionless or connection oriented service interface. The abstract interfaces and security services provided are dissimilar to the extent that the modes of operation should be viewed and as two distinct protocols. Connectionless NLSP (NLSP-CL) supports the following security services:

- (a) Connectionless Integrity,
- (b) Connectionless Confidentiality,
- (c) Traffic Flow Confidentially
- (d) Data Origination Authentication, and
- (e) Access Control.

The connection oriented NLSP (NLSP-CO) supports the following security services:

- (a) Connection Integrity without Recovery,
- (b) Connection Confidentiality,
- (c) Traffic Flow Confidentially,
- (d) Peer Entity Authentication,
- (e) Data Origination Authentication, and
- (f) Access Control.

In either mode of operation, CO or CL, the Data Confidentiality service is essentially equivalent. The same can be said for Access Control and Data Origination Authentication, but not Data Integrity. NLSP-CL integrity is done on a PDU basis and is unable to detect additions or deletions of PDUs; only modification of a PDU is detectable. NLSP-CO can maintain an integrity sequence number space to protect against insertions and deletions. Many commercial applications need the Connection Integrity service.

Traffic Flow Confidentiality is a service provided by either mode of NLSP to deter traffic flow analysis. The benefits of this feature have long been known by the military community, while the commercial community is just beginning to see a need for its use. Support for traffic padding allows for identical PDU sizes and/or constant PDU streams to be generated. Note that full Traffic Flow Confidentiality is not possible at the network layer, since at this layer the introduction of spurious traffic cannot be properly regulated. Nevertheless, the partial service may be adequate for some organizations.

An aspect of Traffic Flow Confidentiality unique to NLSP is address hiding. This feature counters traffic flow analysis by hiding the end-system addresses of a PDU sent to a gateway where protection is provided. Address hiding is useful in both military and commercial systems where one advertises the NLSP gateway, but not the end-systems connected to the gateway. Traffic analysis would reveal that PDUs came from an organization, but the specific end-system could not be determined.

For an architecture that is connection oriented at both the transport and network layer, NLSP-CO could be an appropriate solution since it provides the Connection Oriented Integrity service, albeit without Recovery. This is especially relevant for the class 0, transport protocol where there are no transport sequence numbers to use for recovery. With other classes of transport there would be some redundancy as both the transport protocol and NLSP-CO supply a sequence number. Such duplication should be avoided if possible.

NLSP used within an end-system has some similar characteristics to TLSP. This is understandable since many of the security mechanisms used are the same, and both employ a similar PDU format. Their functionality within the OSI model is close to one another. TLSP resides near the bottom of the transport layer and NLSP is near the top of the network layer. These factors contribute to the subtleness of the characteristics that distinguish them.

As described for TLSP, a security association establishment protocol is also incorporated as an optional part of NLSP. NLSP security associations are formed between protocol entities residing at either end-systems or intermediate-systems. Since NLSP-CL lacks the robustness needed for many environments, the security association establishment protocol is unlikely to be used with it. For NLSP-CO, Peer Entity Authentication is done initially as part of setting up an association, and thereafter, anytime the peer entities require reauthentication. The level of granularity for NLSP security associations is high compared with TLSP security associations that are established on a per connection basis. In some applications, this could be a concern if only a small percent of the communications requires protection, but due to NLSP granularity limitations the same level of protection is applied to all.

2.3 The Secure Data Exchange (SDE) Protocol

The SDE protocol is one part of a multi-part Institute of Electrical and Electronics Engineers (IEEE) Standard for Interoperable Local Area Network (LAN) Security (SILS). Note that the term "LAN" as used here refers strictly to the bottom two layers of the OSI reference model and not to client-server relationships formed over a local network. SDE was developed to address the security of local area and metropolitan area networks (LANs and MANs), and to be compatible with the existing IEEE 802 and OSI architectures. The SDE protocol gained ballot approval as an IEEE Standard in the latter part of 1992.

The SDE protocol provides services that allow the secure exchange of data at the data link layer. Similar to the other security protocols mentioned, SDE applies a simple security encapsulation/decapsulation procedure. For IEEE LANs and MANs, the data link layer is modeled as two sublayers: the Logical Link Control (LLC) sublayer, and the Medium Access Control (MAC) sublayer. SDE is modeled as part of the LLC (IEEE 802.2) [15], and therefore, can provide connectionless security services across different types of media, represented by the various MAC sublayer protocol standards. Figure 4 illustrates the SDE protocol.

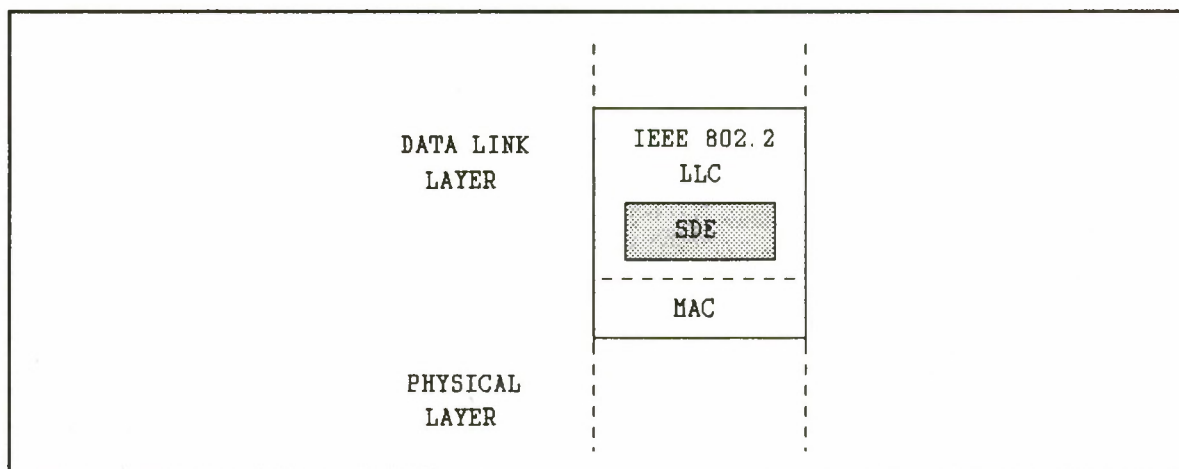


Figure 4: SDE within the Data Link Layer

The SDE protocol supports the following set of security services:

- (a) Connectionless Integrity,
- (b) Connectionless Confidentiality,
- (c) Data Origination Authentication, and
- (d) Access Control.

It is important to note that the OSI Security Architecture prescribes only the Data Confidentiality service for the data link layer. For this reason, it is unlikely that the IEEE standard will become an ISO standard in the near future, as other IEEE 802 standards have done. On the other hand, its location at the data link layer allows the SDE protocol to provide a desirable set of security services to any suite of protocols operating above. Furthermore, it is designed to protect multicast as well as point-to-point communications.

Because of the NLSP and TLSP alternatives, the SDE protocol is usable only in limited situations. If data is intended only for a LAN environment then the SDE protocol is a viable solution. This includes situations where LANs are connected via bridges. SDE offers a single solution for both local and remote bridging environments, and can be used to protect both end-station and bridge control traffic. However, if there is any thought of the data traffic leaving the LAN environment then another security protocol is needed to protect the data.

Key management is required to support the SDE protocol. Another part of the SILS standard addresses this topic. However, the status of the key management protocol standard is much less mature than the SDE protocol. Unlike the security association establishment protocols described earlier for TLSP and NLSP, the SILS key management protocol is being developed as an application layer protocol, than at the data link layer. The application layer has distinct advantages for specifying security information objects to be exchanged, allowing this protocol to be used by any lower layer security protocol.

3. ENVIRONMENT

Many strategies for applying protection are possible using the security protocols that have been discussed. At the transport layer there are two scenarios: connectionless for use with ISO 8602 and connection oriented for use with ISO 8073. At the network layer there are four scenarios. The first two scenarios concern the connection and connectionless modes within an end-system, and the third and fourth cover their use in intermediate systems. Finally, SDE can be used within a LAN/MAN environment.

Although many different approaches are possible, only some of them are likely to have wide spread applicability. Each has advantages with respect to a particular environment that make it an attractive

approach. Several highly pertinent approaches are considered below. For simplicity they are organized according to a simple principle: whether security services are to be provided by the information systems (i.e., end-systems exclusively) or by the network communications (i.e., intermediate-systems exclusively).

3.1 Protected Information System Environment

3.1.1 Transport Layer Approach

TLSP used with the connection oriented transport protocol (ISO 8073) is appropriate for security domains that are end-system oriented in their administration of security mechanisms such as encipherment, and require accountability and protection of data on a transport connection basis. TLSP may be especially relevant to agencies and organizations who have determined that the required means of providing reliable end-to-end communications is through the class 4 transport protocol. This is due to the tendency to administer such systems from a transport standpoint, and the likelihood of extending this perspective to incorporate security services. The United States Government OSI Profile [13] mandates ISO 8073 class 4 over the connectionless network service as the sole means of interoperability among U.S. GOSIP compliant computer systems.

ISO 8073 may operate over either connectionless or connection oriented network service. If TLSP is used with ISO 8073, one can realize Connection Confidentiality and Connection Integrity with Full Recovery. An added benefit is that protection can be applied to only those connections that require it, and to the extent needed. TLSP also allows the possibility to use a robust integrity algorithm for the Data Integrity service, as a preferable alternative to the transport checksum function.

TLSP is normally deployed within each protected end-system with a supporting cryptographic facility. Although the cryptographic facility is an overhead, many application layer security protocols, such as secure mail, also require a cryptographic facility in each end-system. Thus, there may be an opportunity to share the facility and prorate the associated overhead across all applications.

3.1.2 Network Layer Approach

It is also possible to employ either variant of NLSP within each end-system as with TLSP. One possible benefit to using NLSP is that the implementation of a security protocol at the network layer may attain a higher level of assurance than one at the transport layer. NLSP used with the connectionless network protocol is much simpler than TLSP with ISO 8073, since the former combination is connectionless and requires no state information to be retained. NLSP-CO is designed to be easily isolated from the connection oriented network protocol. NLSP-CO is envisaged to be used with either class 0 or 2 of ISO 8073 and provide a common approach for ISO 8208 packet switched networks currently deployed. Either approach would be appropriate for a classified or highly sensitive information processing environment. The abilities to provide Traffic Flow Confidentiality and address hiding also lend itself to this type of application.

3.2 Protected Network System Environment

3.2.1 Connectionless Network Approach

NLSP can be use exclusively within intermediate systems to form a protected network system. One benefit of this use is by restricting security services and associated cryptographic devices to intermediate systems (e.g., external Wide Area Network (WAN) gateways) the expense may be lower than one that depends on an end-system approach. NLSP-CL with ISO 8473 (CLNP) is considered a widely applicable means of protection for this environment. One drawback may be that since integrity sequence numbers are not maintained by either protocol, only the Connectionless Integrity service can be provided -- not Connection Oriented Integrity.

With such an approach, security outside the intermediate systems would be a concern. This includes the situation where an intermediate-system gateway couples a WAN to a LAN of connected end-systems (e.g., workstations). If the LAN environment can be sufficiently protected, then this approach is appropriate. However, in office buildings where the LAN may be shared with, or accessible by other organizations, this approach may allow compromise, and therefore, would be unacceptable. In situations as these, using SDE to protect the LAN communication traffics nicely complements the gateway protection.

3.2.2 LAN Approach

The SDE protocol is appropriate if communications are limited entirely within a LAN environment. The SDE protocol may be used in each station attached to a LAN or restricted to bridge devices used between geographically dispersed segments. SDE is presently the only security protocol standard suitable for protecting communications between bridged LANs. Use of the SDE protocol depends on a variety of factors including the security policy of the organization, the size and topology of the LAN, the perceived threats, and the value of the information assets. Because of the alternative approaches of using TLSP or NLSP, consideration should be given to future requirements since many networks start out local and then expand to the wide area.

4. INTEROPERABILITY

Because of the range of possible security solutions, the question of interoperability arises. Security and interoperability are somewhat opposing ideas. To some degree security is gained by not being interoperable. Nevertheless, security is ultimately achieved through the effectiveness and correctness of the security mechanisms used, including the suitability of the underlying algorithms.

Ideally, the lack of interoperability should be limited to only those aspects on which security depends. For that reason, lower layer security protocol standards are specified to be independent of the algorithms used for encipherment, signature, and other security mechanisms. Therefore, two implementations of the same protocol may not be interoperable with regard to the suites of algorithms supported or the schemes employed for security association management. It is expected that a limited number of algorithm suites will be supported in products. One can envision several broad categories: classified government, sensitive government, commercial, and private.

The existing security infrastructure normally dictates the strategy for interoperability. A security infrastructure includes the communications architecture, the security architecture design, security management, and supported suites of algorithms. A security protocol must be incorporated into a security infrastructure to be useful. For example, the security protocol that one chooses often revolves around the communications architecture for a community of interest. If the community of interest requires class 4, ISO 8073 over CLNP in its communications architecture, then an NLSP-CO approach should not be used. The choice would be restricted to a compatible method such as using either TLSP or NLSP-CL. Similarly, if communications are needed with organizations that favor NLSP-CO in gateways, then that requirement would dictate the approach to be employed.

5. SUMMARY

It should be clear from the discussion that no single lower layer security protocol solution is appropriate for all possible environments. A distinction of whether to administer security from an information system or communications network perspective (i.e., end-system versus intermediate system orientation) nearly evenly divides the options that users must consider. The requirement to participate with a specific community of interest may further restrict the choices, or become an additional alternative to be supported. Similarly, an existing communications infrastructure may dictate the choice of solution. Through market forces some consolidation may be accomplished. The final choice for a solution requires a series of tradeoffs matching the requirements of an organization's security policy to the strengths and weaknesses of the security protocols considered.

REFERENCES

- [1] ISO IS 7498, Information Processing Systems - Open systems Interconnection - Basic Reference Model, 1984.
- [2] ISO IS 7498/2, Information Processing Systems - Open Systems Interconnection - Basic Reference Model - Part 2: Security Architecture, 1988.
- [3] ISO/IEC IS 10736, International Standard - OSI Transport Layer Security Protocol, December, 1992.
- [4] ISO/IEC DIS 11577, Draft International Standard - Information Technology - Telecommunications and Information Exchange Between Systems - Network Layer Security Protocol, November, 1992.
- [5] IEEE 802.10B, IEEE Standard for Interoperable Local Area Network (LAN) Security (SILS), Part B - Secure Data Exchange, September, 1992.
- [6] NATO OSI Security Architecture, version 3.1, TSGCEE Sub-Group 9, September, 1988, NATO Unclassified.
- [7] DOD 5200.28-STD, DOD Trusted Computer System Evaluation Criteria, August 1983.
- [8] ISO IS 8073, Information Processing Systems - Open Systems Interconnection - Connection Oriented Transport Protocol Specification, 1988.
- [9] ISO IS 8602, Information Processing Systems - Open Systems Interconnection - Protocol for Providing the Connectionless Mode Transport Service, 1987.
- [10] ISO/IEC JTC1/SC6 N7598, International Standard - Transport Layer Security Protocol - Amendment 1 - Security Association Protocol, December, 1992.
- [11] ISO/IEC IS 8208, Information Processing Systems - Data Communications - X.25 Packet Layer Protocol for Data Terminal Equipment, 1990.
- [12] ISO/IEC IS 8473, Information Processing Systems - Data Communications - Protocol for Providing the Connectionless-mode Network Service, 1988.
- [13] Government Open Systems Interconnection Profile (GOSIP), Federal Information Processing Standard (FIPS) 146-1, National Technical Information Service, April, 1991.
- [14] Paul Lambert, The Lowdown, on Lower Layer Security Protocols, Proceedings of the Sixth Annual Computer Security Applications Conference, December, 1990.
- [15] IEEE 802.2, Information Processing Systems - Local Area Networks - Part 2: Logical Link Control, 1989.

**MESSAGE HANDLING SYSTEMS
(X.400)
THREATS, VULNERABILITIES, AND COUNTERMEASURES**

**Michelle J. Gosselin
The MITRE Corporation
202 Burlington Rd
Bedford MA 01730
(617) 271-7737
mgoss@mitre.org**

INTRODUCTION

With the proliferation of computing systems made available from a multitude of vendors, communication between different organizations, and even within the same organization, could be difficult without the presence of standard communication protocols. One standard for message handling systems is described in the 1988 International Telegraph and Telephone Consultative Committee (CCITT) "Data Communication Networks Message Handling Systems, Recommendations X.400-X.420" (X.400). Many organizations are standardizing on the use of X.400 and are integrating an X.400 MHS into their existing networks. Since X.400 will soon become a prevalent method of exchanging messages, organizations not currently doing so should consider integrating X.400 into their networks.

While providing a standard messaging environment, however, the integration of an X.400 MHS into an existing network may introduce security flaws into that network. To protect themselves and their information from the exploitation of these security flaws by both malicious and nonmalicious entities, organizations should investigate the security issues involved in integrating X.400. Depending on the value of their data, organizations might also take steps to eliminate or lessen these security flaws.

In support of the need to protect the systems in which X.400 capabilities were integrated and to protect the data exchanged via X.400, an analysis was conducted to investigate the threats and vulnerabilities introduced by X.400 and to identify possible countermeasures to these threats and vulnerabilities. The results of the analysis are reported in this paper.

BACKGROUND

In preparation for the presentation of the analysis results, the paper discusses basic MHS and security concepts as well as the scope of and approach taken in conducting the analysis.

MHS Concepts

Figure 1 depicts the functional model of an MHS. The heart of an MHS, the message transfer system (MTS), relays messages within the MHS so that they can be delivered to the appropriate user.

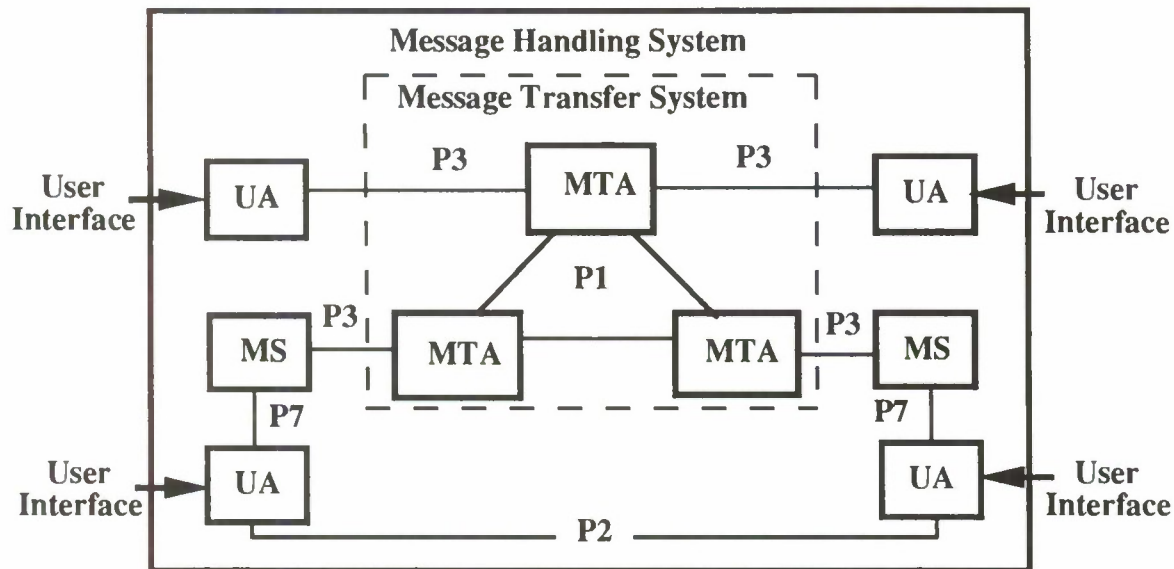


Figure 1. MHS Functional Model

A message transfer agent (MTA), a functional entity within the MTS, takes part in the relaying of messages by transferring messages to other MTAs. MTAs transfer messages to each other via the message transfer protocol (P1). MTAs also accept messages from and deliver messages to components that are external to the MTS via the MTS access protocol (P3).

A user agent (UA), an MHS component external to the MTS, provides a mail interface to a user that allows the user to gain access to the MHS to send messages to other users. A UA can access the MTS via P3.

A message store (MS), an optional MHS component, accepts delivery of messages on behalf of a user and retains them for subsequent retrieval by that user's UA. The MS also provides indirect message submission and message administration services to the UA. A UA accesses an MS via the MS access protocol (P7).

The higher level protocol that provides messaging between users through UAs and makes use of P1, P3, and P7 is the interpersonal messaging protocol (P2).

Not depicted in the functional model, but included in the recommendations, is an access unit (AU). An AU links another communication system (e.g., a physical delivery system or the telex network) to the MTS. Security issues relating to AUs were not addressed.

Although the functional model depicts the UA, the MS, and the MTA as being distinct entities, it is possible for them to be physically co-resident.

Security Concepts

The term "threat" bundles together several related concepts. According to dictionary definitions, a threat can be an expression of intent to cause harm (e.g., a person threatens to perform a burglary). A threat can also be an action taken to cause harm (e.g., the actual burglary), or the undesirable outcome that results from a sequence of actions taken to cause harm (e.g., loss of assets).

A vulnerability is the property of being open to attack or damage (i.e., an undesirable outcome). For instance, an unlocked door makes a house vulnerable to the threat of burglary.

A countermeasure is a feature that reduces or eliminates the possibility of exploiting a vulnerability to cause an undesirable outcome from occurring. For instance, a deadbolt lock makes a house less vulnerable to the threat of burglary.

A number of general threats exist for any service that is to be integrated into a network (where "service" includes OSI services, other networking software, and other applications). The three major types of undesirable outcomes an attacker might seek to achieve are modification, disclosure, and denial of service. Modification results in the unauthorized changing, deleting, or adding of data. When a subject reads data without proper authorization, disclosure occurs. Denial of service results from the unavailability of a feature or system.

A variety of threat actions can cause each of these undesirable outcomes. These threat actions include the masquerading of a subject as some other subject, the resequencing of data, and the false denial by a user of having either sent or received data (repudiation).

Scope and Approach of Security Analysis

The scope of the effort was limited to investigating those security issues that pertain to the protocols and services defined in the 1988 X.400 series of recommendations. The areas considered are depicted in figure 2 by the greyed boxes and the dashed line between them (representing the P1, P2, P3, and P7 protocols.)

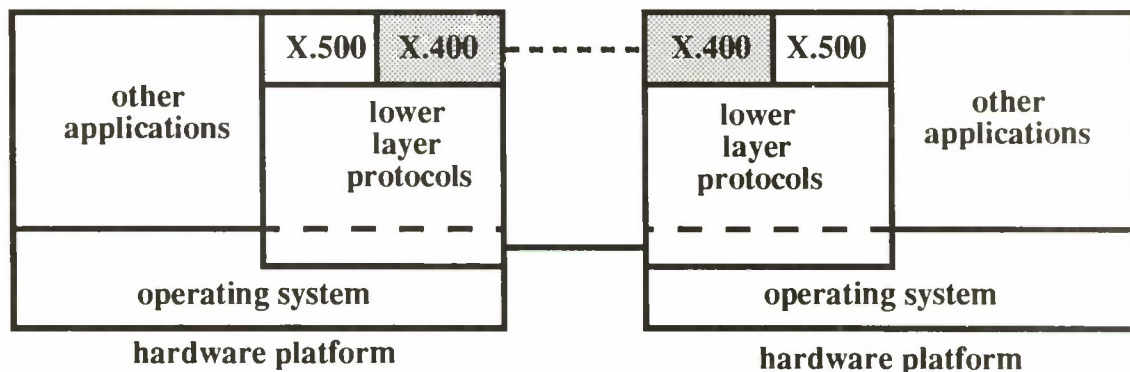


Figure 2. Scope of Analysis

The analysis did not consider risks relating to the operating system (except to a minimal extent), the hardware platform, X.500, other applications, and the lower layer networking protocols. Networking vulnerabilities not addressed include, for example, eavesdropping. However, there are no new networking related vulnerabilities introduced with the use of X.400 that do not exist with current messaging.

The analysis also assumed that classified data was not present within the environment targeted for the introduction of an X.400 MHS. Although potentially beneficial, a covert channel analysis and an exploration of mandatory access control violations were not performed given the fact that time was limited and these types of studies are not required for the unclassified environment.

Other vulnerabilities not considered were those related to indirect users since indirect users need not be configured into an MHS environment. Indirect users include those users who require physical, rather than electronic, delivery of their messages, or users that employ telematic services to receive, for example, voice mail.

The approach taken in performing the security analysis was to identify where an implementation could potentially create vulnerabilities once integrated into an existing network. The analysis was conducted at two levels. First, threats and vulnerabilities that could arise with the introduction of any new service were analyzed. These threats and vulnerabilities were derived from Information Processing Systems - Open Systems Interconnection - Basic Reference Model - Part 2: System Architecture (ISO-7498-2) and the Trusted Computer System Evaluation Criteria (TCSEC).

The second level of the approach was to identify threats and vulnerabilities specific to MHS by thoroughly searching CCITT recommendations X.400 through X.420 for possible security issues. These recommendations encompass all aspects of message handling elements of service. Individual implementations have not yet been reviewed; however, we expect that specific implementations will address many of the identified vulnerabilities.

SECURITY ISSUES SPECIFIC TO X.400

Now that background information has been presented, the security issues specific to X.400 can be discussed. The issues can be categorized according to the type of threat that they pose: modification, disclosure, denial of service, masquerade, resequencing, or repudiation. Each of these categories is discussed below.

Modification

As messages travel through the MHS, messages could be modified or destroyed without authorization. In addition to the modification of messages, routing information (the information required to get the message from the originator to its destination) could be corrupted.

Both messages and routing information can be stored within an MHS. The inadequate protection of the storage of any information is a vulnerability that, if exploited, could lead to unauthorized modification.

To adequately protect this stored information, the operating system where the information is located must be

capable of performing discretionary access control. Also, the information must be recognized as an object by the operating system (OS) so that the OS can use discretionary access control (DAC) mechanisms to protect the information. The object must also have the appropriate ownership and permissions associated with it. A group of messages can be contained in one OS object if all messages are owned by one user. However, multiple messages owned by several users cannot be contained in one OS object since operating systems usually perform DAC at the object level.

In order to provide this countermeasure, the MHS administrator must establish and maintain the permissions on administrator-owned objects containing messages and message handling information. The mail administrator also must initialize the permissions on user objects that are message handling related (e.g., individual mail boxes containing messages). Once initialized, the user must maintain these permissions.

Disclosure

As with unauthorized modification, there are also various forms of unauthorized disclosure. Loss of confidentiality occurs when the content of a message is captured and read by users for whom the message was not intended. By reading message header information that may not be protected, an MTS-user can detect who authored a particular message which would result in loss of privacy concerning authorship. Misappropriation occurs when messages are delivered to the wrong MTS-user, either through misuse or errors. Also, message traffic can be analyzed to ascertain information. (Pizza shops in the Pentagon area know important events are transpiring when the number of requests for delivery sharply increase.)

There are a number of vulnerabilities relating to unauthorized disclosure. As described previously, unauthorized access can be gained to stored messages to either read or modify the messages. Other vulnerabilities relate to distribution lists, the *alternate recipient allowed* argument, the *recipient reassignment allowed* argument, and blind carbon copy lists.

Distribution Lists. Distribution lists, provided through directory services (X.500), identify a group of people that have common interests so that messages can be easily sent to all individuals interested in a particular topic. A distribution list contains directory names and possibly other distribution lists.

Since distribution lists may undergo many changes and may be lengthy, the originator of a message using a distribution list may be misinformed as to the actual list membership. Nesting of distribution lists (a list within a list) adds to the confusion. If the originator is misinformed about the list membership, a message could be sent to a recipient whom the originator did not know was on the distribution list.

As one possible countermeasure, the X.500 *list request* operation could automatically report the distribution list membership to the originator. However, in addition to requiring modification of the OSI code, reporting the distribution list could be cumbersome for the user. As stated previously, distribution lists can be lengthy, and, when the message content is not sensitive in the originator's opinion, the originator may not want to see every name on the distribution list.

As an alternative countermeasure, a warning could be issued by the MHS administrator to users that this vulnerability exists and that they should use either the *list request* operation or the *distribution list expansion prohibited* argument when they send information that they consider sensitive.

Alternate Recipient. When a primary recipient cannot be determined from the information provided by the originator, the *alternate recipient* feature allows a destination MTA to deliver a message to an alternate recipient designated by that MTA. For this feature to work, the *alternate recipient allowed* argument would have to be specified by the originator during message submission, and the destination MTA would have to have an alternate recipient designated. The problem with this feature is that the originator does not know who the alternate recipient is, and the originator may not want the alternate recipient to receive the information.

To eliminate this vulnerability, the originating MTA could automatically change the *alternated recipient allowed* argument, if supplied, to *alternate recipient prohibited*. If automation is not possible or desired (since at times it is beneficial to have an alternate recipient for critical messages), users could be warned by the MHS administrator to set the *alternate recipient prohibited* argument during submission of messages they consider sensitive. The default specified in the recommendation is to prohibit an alternate recipient.

Recipient Reassignment. The *recipient reassignment* feature allows users to instruct the MTS to redirect incoming messages addressed to them. The intended recipient specifies to whom the messages are to be redirected, without the knowledge or approval of the originator. With this feature, the intended recipient never receives the message. For this feature to work, the *recipient reassignment allowed* argument would have to be specified by the originator during message submission, and the intended recipient would have to have an alternate recipient designated.

As with the *alternate recipient* feature, the originating MTA could automatically change the *recipient reassignment allowed* argument, if supplied, to *recipient reassignment prohibited*. If automation is not possible or desired, users could be warned by the MHS administrator to set the *recipient reassignment prohibited* argument during submission of sensitive messages. The default specified in the recommendation is to allow recipient reassignment. This default should be changed so that this feature is not invoked without the originator specifically allowing it.

Blind Carbon Copy Recipients. The *blind carbon copy* (BCC) feature allows a user to specify recipients of a message that direct recipients and carbon copy recipients of the message do not see. The concern with this feature involves replies to messages that have BCC recipients. A user can globally reply to a message and have the reply automatically sent to the originator and all recipients of the message. The recommendation does not state that BCC recipients should not receive any replies to a message. Therefore, depending on the implementation, a reply could be sent to someone without the knowledge of the replier.

To eliminate this vulnerability, any MTA involved in delivering a message to a recipient should remove the BCC list from both the message being delivered and any records stored internally.

Denial of Service

Denial of service results from a breakdown in the network, the failure of an MS or MTA, or the flooding of the MTS with messages. Any of these problems prevents the delivery of messages.

The user's ability to specify the priority of a message creates a vulnerability that, if exploited, could cause a denial of service. The priority of a message can be either urgent, normal, or nonurgent. This is different from an importance indication which informs the recipient whether or not the originator considers the message an important one that should be read as soon as possible. An urgent message may be processed by the MTS more quickly than a normal or nonurgent message.

Since the MTS may process urgent messages more quickly than other messages, a user could flood the MTS with urgent messages and delay the processing of normal or nonurgent messages.

As a countermeasure to this problem, the privilege to set the priority on a message should only be granted to the MHS administrator. A standard MHS-user should not be granted this privilege. To prevent the MHS-users or MTAs of one MHS community from flooding another MHS community with urgent messages, a threshold on the number of urgent messages processed could be established within an MHS community. Once the threshold was reached, the MTA could change the priority of the message from urgent to normal. MTAs would have to be modified to check on access rights and thresholds.

These countermeasures potentially pose a loss of functionality to the MHS-user who has a legitimate need to set a high priority on a message.

Masquerade

There are five forms of masquerade that could take place: impersonation of an MTS-user to an MTA, impersonation of an MTA to an MTS-user, impersonation of an MTA to another MTA, impersonation of an MS to a UA, and impersonation of a UA to an MS.

Vulnerabilities that could be exploited to cause masquerading are related to the originator/recipient (O/R) name supplied with many operations, the credentials given during an initiation of a connection (a bind), and the register operation.

O/R Name. An O/R name comprises a directory name, an O/R address, or both. A directory name is intended to be a user-friendly name that can be easily associated with a particular user. An O/R address contains information that enables the MHS to uniquely identify users and to route messages or return notifications to them. The directory name can be used to determine an O/R address by performing a look-up in the X.500 directory.

When both an O/R address and a directory name are given as part of an O/R name, the MHS will use the O/R address but will carry the directory name and present both to the recipient. This presents the opportunity for the sender to supply a false directory name with the intention of deceiving the receiver as to who actually sent the message. When the message is delivered to the receiver, the receiver is more likely to consult the user-friendly

directory name than the O/R address. The receiver could then respond to the message thinking that the response is going to the user associated with the directory name rather than the user associated with the O/R address.

The countermeasure to this vulnerability is to have the destination MTA and, for added security, the originating MTA resolve the directory name and compare the result with the O/R address. If the O/R address did not match the directory name, the message should be discarded or the directory name should be changed to match the O/R address.

This countermeasure is not currently possible, however, when messages are being passed between different MHS communities. An MHS community does not currently have a method of verifying O/R addresses and directory names that are external to it. Eventually, distributed directory services will be more mature, and MHS communities will be able to perform this countermeasure for externally generated messages. Until that time, users within an MHS community should take care with any information that is received from external sources.

Credentials. Another vulnerability concerns the method in which credentials are handled. When one MHS component initiates a connection (binds) with another MHS component, credentials must be supplied by the initiator. If simple authentication is used, the credentials contain a password. If strong authentication is used, the credentials contain a token and, optionally, a certificate.

Although the credentials must be supplied by the initiator, the responder is not required to perform any authentication using these credentials. If no authentication is performed, the initiator can supply any credentials.

A countermeasure to false credentials is to have a valid authentication scheme resident on all MHS components.

Register Operation. The third vulnerability that could be used to cause masquerading is related to the *register* operation. Through the *register* operation, an MTS-user can change various user parameters held by the MTS responsible for delivery of messages to that MTS-user. Two of these parameters are the user name and the user address. The recommendation does not specify any restrictions concerning the use of the *register* operation. Therefore, an MTS-user can supply any name and any address. Depending on how the MTS uses this information, other MTS-users or the MTS itself could be deceived as to who the actual user is. Access to this command needs to be restricted to the MHS administrator.

Resequencing

In terms of resequencing, messages can be replayed, reordered, preplayed, or delayed. Any of these resequencings could cause confusion or result in information arriving too late or too early.

Cancelling a deferred delivery could cause preplay of messages. The *deferred delivery* feature allows an originator to submit a message to an MTS but request that the MTS not deliver the message to the intended recipient until a specific time. As a complement to this feature, the *cancel deferred delivery* operation allows a user to cancel the delay time associated with the delivery of a deferred message and have the message delivered

immediately. However, the only argument that a user must supply to perform a cancellation is a *message submission identifier*. Therefore, one user could supply any *message submission identifier* and cancel another user's deferral resulting in the delivery of a message earlier than intended by the originator.

As a countermeasure to this vulnerability, the MTA performing the cancellation of the deferred delivery time should authenticate that the user requesting the cancellation is the originator of the deferred message.

Repudiation

Repudiation could take three forms within an MHS. The author could deny having originated the message (denial of origin), the MTS could deny having received the message from the originator (denial of submission), and the recipient could deny having received the message from the MTS (denial of delivery).

The method in which held messages are protected could result in a user being able to deny having been delivered a message. If the temporary storage where held messages are located is not adequately protected, users can gain access to the storage, read the messages that are being held for them, and then repudiate having received them since the messages were never actually delivered to the users.

To eliminate this vulnerability, proper discretionary access control on the temporary storage should be present, as should be done with the disclosure and modification threat. These held messages should be owned by the MHS administrator and should be readable only by the MHS administrator.

SUMMARY

The following is a list of countermeasures that are recommended to address the threats and vulnerabilities described in this paper.

- Discretionary access control mechanisms properly enforced on all types of stored information.
- Informing originators of the dangers of not using the following message arguments:
 - *distribution list expansion prohibited*
 - *alternate recipient prohibited*
 - *recipient reassignment prohibited*
- Making BCC lists inaccessible once the message is deliverable.
- Limiting access to and establishing thresholds on the use of the *priority* argument.
- Resolving the O/R name at the delivery MTA.
- Authenticating the credentials supplied in a *bind* operation.
- Limiting access to the *register* operation.

- Authenticating the requester of a *cancel deferred delivery* operation as the originator of the deferred message.

The threats, vulnerabilities, and countermeasures identified during the analysis are intended to be exhaustive given the scope of the analysis. However, additional issues may be identified as penetration testing and evaluations of specific implementations occur. It is important to note that many of these vulnerabilities, as well as many others, exist within current methods of messaging (e.g., Simple Mail Transfer Protocol). Also, individual implementations may address and remove many of these vulnerabilities.

LIST OF REFERENCES

The International Telegraph and Telephone Consultative Committee, November 1988, *Data Communication Networks Message Handling Systems, Recommendations X.400 - X.420*.

ISO-7498-2, 15 February 1989, *Information Processing Systems - Open Systems Interconnection - Basic Reference Model - Part 2: System Architecture*.

Gosselin, Michelle J., April 1993, *Message Handling System (X.400) Threats, Vulnerabilities, and Countermeasures*, The MITRE Corporation, M 93B0000037.

LOCKHEED TESTING PROGRAM OF THE MOTOROLA NETWORK ENCRYPTION SYSTEM (NES)

Joyce Capell, Kevin Gentile, and Greg LaPlant
The Computer Security Technical Group

Lockheed Missiles & Space Company, Inc.
1111 Lockheed Way
P.O. Box 3504
Sunnyvale, CA 94089-3504

Point Of Contact: Joyce Capell, O/27-33, B/575A, (408) 742-3376

Abstract

The requirement to provide a means for transmitting classified and unclassified data over shared public and private networks led several Lockheed companies to adopt the Motorola Network Encryption System (NES) for encryption. This device encrypts data within a datagram, leaving vital addressing information in the clear. As a result, NES encrypted data may be transmitted along with unclassified data on shared networks.

In terms of performance, Motorola claims that the maximum throughput of the NES is 800 KBPS from the Ethernet port on the red side to the Ethernet port on the black side. Preliminary Local Area Network testing of the NES by Lockheed Missiles & Space Co. (LMSC) demonstrated that throughput over the LAN was somewhat slower. However, there was insufficient information available about connectivity and performance characteristics of this new COMSEC device over public networks.

In order to determine whether the NES could be incorporated into existing and future networks, LMSC led a testing program with participants from Lockheed Aeronautical Systems Co. (LASC), Lockheed Sanders Co., GE Aircraft Engine Co. (GEAE), and Motorola. The goal of the testing program was to determine the connectivity and performance characteristics of the NES on public Wide Area Networks: Frame Relay, ISDN to Switched 56, and X.25.

The testing program demonstrated that when connecting to a Cisco Systems network router, the NES is capable of operating over all networks tested. The performance varied by several factors: type of workstation, communications software, and packet size. Once packet size was controlled to avoid fragmentation performance was optimized. In conclusion, the NES testing program was a success.

©1993 Lockheed Missiles and Space Company, Inc.

Introduction

Background

Up until recently Government Contractors have been limited to link encryptors as the approved method for transmitting classified data across networks. Link encryptors require dedicated circuits resulting in classified data being handled separately from unclassified data traffic.

In 1990, the Motorola Network Encryption System (NES) was endorsed by NSA as a Controlled Cryptographic Item (CCI). This new device encrypts data within the packet, leaving vital addressing information in the clear, thus NES-based networks support transparent network routing. Additionally, end-to-end encryption provides the means to transmit classified and unclassified data across the same Local Area and Wide Area Networks (LANs and WANs) creating a more cost effective and efficient way of transmitting classified data.

Several Lockheed companies adopted the NES device for use on classified networks in support of current and future classified programs. Preliminary testing of the NES over a local area network at Lockheed Missiles and Space Co. confirmed that the NES was able to perform transparently through a Cisco router. However, performance varied greatly depending on several factors: packet size, communications software, and workstation speed. Since the majority of Lockheed classified network implementations require connections to sites scattered around the U.S., it was decided to test the NES over various wide area networks. The purpose of the testing program was to test the ability of the NES to connect over diverse networks. Only secondarily was the test intended to test performance. It was never the intention of the testing program to test actual throughput, rather the intention was to compare performance with the NES to performance without the NES over varying networks.

Network Encryption System (NES)

The NES utilizes mandatory and discretionary access controls. Mandatory access controls (MAC) are provided by keying material which is supplied by NSA at a specific security level. Discretionary access controls (DAC) are provided by device address tables which pair specific devices across the network. Only devices which appear in the device address tables

may communicate. In order to exchange data, the NESs must have the same address pairs in their address tables and be keyed at the same security level. When a device on one NES network must communicate with a device on another NES network a "handshake" occurs. The two NES devices authenticate each other and exchange cryptographic session keys. Data packets are then encrypted and "encapsulated" within a new data packet, with the source and destination addresses of the NESs that are communicating. The addressing information travels in the clear so it can be routed across a variety of networks. Maximum throughput is 800 Kbps between the red and black side of the NES.

The NES Security Platform is configured using a configuration disk that is created by the Product Server (a stand alone PC running Motorola NES Product Server software). The Product Server software provides a set of tools to support the installation, administration, and maintenance of NES applications in the user's operational environment. The software supports up to 2000 NES address pairs. The configuration disk is cryptographically bound to the NES Security Server when installed. Audit events are recorded on the configuration disk for review by the Product Server. The NES is keyed using material provided by the NSA Electronic Key Management System (EKMS). The keying material is non-forgable and contains the NES identity and security classification level for the network.

NES Test Program

The ability of the NES to transmit classified data over shared public and private networks provides the opportunity to utilize existing company LANs and WANs as well as explore new wide area networking technologies. However, since this product is so new there has been practically no information available to evaluate connectivity or performance characteristics of the NES on other than local area networks. Before implementing this new product we wanted to test it in a wide area environment.

Aside from Lockheed Missiles & Space Co. (LMSC) several other Lockheed Companies were planning to use the NES for new classified networks, so we decided to team up on a testing program. As a result of discussions with members of the Aerospace Industries Assoc. (AIA), G.E. Aircraft Engines Co. (GEAE) expressed an interest in testing the NES too, so they were invited to join in the testing program. The participants were LMSC in Sunnyvale, CA,;

Lockheed Aeronautical Systems Co. (LASC) in Marietta, Georgia; Lockheed Sanders (Sanders) in Nashua New Hampshire; and GE Aircraft Engines (GEAE) in Cincinnati, OH would participate in a testing program, assisted by Motorola who would supply NES devices to sites which did not already have them. After a preliminary meeting with all participants the following wide area networks were selected for testing: Sprint Frame Relay, AT&T Switched 56 with local ISDN access, and X.25. It was further agreed that all test sites would participate in the Frame Relay test, but that the Switched 56 test would be between LMSC and LASC only, and that GEAE would conduct its own X.25 test.

Following is a brief description of each of the networks that was tested and the equipment used in the testing program.

Network Technologies

Frame Relay

Frame Relay is a high-speed data communication interface closely related to the X.25 interface. It is based on the fact that current telecommunications technologies are less prone to error. In contrast to X.25, Frame Relay relies on the transport layer for frame re-transmission and acknowledgment. Frame Relay networks can burst as high as the speed of the local loop, typically 56 Kbps to T1 rates, but operate on a committed information rate (CIR).

Switched 56

Switched 56 is a tariffed wide area network service which provides data transmission over 56 KB circuits on a usage sensitive basis. It is essentially "dial-up" digital service at 56 KBPS. Switched 56 long-haul service is accessed through the local telephone service provider. Access may be via ISDN service, local 56 switched service, or dedicated 56 KB circuits.

ISDN

Integrated Services Digital Network (ISDN) is a service offered by local telephone companies combining voice and digital network services in a single medium, making it possible to offer customers digital data services as well as voice connection through a single "wire". Basic Rate Interface (BRI)

ISDN dial up service may be used as an interface to Switched 56 long haul service.

X.25

X.25 networks follow a standard that defines the packet format for data transfers in a public data network. X.25 networks provide error correction and detection and support most data protocols. Most public X.25 networks operate on circuits that range from 56 Kbps to T1.

Communications Equipment and Software

Cisco Router

The Cisco router was selected as the network interface device since it is capable of providing transparent routing and supports multiple network interfaces. It is capable of routing data packets to most networks and/or devices on those networks.

Cisco routers used during the testing included the IGS, MGS, MGS/2, AGS, and AGS+. The PROM software, provided by Cisco Systems, varied between sites from version 8.2 GS2-BRX to 8.3 IGS-BRX, with B=Bridging software, R=Standard System software which executes out of ROM, and X=Standard + Commercial/DDN X.25 software. Both Frame Relay and X.25 networks require the X functionality in order to operate.

CSU/DSU

The Channel Service Unit/Digital Service Unit (CSU/DSU) controller connects to the wide area network. It connects the Cisco router to dedicated lease lines (T1) where it converts V.35 or RS-449 signals to the properly coded T-1 transmission signal.

ISDN Terminal Adapter and NT1

The Terminal Adapter provides an interface so that non-ISDN equipment can use ISDN lines. It converts from the ISDN standard interface to the serial interface (e.g. RS-232, V.35, or RS-449) on most data devices. It also provides dialing capabilities to access the switched ISDN network which in-turn hands the call off to the Switched 56 network.

The Network Termination unit (NTI) converts 2-wire phone network interfaces to 4-wire ISDN terminal interfaces. It performs the multiplexing required for the ISDN line, as well as echo cancellation.

NES Software

For the testing program both the DoD LAN/INTERNET and Transparent LAN/INTERNET software were used. The Transparent software provides end-to-end security between hosts running arbitrary network protocols, such as XNS, DECnet, Ethernet, etc., while the DoD software supports only TCP/IP at the network layer and Ethernet versions I & 2 MAC as the link and physical protocols respectively.

Equipment Setup

Since Unix Workstations use IP addressing each workstation must have its own unique IP address in order to communicate. IP addresses are composed of a network number and a host number on the affiliated network. Class B IP addresses were used. The lower two octets of the IP address varied with each test site and host.

SUN Workstations used during the testing program included the SPARCstation 1 and the SPARCstation IPX. The SPARCstation 1 is a 12.5 MIPS (Million of Instructions Per Second) machine running Unix 4.0.3c. The SPARCstation IPX is a 28 MIPS machine running Solaris 1.3 in a Unix environment. The SUNs setup was modified to meet certain requirements for the NES that were not part of SUN's standard network setup. A default routing IP address was added to allow the SUNs to communicate with the RED side of the NES. Since the NES adds overhead to the packet, the MTU (Maximum Transfer Unit) was changed from 1500 bytes to 1150 bytes to eliminate fragmentation. When this change was made performance with the NES was almost twice as fast as when fragmentation occurred. The send and receive buffers were increased from 4096 bytes to 16384 bytes to allow a greater amount of data to be accessed and produce a more accurate representation of the transmission rates.

Hewlett Packard (HP) 9000 Series 360 Workstations were used. This is a 22 MIPS (Million of Instructions Per Second) machine running HP-UX 8.0 in a Unix environment. The HP systems setup used the same parameters as the SUNs. The MTU on the HP is

coded in ROM and was not changed at the time of testing.

The DEC/VAX Workstation family uses DECnet addressing, which is the physical addressing scheme. Thus, testing with DEC workstations utilized the "Transparent Mode" NES software instead of "DoD Internet". Transparent mode software utilizes the physical (Ethernet) address on the red side of the NES.

The DEC systems used for the network set-up were VAXstation 2000 and MicroVAX 3800. The VAXstation 2000 runs VMS 4.5B with a processor speed of 0.9 MIPS. The MicroVAX 3800 runs VMS 5.0 with a processor speed of 2.7 MIPS. DEC VAXstations setup addressed the Executor Node for each site. The changes made were the same as for the SUNs, with only a labeling difference of the setup parameters. The MTU on the DEC systems is called the Line buffer and the send and receive buffers are called Buffer and Segment buffers. These parameters are changed in NCP (Network Configuration Program). The Line, Buffer and Segment buffers were set to 1150 bytes. The total packet size including overhead was set so as to never exceed the limit of 1236 bytes.

Initially PCs with Ethernet cards were used during the test program, with the TCP/IP application software varying by site. LMSC used National Center for Supercomputing Applications (NCSA) "telnet" software. Other sites used "FTP telnet" software or S/W from different vendors. Because of the variance between sites in PC's processing power and communications software, performance varied so greatly it was difficult to separate out the causes. For that reason the PC tests are not reported in this paper.

Frame Relay Testing

Frame Relay Network

In support of Frame Relay testing Sprint provided service to all test sites, as well as to Motorola for monitoring purposes. Technical support was available throughout the testing program. Sprint also provided Sprintfax and audioconferencing service to support the testing program. Weekly audioconferences were held during testing.

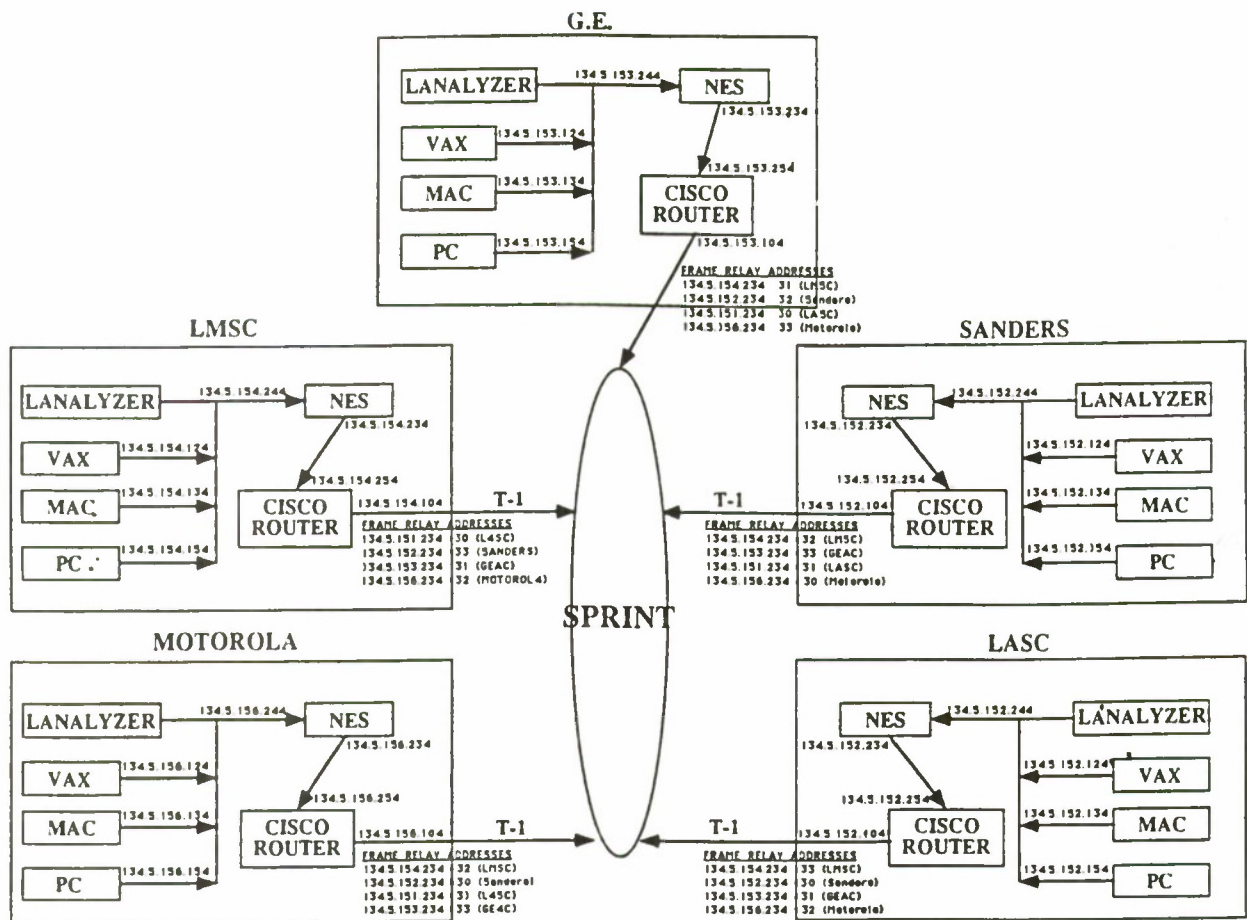


Figure 1. Frame Relay Configuration Diagram

In order to have a clear concept of wide area network configuration a detailed network map was drawn, complete with TCP/IP addresses for all devices, Figure 1. This is a "must" when configuring a network with this level of complexity. Each site had a copy of the network map, which was used extensively for trouble-shooting.

Sprint Frame Relay implementations use Permanent Virtual Circuits (PVC), Figure 2. A Frame Relay station's address is its Data Link Connection Identifier (DLCI), a 10-bit address field in the first two octets of the Frame Relay protocol header. The DLCI is used to specify particular Permanent Logical Link (PLL) endpoints within a user's access channel, and has local significance only to given channels and networks as a whole, Figure 3.

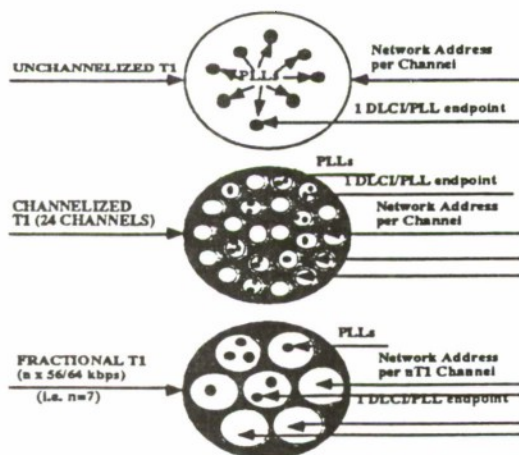


Figure 2. DLCI/PLL Assignments

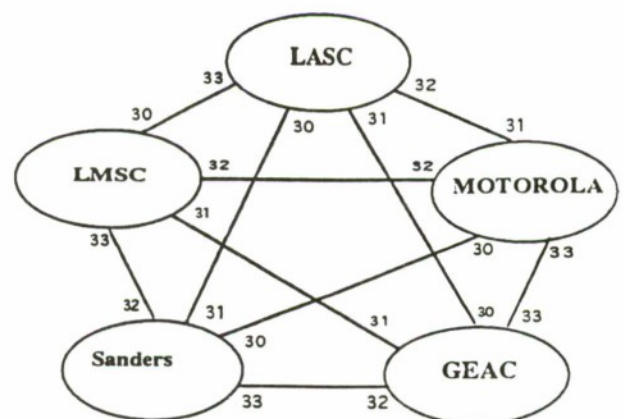


Figure 3. Frame Relay Network PLL/DLCI Mesh

Sprint provided all sites access to the Frame Relay network using half of the bandwidth (768 Kbps) on T-1 access lines. The duration of the test was 115 days. Motorola was the only site with an access rate of 54/64 Kbps (but Motorola only monitored the testing program, therefore there is no test data reported from that site). Frame Relay Committed Information Rates (CIR) varied by test site and ranged between 128 Kbps and 192 Kbps. (Fig. 4)

	LMSC	LASC	SAND	GEAE	MOTO
LMSC	X	192	128	128	4.8
LASC	192	X	128	128	4.8
SANDERS	128	128	X	192	4.8
GEAE	128	128	192	X	4.8
MOTOROLA	4.8	4.8	4.8	4.8	X

Figure 4. CIR Channel Assignments (KBits)

Testing consisted of file transfers between workstations. File transfer rates between two sites were compared: **without** the NES vs. the same file transfer **with** the NES.

The routers were setup in a standard configuration for IP routing with IGRP enabled, Figure 5. The configuration for Frame Relay was added to the standard router setup by encapsulating the DLCI address to the serial 0 IP address and setting the "keepalive" timer equal to 10 seconds. The ethernet 0 port was set with its IP address and the "keepalive" timer equal to 5 seconds. DECnet was setup in the standard configuration for testing without the NES. When the NES was included in the setup DECnet information was removed from the router since the NES puts an IP header on the DECnet packet and routes it as an IP packet.

```
enable - password temp
.....
decnet routing 4.4
decnet node - type routing iv
decnet max - address 1023
!
interface Ethernet 0
ip address 134.5.154.244 255.255.255.0
decnet cost 10
frame-relay keepalive 5
!
interface Serial 0
ip address 134.5.50.101 255.255.255.0
decnet cost 5
encapsulation FRAME-RELAY
frame-relay map IP 134.5.50.102 31 broadcast
frame-relay map IP 134.5.50.103 33 broadcast
```

```
frame-relay map IP 134.5.50.104 30 broadcast
frame-relay map IP 134.5.50.105 32 broadcast
frame-relay keepalive 10
!
router igrp 4
network 134.5.0.0
!
ip name - server 134.5.50.101
ip host LMSC 134.5.154.244
snmp - server community
hostname LMSC-Framc-Relay
.....
end
```

Figure 5. Frame Relay Cisco Router Configuration

Frame Relay Test Results

The rates in the tables of Figure 6 are transfer rates using the GET and PUT options under the File Transfer Protocol procedure. The UDP spray command was used to evaluate maximum buffer to buffer transfer without involving delay times of disk drives. The DECnet copy command was used for the NES transparent mode set-up. All monitoring of traffic was done at individual sites.

LMSC > LASC SUN to HP (FTP) File size: 2963280 bytes

w/o NES	total bytes	total time	KBPS
GET	3224799	150 sec	168
PUT	3224799	170 sec	152
with NES	total bytes	total time	KBPS
GET	3224799	230 sec	112
PUT	3224799	290 sec	96

GET % of Throughput Loss with NES	33.33%
PUT % of Throughput Loss with NES	36.84%

LASC > LMSC HP to SUN (FTP) File size: 2963280 bytes

w/o NES	total bytes	total time	KBPS
GET	3224799	60.8 sec	424
PUT	3224799	41.9 sec	616
with NES	total bytes	total time	KBPS
GET	3224799	108 sec	312
PUT	3224799	53.7 sec	480

GET % of Throughput Loss with NES	26.42%
PUT % of Throughput Loss with NES	22.08%

LMSC > Sanders SUN to SUN (FTP) File size: 2963280 bytes

w/o NES	total bytes	total time	KBPS
GET	3224799	123.8 sec	216
PUT	3224799	130 sec	200
with NES	total bytes	total time	KBPS
GET	3224799	161 sec	160
PUT	3224799	169 sec	152

GET % of Throughput Loss with NES	25.93%
PUT % of Throughput Loss with NES	24.00%

Sanders > LMSC SUN to HP (FTP) File Size:

No data was received from Sanders for this report

LMSC > GEAE SUN to HP (FTP) File size: 2963280 bytes

w/o NES	total bytes	total time	KBPS
GET	3224799	138 sec	200
PUT	3224799	144 sec	192
with NES	total bytes	total time	KBPS
GET	3224799	140 sec	176
PUT	3224799	150 sec	168

GET % of Throughput Loss with NES	12.00%
PUT % of Throughput Loss with NES	12.50%

GEAE > LMSC HP to SUN (FTP) File size: 2963280 bytes

w/o NES	total bytes	total time	KBPS
GET	3224799	54.3 sec	432
PUT	3224799	33.6 sec	688
with NES	total bytes	total time	KBPS
GET	3224799	76.3 sec	304
PUT	3224799	52.3 sec	456

GET % of Throughput Loss with NES	29.63%
PUT % of Throughput Loss with NES	33.70%

LMSC > GEAE VAX to VAX (DECnet) File size: 2963280 bytes

w/o NES	total bytes	total time	KBPS
GET	3224799	69 sec	419
PUT	3224799	67 sec	402
with NES	total bytes	total time	KBPS
GET	3224799	86 sec	325
PUT	3224799	85 sec	329

GET % of Throughput Loss with NES	22.43%
PUT % of Throughput Loss with NES	18.15%

GEAE > LMSC VAX to VAX (DECnet) File size: 2963280 bytes

w/o NES	total bytes	total time	KBPS
GET	3224799	83 sec	339
PUT	3224799	80 sec	351
with NES	total bytes	total time	KBPS
GET	3224799	108 sec	260
PUT	3224799	99 sec	272

GET % of Throughput Loss with NES	23.30%
PUT % of Throughput Loss with NES	22.50%

Figure 6. Frame Relay Test Results

Analysis of Frame Relay Tests

During the Frame Relay testing a great deal was learned about disparities in throughput between sites on the network and packet fragmentation. Throughput was almost twice as high when packet fragmentation was avoided. The test results reported show the highest throughput achieved when the MTU was optimized to avoid packet fragmentation. Earlier tests show almost twice as much loss in throughput before the MTU was adjusted.

During testing of the VAX workstations only LMSC and GEAE were involved. There was no contention for the network from the other sites and throughput was much higher, however the percentage of loss with the NES was consistent with the UNIX/TCP/IP testing.

ISDN/Switched 56 Testing

The Switched 56 testing program, Figure 7, involved only two sites, LMSC and LASC. LMSC accessed the ATT Accunet 56 Switched network via a local ISDN connection provided by Pacific Bell who also provided the network interface devices (Fujitsu Terminal Adapter and NT1). LASC accessed Accunet via a local 56 KB switched service from Southern Bell.

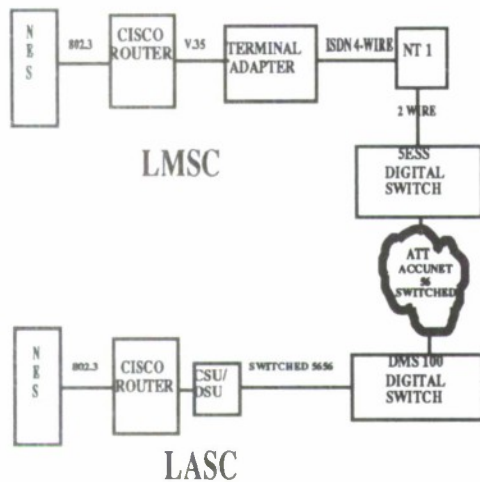


Figure 7. Switched 56 Network

Results demonstrated that the interface into the Switched 56 network was transparent and did not affect the testing program. The following tables illustrate the Switched 56 results, Figure 8.

LMSC > LASC SUN to IIP (Ethernet) File size: 3034090 bytes

w/o NES	total bytes	total time	KBPS
GET	3104900	540 sec	44.8
PUT	3104900	460 sec	52.8
with NES	total bytes	total time	KBPS
GET	3104900	590 sec	40.8
PUT	3104900	480 sec	50.4

GET % of Throughput Loss with NES	8.93%
PUT % of Throughput Loss with NES	4.55%

LASC > LMSC IIP to SUN (Ethernet) File size: 3034090 bytes

w/o NES	total bytes	total time	KBPS
GET	3224799	463 sec	52.3
PUT	3224799	523 sec	45.9
with NES	total bytes	total time	KBPS
GET	3224799	487 sec	49.8
PUT	3224799	573 sec	41.8

GET % of Throughput Loss with NES	4.78%
PUT % of Throughput Loss with NES	8.93%

Figure 8. Switched 56 Test Results

X.25 Testing

The X.25 testing program involved one site, GEAE. GEAE connected two HP 9000 Workstations through a X.25 Modem Eliminator. The data presented, below represents the throughput of the X.25 network using the NES while varying the data rate, Figure 9. There was no attempt to compare the throughput without the NES.

GEAE > GEAE IIP to IIP File size: 2.3 KB bytes, Data Rate=64 KBS

with NES	total bytes	total time	KBPS
GET	231011	32.45 sec	55.6
PUT	231011	32.83 sec	54.96

GEAE > GEAE IIP to IIP File size: 2.3 KB bytes, Data Rate=56 KBS

with NES	total bytes	total time	KBPS
receive	231011	36.90 sec	48.88
transmit	231011	37.18 sec	48.56

GEAE > GEAE IIP to IIP File size: 2.3 KB bytes, Data Rate=48 KBS

with NES	total bytes	total time	KBPS
receive	231011	43.07 sec	41.92
transmit	231011	43.37 sec	41.6

GEAE > GEAE IIP to IIP File size: 2.3 KB bytes, Data Rate=19.2 KBS

with NES	total bytes	total time	KBPS
receive	231011	113.51 sec	15.2
transmit	231011	110.18 sec	16.4

LASC > LASC IIP to IIP File size: 2.3 KB bytes, Data Rate=9.6 KBS

with NES	total bytes	total time	KBPS
receive	231011	214.74 sec	8.4
transmit	231011	215.08 sec	8.4

Figure 9. X.25 Test Results

Conclusions

The testing program successfully demonstrated that connectivity was possible using the Motorola NES over a variety of Wide Area Networks (WANs). Performance varied by several factors: 1) type of network 2) packet size, 3) available bandwidth, 4) communication speed of the actual workstation, and communication software. When packet fragmentation was controlled by reducing the MTU size, performance with the NES was improved significantly. Even with packet fragmentation, the

NES performance was within an acceptable range for lower speed networks. Frame Relay test results show the greatest variation in NES performance. Were the tests to be performed again, it is predicted that NES performance could be enhanced by controlling the MTU. Much was learned about this factor during the testing. Later tests on lower speed networks (56KB) show very little degradation in throughput with the NES.

The NES currently can support a maximum throughput of 800 KBPS between the Ethernet ports on the red side and black side of the NES. Motorola has indicated that further product enhancements will bring the throughput up to T1 rates. Our test results illustrate what can be expected at less than T1 rates. This performance limitation precludes the NES for high speed data transfer requirements. But, when throughput requirements for classified data fall within the range of T1 rates or lower, the NES provides all the advantages of packet switched networking without the disadvantages of link encryptors and dedicated data circuits.

Test Participants

Lockheed Missiles & Space Co.
1111 Lockheed Way
Sunnyvale, California 94089-3504
Contact: Joyce Capell (408) 742-3376

Lockheed Aeronautical Systems Co.
Georgia Division
86 S. Cobb Drive
Marietta, Georgia 30063
Contact: Jim Berg (404) 916-2671

Lockheed Sanders Co.
NHQ
P.O. Box 0868
Nashua, New Hampshire 03061-0868
Contact: Dave Morin (603) 885-3509

General Electric Aircraft Engines
8700 Governors Hill Drive
Cincinnati, Ohio 45249
Contact: Steve Giese (513) 583-3592

Supporting Organizations

Motorola Gov't Electronics Group
8201 E. McDowell Road
Scottsdale, Arizona 85257
Contact: Olan Wade (602) 441-2963

Sprint
535 Anton Blvd., Suite 1100
Costa Mesa, California 92626
Contact: Andryce Zurick (714) 435-3200x336

Cisco Systems
1525 O'Brien Drive
Menlo Park, California 94025
Contact: Sunil Dhar (415) 326-1941

Pacific Bell
1550 Leigh Avenue
San Jose, California 95125-5385
Contact: Alyson Harber (408) 723-1786

CERTIFICATION AND ACCREDITATION APPROACH FOR THE WWMCCS GUARD

Brian Tretick
Booz●Allen & Hamilton
8283 Greensboro Drive
McLean, VA 22102-3838
tretickb@jmb.ads.com

INTRODUCTION

This paper describes the certification and accreditation (C&A) approach being undertaken for the U.S. Central Command (USCENTCOM) Worldwide Military Command and Control System (WWMCCS) Guard. The USCENTCOM command center staff requires responsive communications among the local WWMCCS computers, WWMCCS Intercomputer Network (WIN), and the Command Automation System (CAS) to support operations, exercises, and contingencies. The purpose of the WWMCCS Guard is to provide a secure, bidirectional communication channel between the Top Secret WWMCCS and the Secret CAS for the transfer of time phased force and deployment data (TPFDD), Status of Readiness and Training System (SORTS) data, WIN teleconference messages, and electronic mail.

The USCENTCOM WWMCCS Guard is a second generation WWMCCS Guard, founded on enhancements to the WWMCCS/Crisis Action Team (CAT) Guard developed under the Air Force Project High Gear for the Air Mobility Command. The WWMCCS/CAT Guard began development in November 1990 and was installed at USTRANSCOM/AMC in March 1992. It is currently certified, accredited, and operational at the Air Mobility Command in Scott Air Force Base, IL. The first generation WWMCCS Guard was the U.S. Forces Command (FORSCOM) Security Monitor.

A guard is a device that is trusted to mediate the transfer of data between systems operating at different security levels. Guard technology is being used to alleviate operational and security constraints associated with the high-to-low and low-to-high data flow between the WWMCCS and the CAS. This capability will eliminate time consuming and cumbersome magnetic media transfers and associated manual procedures that are used today for data transfer between the two systems. The intent is that users will have more timely and accurate data on which to base command and control decisions.

A key issue in automating the data flow process involves the security of both the data being transferred and the processes providing the transfer mechanism, including data confidentiality, data integrity, and system integrity. Security mechanisms are needed in the process to ensure that inadvertent disclosure and unauthorized transfers do not occur. Security mechanisms are also needed to provide identification and authentication of the users conducting and reviewing the data transfer, to provide audit trails of the transfers and other relevant events, and to provide access controls to prevent unauthorized persons from performing the transfers or from accessing classified data. Along with security mechanisms, assurances that the security mechanisms work correctly and cannot be circumvented are needed before any trust can be placed in the automated interconnections.

ROLES AND RESPONSIBILITIES

The USCENTCOM WWMCCS Guard development is sponsored by the DoD Multilevel Security (MLS) Program at the Defense Information Systems Agency and National Security Agency. The organizations involved with the C&A of the WWMCCS Guard are the:

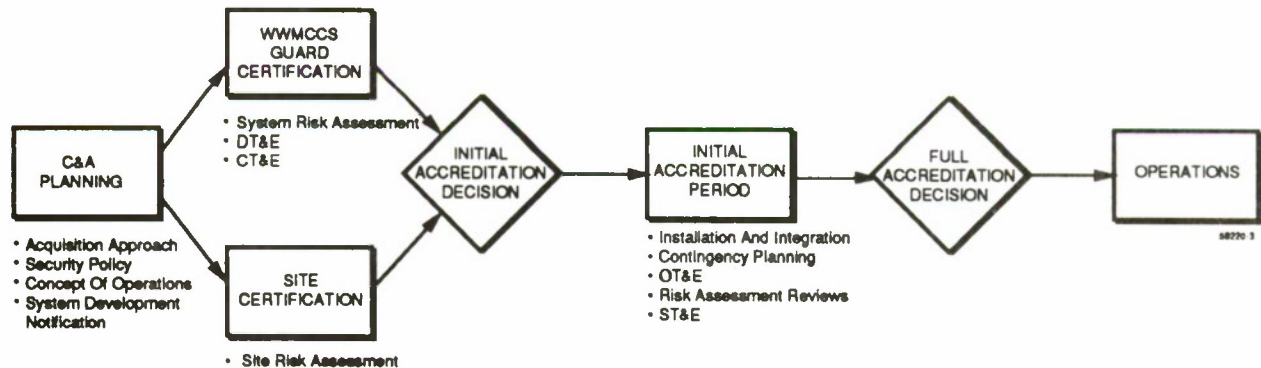
- **Joint Staff Accreditation Authority.** Director for Command, Control, and Communications Systems, Office of the Joint Chiefs of Staff (JCS/J-6)
- **USCENTCOM Accreditation Authority and Program Manager.** Command and Control, Communications, and Computer Systems Directorate, USCENTCOM (USCENTCOM/CCJ6)

- **Certification Authority.** Director, Defense Systems Support Organization (DSSO), ADP Security Division (DISA/DSSO/JPS)
- **Sponsor.** Defense Information Systems Agency DoD MLS Program Office (DISA/JIEO/TFDA)
- **Certification Technical Agent.** National Security Agency (NSA/V37)
- **Acquisition Office.** Air Force Electronic Systems Center, Security Products Program (ESC/ENS).

CERTIFICATION AND ACCREDITATION APPROACH

The C&A approach is founded on the system development approach being undertaken for the guard. The goal of the C&A approach is to provide sufficient, credible information to both the Joint Staff and USCENTCOM accreditation authorities so that they may make informed decisions regarding the approval to operate the guard. Such informed decisions require a detailed understanding of both operational and security requirements for the guard. The approach outlined in the following sections accounts for the technical (e.g., computer system related) and site (e.g., physical and procedural) certifications of the WWMCCS Guard and the various decision points to be encountered during the development and deployment of the guard. Exhibit 1 illustrates the C&A approach being undertaken.

EXHIBIT 1 WWMCCS GUARD CERTIFICATION AND ACCREDITATION APPROACH



CERTIFICATION AND ACCREDITATION PLANNING

The C&A planning activities guide the guard development and deployment and provide the foundation for the certification activities and accreditation decisions. The C&A plan is the result of some of the earliest planning for the WWMCCS Guard. The following activities will comprise this phase of the C&A approach:

Review Acquisition Approach. The acquisition documentation (e.g., contracts, statements of work, contract requirements deliverable lists) for the WWMCCS Guard define the activities and deliverables that will be completed by the system integrator and made available to the various participants of the WWMCCS Guard C&A process. This documentation and other related information will be reviewed by the program manager and approved by the accreditation authorities before award. This step is included in the process to ensure that provisions for meeting accreditation authority requirements are addressed. Of particular note, in addition to the technical specifications of security requirements for the WWMCCS Guard, the requirements of Joint Publication 6-03.7 must also be met, as applicable to the guard and its operations.

Develop Concept of Operations. The concept of operations defines the functionality of the WWMCCS Guard within the USCENTCOM operational environment, including descriptions of the intended data flow between the WWMCCS and the CAS, controls placed on that flow, and user demographics (e.g., clearances and authorizations, experience base) and responsibilities. The acquisition office will prepare the concept of operations document, following which the USCENTCOM user community and the Joint Staff and USCENTCOM accreditation authorities will review and approve the document.

Define Security Policy. The security policy defines the rules to be enforced by the guard to control the data flow between the WWMCCS and the CAS. The security policy document will be developed by the acquisition office and reviewed and approved by the USCENTCOM user community and the Joint Staff and USCENTCOM accreditation authorities.

Prepare System Development Notification. The system development notification (SDN) is a mechanism by which modifications to local WWMCCS configurations are coordinated with the appropriate command and Joint Staff offices. The SDN process is defined in Joint Publication 6.03-11. The SDN, which includes elements of the concept of operations, will be prepared by the USCENTCOM and the draft submitted for review by the Air Staff before the final version is sent to the Joint Staff accreditation authority for approval.

WWMCCS GUARD CERTIFICATION

The WWMCCS Guard certification involves an assessment of its automated security services, security assurances, and security properties. The scope of the certification includes all equipment (hardware and software) associated with the transfer of data between the WWMCCS and CAS. The WWMCCS Guard certification activities include:

Perform System Risk Assessment. The certification test and evaluation (CT&E) team, comprised of DISA and NSA, will perform a system risk assessment. The system risk assessment will be based on the WWMCCS Guard concept of operations, system specifications, and other information available to identify threats to and vulnerabilities in the guard that require corrective action to achieve acceptable levels of risk. This system risk assessment will address the technical and procedural aspects of the system that comprise the data flow process. Assumptions regarding the physical security, personnel clearances, and other site issues will be made. These assumptions will be validated in the site risk assessment and through revisiting the risk assessments during the initial accreditation period. The results of this activity will be documented in a system risk assessment report.

Monitor Developmental Test and Evaluation. The CT&E team will monitor the system integrator's developmental test and evaluation (DT&E) as part of the preparation for the CT&E. The CT&E team will make recommendations, as appropriate, on DT&E test objectives and procedures related to system security.

Conduct Certification Test and Evaluation. The CT&E team will plan, conduct, and report on the WWMCCS Guard CT&E. The CT&E will include functional security and penetration testing, trusted software assessment, documentation review, design analysis, and related technical evaluation of the WWMCCS Guard. The results of this activity will be documented in the CT&E report that includes the conclusions and recommendations of the CT&E team. The CT&E report will be presented to the certification authority, the Defense Systems Support Organization (DSSO), for review. If appropriate, the certification authority will prepare a certification statement for the guard.

SITE CERTIFICATION

Site certification is performed to ensure that the appropriate security measures are established for the operating environment. Site certification for the WWMCCS Guard will address physical security, TEMPEST protection, RED/BLACK separation issues, personnel security, and procedural security. The site certification activities are the responsibility of the local security authority, the WWMCCS ADP System Security Officer (WASSO) for systems involved with WWMCCS. The site certification activities are addressed in the site risk assessment.

Perform Site Risk Assessment. The WASSO will conduct or update a site risk assessment for the WWMCCS Guard operating environment to serve as the foundation for the site certification. The site risk assessment will address computer centers, user work areas, storage areas, utilities, and personnel related to the installation and operation of the WWMCCS Guard. The site risk assessment will become an integral part of the accreditation packages used to support the initial and full accreditation decisions.

INITIAL ACCREDITATION DECISION

The initial accreditation decision occurs at the development milestone after completion of the certification activities and before the completion of installation and integration activities in the operational environment. The initial accreditation decision allows for the installation of the WWMCCS Guard at USCENTCOM and the commencement of functional testing, operational test and evaluation (OT&E), user training, and other activities in the operational environment. The decision involves both the Joint Staff and USCENTCOM accreditation authorities, with the first decision to be made by the Joint Staff. The USCENTCOM accreditation authority will then use the Joint Staff initial accreditation statement as a basis for the local USCENTCOM decision, but will also weigh local security issues that the Joint Staff accreditation authority may not consider. This process ensures that the operational command at USCENTCOM does not employ equipment or techniques that were not approved by the WWMCCS authority.

Ensure Compliance with Minimum Security Requirements for Initial Accreditation. The Joint Publication 6-03.7 mandates the following for initial accreditation:

- Physical security of all WIN-related components must meet the Joint Staff requirements
- All WIN users must have interim or final US Top Secret clearances
- The WASSO and WWMCCS ADP Terminal Area Security Officers (WATASOs) must be appointed in writing
- All user personnel must be instructed in their responsibilities for protecting their Top Secret passwords and the rules for accessing all WIN-related resources
- Communications links among the WIN-related components must be protected at the Top Secret level.

The WASSO will prepare a statement assuring that these requirements have been met and will forward that statement to the Joint Staff and USCENTCOM accreditation authorities.

Review Initial Accreditation Package. The Joint Staff accreditation authority will be presented the initial accreditation package for review and to support the initial accreditation decision. The initial accreditation package will include:

- DT&E Summary
- SDN
- Risk Assessment Summary
- Statement of Minimum Security Requirement Compliance
- Certification Statement.

Make Initial Accreditation Decision. Based on this input and verbal discussions with the certification authority and other program participants, the Joint Staff accreditation authority will decide whether to grant initial accreditation at this time. If initial accreditation is granted, the Joint Staff accreditation authority will generate the Joint Staff initial accreditation statement and inform the program participants, including the USCENTCOM accreditation authority, of the decision. If initial accreditation is not granted at this time, the Joint Staff accreditation authority will state the deficiencies in the accreditation package. A plan of action will be developed by the acquisition office to remedy those deficiencies.

Locally Review Initial Accreditation Package. Even though the system has been granted initial accreditation by the Joint Staff accreditation authority, approval by the USCENTCOM accreditation authority is also needed to continue with the installation and integration of the WWMCCS Guard in the USCENTCOM operational environment. To support this decision, the initial accreditation package will be presented to the USCENTCOM accreditation authority for review. The initial accreditation package for USCENTCOM review will include:

- Joint Staff Initial Accreditation Statement
- DT&E Summary
- SDN
- Risk Assessment Summary
- Statement of Minimum Security Requirement Compliance
- Certification Statement.

Make Local Accreditation Decision. Based on this input and verbal discussions with the WASSO and other program participants, the USCENTCOM accreditation authority will decide whether to grant initial accreditation at this time. If initial accreditation is granted, the USCENTCOM accreditation authority will generate the USCENTCOM initial accreditation statement and inform the program participants, including the Joint Staff accreditation authority, of the decision. If initial accreditation is not granted at this time, the USCENTCOM accreditation authority will state the deficiencies in the accreditation package. A plan of action will be developed by the acquisition office to remedy those deficiencies.

Upon successful completion of these activities, the WWMCCS Guard will have received initial accreditation from both the Joint Staff and USCENTCOM, which will allow further installation and integration efforts, including connection to WWMCCS, to proceed.

INITIAL ACCREDITATION PERIOD

During the initial accreditation period, the WWMCCS Guard may be installed and integrated into the operational environment. However, at this time it will not be considered operational and will not be used to transfer operational data except for controlled testing. The following activities will be performed during this period to support the full accreditation of the WWMCCS Guard once it is successfully installed and integrated in the operational environment. The primary participants in these activities are the WASSO (or representative) and the security test and evaluation (ST&E) team, comprised of DISA, NSA, and USCENTCOM personnel. The initial accreditation period activities include:

Review System Installation. The WASSO and ST&E team will review the installation of the WWMCCS Guard to ensure that all aspects of system security are being addressed. The review will consider:

- Physical configuration and location of equipment, cabling, and media
- Security procedures established for the installation and integration efforts
- TEMPEST and RED/BLACK separation considerations.

The WASSO and ST&E team will make recommendations as appropriate if security deficiencies in the installation are identified.

Monitor Operational Test and Evaluation. The WASSO and ST&E team will monitor the execution of the OT&E conducted by USCENTCOM as a means to prepare for the ST&E. As appropriate, the WASSO and ST&E team will make suggestions for security-related functional tests to the OT&E team.

Revisit Risk Assessments. The WASSO will revisit the existing site risk assessments to identify any vulnerabilities that may have been introduced through the installation and integration of the WWMCCS Guard. The WASSO will take corrective action to ensure that an acceptable level of protection is in place for the system to be used in the operational environment.

Develop Contingency Plan. The USCENTCOM operations staff will develop a contingency plan for the WWMCCS Guard. The contingency plan will address the resumption of manual or other alternative procedures in case of failures in or other unavailability of the guard. The objective of this activity is to ensure that technical or security failures in the guard do not result in the loss of command and control capabilities.

Conduct Site Security Test and Evaluation. The ST&E team will conduct the site ST&E for the WWMCCS Guard. The ST&E will address the accepted test objectives defined in the ST&E plan and other security concerns identified in the initial accreditation statements. The ST&E team will document the results of the testing in the ST&E report.

Before proceeding to the full accreditation decision point, the acquisition office will ensure that all the accreditation authority requirements have been appropriately met.

FULL ACCREDITATION DECISION

The full accreditation decision considers the ability of the deployed WWMCCS Guard to securely and effectively function in the USCENTCOM operational environment. As with the initial accreditation decision, the full accreditation decision involves both the Joint Staff and USCENTCOM accreditation authorities, with the first decision to be made by USCENTCOM with full consideration to both technical and operational issues. The Joint Staff accreditation authority will then use the USCENTCOM accreditation statement as a basis for the final decision. This process ensures that the Joint Staff decision is based on conditions that are acceptable to USCENTCOM as the operational command.

Locally Review Full Accreditation Package. The USCENTCOM accreditation authority will be presented the full accreditation package for review and to support the full accreditation decision. The full accreditation package will include:

- OT&E Summary
- Contingency Plan
- SDN
- Revised Risk Assessment Summary
- ST&E Summary.

Make Local Accreditation Decision. Based on this input and verbal discussions with the WASSO and other program participants, the USCENTCOM accreditation authority will decide whether to grant full accreditation at this time. If full accreditation is granted, the USCENTCOM accreditation authority will generate the USCENTCOM accreditation statement and inform the program participants, including the Joint Staff accreditation authority, of the decision. If full accreditation is not granted at this time, the USCENTCOM accreditation authority will state the deficiencies in the accreditation package. A plan of action will be developed by the acquisition office to remedy those deficiencies.

Review Full Accreditation Package. The Joint Staff accreditation authority will be presented the full accreditation package for review and to support the full accreditation decision. The full accreditation package will include:

- USCENTCOM Accreditation Statement
- OT&E Summary
- Contingency Plan
- SDN
- Revised Risk Assessment Summary
- ST&E Summary.

Make Full Accreditation Decision. Based on this input and verbal discussions with the certification authority and other program participants, the Joint Staff accreditation authority will decide whether to grant full accreditation at this time. If full accreditation is granted, the Joint Staff accreditation authority will generate the Joint Staff accreditation statement and inform the program participants, including the USCENTCOM accreditation authority, of the decision. If full accreditation is not granted at this time, the Joint Staff

accreditation authority will state the deficiencies in the accreditation package. A plan of action will be developed by the acquisition office to remedy those deficiencies.

Upon successful completion of these activities, the WWMCCS Guard will have received full accreditation from both the Joint Staff and USCENTCOM, which will allow operations using the WWMCCS Guard to commence. The accreditation statements, however, may impose operating constraints or require additional activities to be completed for reaccreditation once the original accreditation has expired.

OPERATIONS

During this stage of the WWMCCS Guard life cycle, the guard will be used operationally to serve its mission. Changes to the guard may require recertification and reaccreditation, depending on the nature of change. USCENTCOM will inform the USCENTCOM and Joint Staff accreditation authorities when any of the following events regarding the WWMCCS Guard occur:

- A change in the operational use of the guard
- A change in the security characteristics of the user population
- A change in connectivity with external systems
- A change in system size, scope, or location that increases the system risk
- The manifestation of unanticipated threats.

Any such event invalidates the original accreditation. In addition, an accreditation will be valid for no more than three years. A new accreditation will be made by the USCENTCOM and Joint Staff accreditation authorities before operations may continue.

SUMMARY

This approach for certifying and accrediting a guard between the Top Secret WWMCCS and the Secret command system at the USCENTCOM will also be applied for the other DoD MLS Program deployments of the standard WWMCCS Guard in fiscal year (FY) 94 and FY 95. Although specific elements of the process address the WWMCCS security regulations imposed by the Joint Staff (e.g., the SDN process), this process in general applies to other guards or for systems involving multiple accreditation authorities.

ACRONYMS

ADP	Automated Data Processing
AMC	Air Mobility Command
C ³ I	Command, Control, Communications, and Intelligence
C&A	Certification and Accreditation
CAS	Command Automation System
CAT	Crisis Action Team
CINC	Commander-in-Chief
CONOPS	Concept of Operations
CT&E	Certification Test and Evaluation
DAA	Designated Approving Authority
DISA	Defense Information Systems Agency
DSSO	Defense Systems Support Organization
DT&E	Developmental Test and Evaluation
ESC	Electronic Systems Center
FORSCOM	Forces Command
INFOSEC	Information Systems Security
JOPES	Joint Operations Planning and Execution System
JCS	Joint Chiefs of Staff
LAN	Local Area Network
MLS	Multilevel Security
NSA	National Security Agency
OT&E	Operational Test and Evaluation

SDN	System Development Notification
ST&E	Security Test & Evaluation
TPFDD	Time Phased Force and Deployment Data
USCENTCOM	U.S. Central Command
WASSO	WWMCCS ADP System Security Officer
WIN	WWMCCS Intercomputer Network
WWMCCS	Worldwide Military Command and Control System

REFERENCES

- [1] Defense Information Systems Agency, "Certification and Accreditation Plan for the USCENTCOM WWMCCS Guard," DISA/TFDA, March 1993.
- [2] Department of Defense, "Security Requirements for Automated Information Systems," DoD Directive 5200.28, March 1988.
- [3] Joint Staff, "Management Procedures for the WWMCCS Standard ADP System and the WWMCCS Information System," Joint Publication 6.03-11.
- [4] Joint Staff, "Security Policy for the WWMCCS Intercomputer Network," JCS Publication 6-03.7, 14 January 1993.

A CONCEPT FOR CERTIFICATION OF AN ARMY MLS MANAGEMENT INFORMATION SYSTEM

Victoria P. Thompson
F. Stan Wentz
Program Management Office
Reserve Component Automation System
8510 Cinderbed Road
Newington, VA 22122-8510

ABSTRACT

The Reserve Component Automation System is an Army management information system designed to support multilevel secure (MLS) operations. The process of certifying the system for generic accreditation for MLS operation presents challenges which require innovative solutions. The RCAS Certification and Accreditation Team has developed and is implementing a concept for assessing the technical security features and obtaining appropriate assurances through program-sponsored independent test and analysis of the trusted computing base.

INTRODUCTION

With increasing Department of Defense interest in trusted systems, particularly those designed to operate in a multilevel secure (MLS) mode, there is a growing need for new approaches to security certification and accreditation. System developers and accrediting authorities need to be able to assess the trustedness of systems in light of their unique requirements and environments. There is increasing recognition that even systems incorporating products that have successfully completed National Computer Security Center (NCSC) evaluation require further testing and analysis to meet system-level assurance requirements for accreditation. Quite often, the NCSC component evaluation

process is unable to keep pace with technology change and the requirement to field state-of-the-art systems. Even when a component has been evaluated, the assurances provided may not be applicable to a specific implementation or integration of the component into a system. Therefore, it is impractical and inappropriate to rely solely upon NCSC component evaluation to provide assurances necessary to support system certification. Faced with this dilemma, the RCAS Program Management Office (PMO) and the accreditation support team provided by US Army Communications-Electronics Command (CECOM) developed a concept for assessing the security features and assurances needed for MLS accreditation.

The Reserve Component Automation

System is a nation-wide computerized management information system designed to support the day-to-day peacetime office automation needs of the Army National Guard and the Army Reserve, as well as provide for planning and execution of mobilization of the Reserve Component forces. It will be fielded under the aegis of the National Guard Bureau to some 9,800 units at 4,700 locations in the continental United States and abroad.

The RCAS was acquired in accordance with Office of Management and Budget Circular A-109, whereby the Government identifies functional requirements but does not specify a technical solution. This process encourages innovative, yet cost-effective, industry response. Three key aspects of the RCAS acquisition are a Congressional prohibition against the use of Government furnished equipment, strict adherence to all applicable Government standards, and emphasis on the use of commercial-off-the-shelf (COTS) products.

The contract for development of the RCAS was awarded to the Boeing Company to provide a system with the necessary security features for Army accreditation for multilevel secure operation. Secret, confidential, and sensitive-unclassified information will be processed on the system, with all users cleared to at least the confidential level.

The RCAS is the first Army management information system to

seek generic accreditation for MLS mode of operation. Although regulations clearly identify the components of such an accreditation, the process for obtaining it is not specified. Since no other large Army management information systems have been accredited for MLS operation, no precedent or "lessons learned" were available. As a result, the PMO RCAS has had to develop a path to generic MLS accreditation. This paper describes the concept used for the security certification, a key component of the generic MLS accreditation of the RCAS.

BACKGROUND

Accreditation of Army systems is governed by AR 380-19, Information Systems Security, which states:

Accreditation is the DAA's formal declaration that an AIS or network is approved to operate--

- (1) In a particular security mode.*
- (2) With a minimally prescribed set of technical and nontechnical security safeguards.*
- (3) Against a defined threat.*
- (4) In a properly secured area in a given operational environment.*
- (5) Under stated short- and long-term goals.*
- (6) At an acceptable level of risk for which the accrediting authority has formally accepted responsibility.*

AR 380-19 identifies two categories of acceptable accreditation: generic accreditation of centrally fielded

systems, and operational accreditation of systems procured locally. Generic accreditation is appropriate for the RCAS because it is being fielded to multiple users by a single agency. Generic accreditation ensures that security considerations are addressed for the system as a whole during its development and throughout its operational lifecycle, regardless of the size and diversity of the user community. The generic architecture is certified for processing in the projected local operating and risk environments prior to fielding, rather than at each installation. The effect is to lessen the administrative burden on the field users, although they are still responsible for ensuring that the system is operated under the terms of the accreditation.

The decision to accredit a system is based upon the results of a certification and risk management review. Certification is the technical evaluation which verifies that the system performs the security functions that support its mode of operation and security policy. The certification process is intended to confirm that appropriate security features and functions are present and working properly, and provides assurance that they are correctly designed and implemented. For multilevel systems, the certification must be particularly sensitive to providing a strong assurance that the system can reliably separate users from data for which they are not authorized. Results of the certification feed a risk management review. If practical, procedures may be identified to reduce

or eliminate identified risk. Residual risk is then evaluated to determine if it can be assumed by the designated accreditation authority (DAA), permitting the system to be accredited for operation.

As the RCAS is an MLS system, AR 380-19 directs that the DAA is the Army Director of Information Systems for Command, Control, Communications and Computers (DISC4). The DISC4 opted to delegate security certification authority and responsibility for developing the RCAS accreditation package to the Program Manager of the RCAS as part of his responsibility for overall system certification. An RCAS Certification and Accreditation Team (RCAT) was formed to assist the program manager by providing independent technical assessment of system security, beginning with the certification process.

THE RCAS CERTIFICATION PROCESS

The first step in the RCAS security certification process was to identify all system security requirements against which testing would take place. Security requirements for the RCAS were defined early in the acquisition with the guidance of an Information Systems Security Task Force (ISSTF), made up of representatives of the Army, Department of Defense, and national security communities. These requirements address perceived threats to management information systems, as well as compliance with relevant

regulations and standards (notably AR 380-19 and DoD 5200.28-STD). In addition, a number of functional requirements influenced the security design. For example, the RCAS must process up to secret information, yet all users are not authorized access to all information on the system. Another functional requirement is that users be able to access all information for which they are authorized from a single point of entry into the system, with a single logon. Security requirements and security-relevant functional requirements together form the RCAS system security requirements which define the system security policy. As a matter of interest, the functional needs of the user community, rather than technical security requirements, resulted in the need for an MLS system.

Once all system security requirements were identified, the system architecture was analyzed to determine the mechanisms intended to meet the requirements. These mechanisms constitute the RCAS trusted computing base (TCB). Appropriate procedures for testing the features and functions of the TCB against the requirements were developed. Consistent with the philosophy of considering security certification as part of overall system certification, the security test and evaluation were integrated with the Government's system technical test program. System level security test results were reported both as part of the RCAS Technical Independent Evaluation Report and in security certification documentation. This

acknowledged that security is an integral part of the system, while allowing the RCAT to focus on the security certification-relevant results.

Technical testing can demonstrate that the required features are functional. However, it cannot provide all of the assurances needed for MLS accreditation. Assurances provide confidence that the TCB works as intended and only as intended, and is relatively tamperproof. Testing and analysis in support of assurances are intended to verify that no design or implementation flaws which violate the security policy exist. This goes beyond testing of features, which verifies only that the required mechanisms are functioning. Ideally, much assurance is derived from the use of components which have been successfully evaluated by NCSC. AR 380-19 points out that the system certification process can be simplified and expedited through the use of NCSC-evaluated components, although the specific implementation of those components must still be validated. The operating systems which form the foundation of the RCAS TCB are undergoing NCSC evaluation, however the evaluations are not expected to be completed in time to support system accreditation. NCSC evaluation schedules are typically independent of program system development schedules. Programs cannot assume that products not currently on the Evaluated Products List will have completed the NCSC evaluation process within required fielding times. While NCSC evaluations provide an excellent assessment of the

trustedness of components, they do not address those components as part of a configured system; nor do they address the use of other software on the system. The environment in which an evaluated component is to operate receives limited consideration. In short, NCSC product evaluations do not provide an assessment of trust for the entire management information system. Furthermore, Boeing has developed a number of trusted applications which are part of the TCB for which no NCSC evaluation process exists. These considerations necessitated the formulation of a plan for obtaining the requisite assurances that did not depend upon NCSC evaluation. With the concurrence of the DAA's staff, the RCAT determined that an independent, in-depth evaluation of the RCAS TCB components along with system level testing and analysis could provide the necessary assurances for the RCAS.

The plan calls for a three-part approach to certification of the RCAS: evaluation of COTS TCB components, evaluation of Boeing-developed TCB components, plus system level evaluation. At all stages of this process, emphasis is placed upon those security features and assurances most applicable to the RCAS.

First, the COTS TCB components as implemented in the RCAS were subjected to detailed test and analysis in order to develop the necessary component level assurances for the RCAS. This required access to developer design documentation and development tools to permit the RCAT

to understand and validate the implementation of security features in these products. Much of the necessary information is highly proprietary, reflecting the newness of the technology. In some cases, the required information had not yet been formally developed, necessitating the acceleration of production schedules. The RCAT also required an access channel to component developers for any additional information or clarification of issues. This information and access is not normally commercially available and had to be negotiated and funded through the prime contractor. RCAS hardware and software configurations were provided to the RCAT evaluators to use in validating the documentation and testing the product. Testing included active challenges to the product security features, as well as verification that all security features operate as described in the product documentation. The intent of the effort was to determine that security features are correctly designed, properly implemented, and tamper resistant, providing component level assurances.

For the Boeing-developed trusted applications, component level evaluation to provide assurances was simplified. System developers consciously kept the amount of trusted applications support software to a minimum, and were sensitive to the security issues associated with the development of trusted code. While the scope and intent of test and analysis were comparable to the assessment of the COTS TCB

components, the necessary information was more easily accessible by the RCAT. All software developed by Boeing for the RCAS is the property of the Government and is developed under the supervision of the Government. As code is developed, the RCAT is able to participate in software design reviews and code walkthroughs, and has access to programmers and their software development folders. The closed development environment, to include programmer qualifications, is monitored as part of the assessment of code integrity. The RCAT witnesses and provides input to the Boeing testing process, monitors the configuration management process, and performs in-depth testing of the applications. As with the COTS TCB components, this testing and analysis provides assurances that the security features are present, working properly, and cannot be exploited or misused.

Finally, the fully integrated system is subjected to testing as part of overall system testing at the RCAS independent testbed at the Institute for Telecommunications and Sciences in Boulder, Colorado (ITS). The RCAT testing team, in concert with the ITS testers, executes test procedures and scenarios developed against the system requirements. The focus of this testing is on the system security requirements as information moves through the system. This testing provides additional verification that the components already assessed individually function properly when integrated. The testbed allows linking of different representative system

configurations to simulate RCAS LAN and WAN connectivity. An effort is made to replicate actual operating conditions in order to develop and document appropriate procedural security measures.

RCAT evaluation of these three elements (COTS TCB components, Boeing developed TCB components, and the integrated system) provide essential system and component level assurances that security is properly implemented within the RCAS. It is acknowledged that the assurances developed according to this plan are unlikely to be relevant to any other system, even if the same components are used, since the requirements and environments of other systems would differ from those of the RCAS. However, the concept of developing system level assurances through Program-sponsored independent test and analysis of components and of the integrated system followed by the RCAS is applicable to other systems. The RCAS experience has demonstrated that the process is feasible and flexible enough to support generic MLS accreditation.

The process of obtaining assurances through Program-sponsored independent test and analysis is more easily described than implemented. A number of valuable lessons are being learned in the course of the RCAS certification. It is essential to identify and obtain the services of qualified evaluators as early as possible in order to adequately plan the certification and accreditation, and to take advantage of

the system development effort. It is imperative that consensus and cooperation on the parts of the DAA, the system developer and their subcontractors, the evaluators and the program management team be obtained. The certification and accreditation process should be anticipated early in the acquisition process so that materials necessary for the certification are specified in the statement of work. Resource requirements, including system configurations, support personnel and funding, should also be anticipated. Program schedules must recognize and allow for security certification and accreditation process, since actual operation of the system depends upon successful accreditation.

CONCLUSION

As of this writing, the process described above is underway but not yet complete. The concept of Program-sponsored independent test and analysis in support of generic MLS system accreditation has met with approval of the DAA and is providing flexibility to meet the accreditation requirements of the RCAS within the wider programmatic framework and schedule.

A number of organizations and individuals are involved in this certification effort. They include the RCAS PMO and Boeing Information Systems Security Engineering teams, the staff of the DISC4 Information Systems Security office, and the NGB Information Systems Security Program

Manager. Primary security testing and analysis support is provided, through the Army Communications-Electronics Command, by a talented and expert team of support contractors. All RCAS security efforts are undertaken with the advice and assistance of the ISSTF. The intense interest and scrutiny to which the RCAS certification approach is being subjected lead us to believe that the RCAS process, if not the results, may be useful elsewhere in the DoD community. To this end, all parties involved have made a conscientious effort to develop and implement an innovative, practical, and replicable certification process which conforms to accepted standards and policy.

CERTIFICATION AND ACCREDITATION APPROACH

Keith P. Frederick, Capt, USAF
Air Force Cryptologic Support Center (OL-FP)
203 W Losey ST Room 1016
Scott AFB IL 62225-5223
DSN 576-8707 / COM (618) 256-8707

INTRODUCTION:

The purpose of this paper is to present an approach for the Certification and Accreditation (C&A) of an Automated Information System (AIS). This approach was developed from the lessons learned during a large and comprehensive C&A effort. However, it may be scaled, in places, in order to accommodate a program of smaller dimensions or of reduced associated risk. Because the information presented herein is intended to serve as a template, it presents a high-level analysis of a C&A program so that the steps may easily be applied and tailored to other C&A efforts.

Justification. Currently, Department of Defense (DOD) policy toward AISs are that automated information systems must undergo an official technical assessment and approval before they are allowed to process classified or sensitive unclassified information. This approach may assist others in accomplishing a C&A effort in lieu of no DOD Directive or Air Force Regulation (AFR) describing the needed steps for C&A of an AIS. The approach is subject to, and consistent with: AFR 205-16, Computer Security Policy [United States Air Force (USAF), 1989], which applies to all USAF activities using AISs; DOD Directive 5200.28, Security Requirements for Automated Information Systems (DOD, 1988b), which is mandated for all DOD sensitive, classified, or critical AISs; and DOD 5200.28-STD (DOD, 1985b), Department of Defense Trusted Computer System Security Evaluation Criteria, as it pertains to product evaluations.

Certification and Accreditation Overview. USAF AFR 205-16 (USAF, 1989) identifies three separate processes that must be completed in order to assess and approve USAF AISs: risk analysis, certification, and accreditation. Risk analysis and

certification are technical activities, whereas accreditation is a management activity. During the risk analysis process, risk is determined and qualified by assessing system value, threats and vulnerabilities, cost-versus-benefit of the security measures, and the test and evaluation results of the security features. During the certification process, the results of the risk analysis process are compiled and analyzed, and residual risks are documented. During the accreditation process, the Designated Approving Authority (DAA) grants or denies permission to operate the system in a specific security mode of operation and environment, based on the level of residual risk. These three processes are generically and collectively referred to as certification and accreditation.

Security Modes of Operation. A security mode is defined as a mode of operation in which the DAA accredits an AIS to operate. For collateral information, DOD Directive 5200.28, Security Requirements for Automated Information Systems (AIS), dated 21 March 1988, defines the four security modes: dedicated, system-high, partitioned, and multilevel. Inherent with each are restrictions on the user clearance levels, formal access requirements, need-to-know requirements, and the range of sensitive information permitted on the AIS.

CERTIFICATION AND ACCREDITATION (C&A) APPROACH:

The C&A approach is implemented as a chain, and like all chains the weakest link (steps) will determine the adequacy as well as the stability of the chain (total certification effort). Figure 1 illustrates the C&A approach which provides for C&A activities throughout the life cycle. The actual thoroughness carried out in each step is dependent on the answers to many questions. Some of those questions are:

- What is being accredited?
- How large is the Automated Information System (AIS)?
- What is the mission and the role of the AIS?
- How much time will the testing effort be given?
- How well the responsible personnel understand and are capable of handling their roles and responsibilities in the C&A of the AIS?

The answer to the question mentioned above on the expertise of the responsible personnel will be the most critical factor in determining not only the thoroughness, but also the extent of testing which can be done. All testing which is to be done, is a reflection of the negotiated agreement between the wants (assurances) and needs (directives and regulations) of the "DAA" (accreditor) and the capabilities and commitment of the "test team" (certifier). (**NOTE:** Many of the steps in this C&A Approach can be accomplished in parallel. Read each steps description for clarification.)

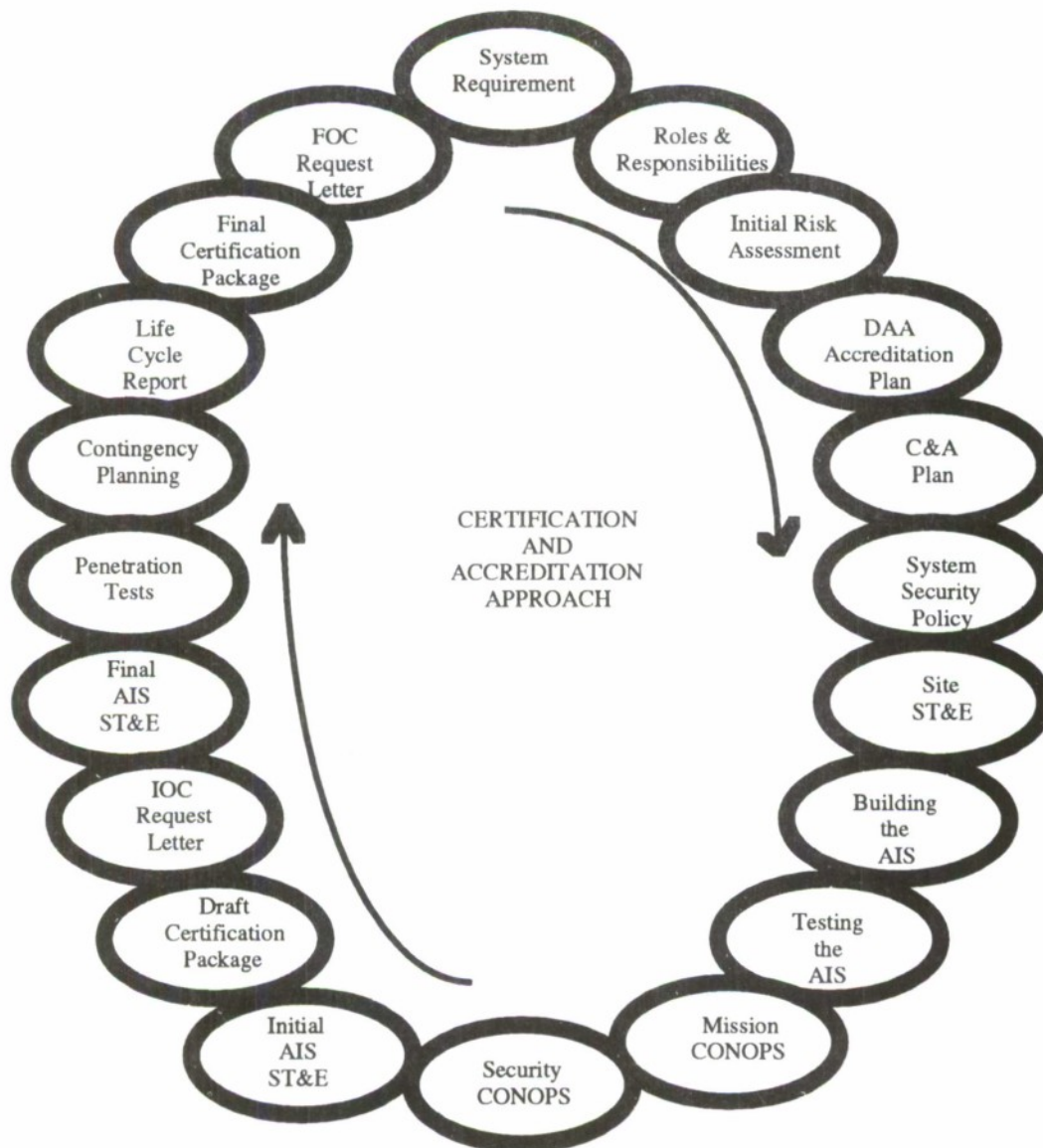


Figure 1

SYSTEM REQUIREMENTS:

The system requirements document captures the requirements of the system as well as the users. It is the key to the entire program. It bonds succeeding steps together to harden the overall chain. There are many positives in doing the system requirements document right. Some of those positives are: it gives the certification team a standard to measure the AIS test results against, it guilds the vendor in constructing the AIS, and it guilds the agent producing the System Security Policy. On the other hand, if this document is inaccurate, incomplete, or missing, the program will be haunted by these shortcomings during its entire lifetime (e.g., the System Security Policy as well as the Concept of Operation (CONOPS) document ask the question, "What is the required information flow?"). The point is requirements have to be determined and written down in a requirements document. If not, they will be "determined" over and over again. The security requirements implied by the general system concept should be noted for comparison with those brought out in the System Operational Concept and the Security CONOPS. Implications for system accreditation may be inferred from the identified user agency and the categories of information to be stored and processed by the system.

Minimum system security requirements are specified in DODD 5200.28 and manual DOD 5200.28-M. The DODD 5200.28 describes the minimum security requirements for an AIS in enclosure 3 and establishes uniform guidelines for techniques and procedures to be used when implementing, testing, or evaluating the security of an AIS. Additional requirements, over and above the minimum requirements, and the determination of the system security mode of operations are based on results of the initial risk assessment procedures described in enclosure 4 of DODD 5200.28. Similarly, AFR 205-16 translates these security requirements into a mode of operation for an AF system and provides guidance on choosing the appropriate level of trust as defined in DOD-5200.28-STD [i.e., the Trusted Computer System Evaluation Criteria (TCSEC)], commonly referred to as the "Orange Book"].

The TCSEC provides a metric which may be used to assess the degree of trust of a system. It specifies seven classes of systems (D, C1, C2, B1, B2, B3, A1), ranging from no security to verified design. Because the TCSEC specifies minimum security requirements for general-purpose operating systems, it is necessary to determine additional requirements based on user, system, and mission needs as well as environmental considerations.

ROLES AND RESPONSIBILITIES:

It is important that the various personnel/organizations and their roles and responsibilities be identified early in the C&A process. The identification of personnel/organizations and their roles and responsibilities will clearly communicate who is responsible for each C&A activity to the DAA and all concerned. This information should be included in the C&A Plan.

The role and responsibility of the DAA and the delegation of responsibilities to the many organizations involved in a certification effort and the interrelationships among these organizations has varied from program to program. The following list provides an overview of typical assignments of responsibilities to the various participants. The number of participating organizations and their assignments will differ between programs based on the guidance set forth by the DAA, the availability of resources, apportioned level of effort for certification, and the security requirements, sensitivity, and criticality of the system.

Accreditor. This responsibility is usually assigned by default to a management position in the organization. This person has the largest effect on the scope of work in the C&A of an AIS. The DAA is the driving force and the final arbiter as to what is included in the C&A Plan and must, therefore, be identified early in the process. The DAA is also responsible for approving many other documents like the requirements documents and security policy. It is important to identify all the DAAs who have a vested interest in the system being procured (e.g., the DAA of each using command, the DAA responsible for the communications network, and the DAA

responsible for the protection of special classes of data). It is also important that any decisions made during the procurement of the AIS which constitute departures from the original system be coordinated with the DAA(s) and documented with supporting rationale.

Each MAJCOM commander is responsible for delegating DAA responsibility for systems that process collateral information for that command or agency. When several MAJCOMs use the same system, they must develop an memorandum of agreement (MOA) to identify individual security responsibilities. Establishing the responsibilities among the agencies and defining the DAAs' requirements for interconnected collateral AF AISs will require the development of an MOA in accordance with DOD Directive 5200.28 and AFR 205-16.

CERTIFIER. This responsibility needs to fall to the most technical security capable personnel in the organization. This group has the second largest effect on the scope of work in the C&A of an AIS.

The Certification Manager is responsible for defining and managing the Security Certification Program. Although AF regulations do not identify the organization responsible for performing certifications, AFR 205-16 states that many of these activities may be delegated by the DAA to a Certification Manager. The recommendation is that the DAA choose the Certification Manager since the DAA must work closely with this individual. The Certification Manager is the formal certifying authority (i.e., Certifier) for the program, and will work very closely with the DAA to ensure that the certification plans appropriately address the goals of the accreditation. This individual will also work closely with the Program Manager to assure that the appropriate overall program planning and programmatic concerns are consistent with those for certification. The Certification Manager may delegate the responsibilities of the Certifier to various certification support groups.

The Certification Manager and/or Program Manager (PM) should, when necessary, seek advice from other related government agencies and

contract organizations to support the certification or system development activities. In providing information about system threats, vulnerabilities and security policy guidance, and in ensuring satisfactory security engineering throughout the acquisition life-cycle, these contractors and agencies provide expertise in conducting the necessary technical evaluations for certifying the system. In some cases, a contractor may be tasked with a specific aspect of the certification (e.g., penetration testing, a trusted application, or formal verification analysis) or they may be tasked with a broader charter to support the entire development cycle as a participant in the efforts of a security evaluation team or Security Working Group (SWG).

END USERS. The end users support the certification process via input to the statement-of-need, operational concepts, and security requirements to the PM and Certifier. The role of the end user is distinguished from that of the operational user; the end user performs the mission, whereas the operational user support the end users by managing the system performing the mission.

OPERATIONAL AND SUPPORT PERSONNEL.

This group of users may have multiple titles and roles. However, the positions of interest for certification are those associated with securely managing the operation of the computer/network facility. The overall responsible individual is the Computer Facility Manager (CFM) who will assign other facility support positions as required [e.g., Computer Systems Security Officer (CSSO), Terminal Area Security Officer (TASO), Network Security Manager, Trusted Programmers/Analysts, etc.]. Many of the responsibilities of the support personnel affect the security of the operational computer system. The support personnel can also be the key in defining the physical environment and they will directly participate in any recertifications after the system has become operational by supporting the CFM's role.

Typically, the CFM institutes the security procedures and measures for the computer facility and obtains written approval from the DAA to process information. To obtain the

approval to operate, the CFM must ensure that the computer facility satisfies the security specification for the highest level of sensitivity and criticality for the systems or data processed and must annually review the security implementation to assure that it meets the terms of the DAA accreditation.

In addition, the CFM is responsible for maintaining the Trusted Facility Manual and recertifying the adequacy of the computer security measures at least every three years or when the system is significantly modified. The CFM also ensures that all personnel report security incidents and that the CSSO or TASO identify and report security problems and vulnerabilities.

SYSTEM DEVELOPER. The System Developer may be either a contractor or an AF development group that is responsible for designing and implementing the system. The System Developer must prove that the system meets the prescribed set of security requirements through testing and documentation as defined in the contract. Often there are multiple System Developers and subcontractors in addition to an overall integration contractor involved in a program. The relationship among the different developing organizations will vary from program to program, and their responsibilities in the certification process will depend on contractual arrangements. The end result should be an effective security development structure for the program.

SECURITY WORKING GROUP (SWG). The goals of the SWG are to provide a forum for communications among the Government Agencies, System Developer, Certification Manager, certification support agencies, and the DAA(s). The responsibility of the SWG is to support the certification process and the accreditation of the system. The SWG will ensure adequate security review and evaluation coverage of all security engineering tasks throughout the development life-cycle. Members of the SWG may perform security trade-studies to evaluate the merits of different security architecture and engineering approaches throughout the planning, design, implementation, and testing phases of the program.

Early formation [i.e., prior to Request for Proposal (RFP) development] of an

SWG is strongly recommended. They will provide:

- The forum needed to formulate the security concept of operations, to identify the security requirements and other RFP inputs, and to develop a security certification/accreditation plan, and to resolve security-related issues prior to RFP release.

- A security knowledgeable staff to evaluate the proposals during source selection.

- A team that will review the contractor-developed documents, monitor the design and implementation of the security requirements, analyze issues, and make recommendations after contract award.

System Program Office (SPO)/ Program Manager (PM). From a security point of view, the SPO or PM is responsible for creating an SWG and ensuring the DAA and users participate throughout the system development cycle in security analyses performed in conjunction with all design and specification reviews. The SPO is responsible for ensuring the appropriate coordination and review of all decisions concerning security trade-off and changes in requirements with the PM, Certification Manager, System Developers, users, and the DAA. In cases where the AF is developing the system themselves, there may only be a PM.

The PM, with help drawn from other agencies and organizations, is the focal point for the certification support. This individual is responsible for conducting the risk analyses, evaluating the adequacy of the security system engineering during the system requirements definition, design, implementation, and testing phases of the program. Most importantly, the PM is responsible for ensuring that the security measures implemented adequately satisfy the security specification and that any residual risks are identified.

Air Force Cryptologic Support Center (AFCSC). The AFCSC is the Office of Primary Responsibility (OPR) for providing guidance on the implementation of computer security policy for the Air Force. In this role, they provide technical assistance

to MAJCOM requests for certifying systems.

Defense Information Systems Agency (DISA). The DISA is in charge of the Component Approval Process (CAP) which oversees all AIS programs excluding command and control, weapon systems programs, and Intelligence programs.

National Security Agency (NSA). The NSA develops computer security standards and guidelines, provides technical evaluation support, advice, and assistance to DOD components, conducts evaluations of industry and government-developed trusted computer systems intended for DOD use, conducts product and system profiles, and publishes and maintains the Evaluated Products List.

Other Agencies involved in Computer Security Policy. There are offices of responsibility for computer security policy within the AF, DOD, and the Intelligence agencies that may influence the security requirements for a program. The roles and responsibilities are delineated in AFR 205-16.

INITIAL RISK ASSESSMENT:

The program office should prepare an initial risk assessment for any proposed system that will process sensitive unclassified or classified data. This assessment should be based on a risk analysis that identifies impacts related to information compromise, loss of integrity, or lack of system availability. The analysis should consider all types of threat sources including accidental and intentional acts, natural disasters, and environmental failures. However, the preliminary risk assessment itself is optimally limited to threats that the technical characteristics of the system design can mitigate. The initial risk assessment should be based, at least in part, on the guidance stated in enclosure 4 to DODD 5200.28 and Chapter 6 of AFR 205-16.

The initial risk assessment can also be used to assess not only risk but also sensitivity, criticality, and economic viability. These assessments should not be confused with those accomplished in the final analysis performed at the end of certification testing. The initial risk assessment, at a minimum,

should describe the role of the AIS, defining the security mode of operation, and risk index (trust level) based on the levels of classification of its information and the clearance levels of all personnel needing access to the AIS. Economic viability plays a key role in determining whether or not the program can afford to reach its needed required trusted level. Lacking the resources or time to build a product or being constrained to using commercial off the shelf (COTS) and government off the shelf (GOTS) products will definitely restrict the programs ability to reach its needed trust level. Any inability to meet a needed level-of-trust should be stated as well as how the program will handle this short-coming of the AIS.

DESIGNATED APPROVING AUTHORITY (DAA) ACCREDITATION PLAN:

This plan is the documented results of the agreement negotiated between the certifier (test team) and the accreditor (DAA). The agreement spells out the minimum requirements needed by the DAA to accredit the AIS at initial operational capability (IOC) and final operational capability (FOC). Approved documentation includes identification of data sensitivity levels, system sensitivity and criticality, modes of operation, system security measures, local security measures, and security certification documentation. AISs may also be **type accredited** by the DAA whenever multiple copies of the system are to be fielded. More on type accreditation can be found in AFSSM-5004.

Interim Approval. An AIS must have the DAA's written approval before operations can begin processing in a specific facility. Interim approval allows the activity to meet its operational requirements temporarily, while further assessing and improving computer security (COMPUSEC). The DAA may use less stringent criteria to grant interim approval (e.g., interim approval allows the activity to meet its operational requirements while assessing and improving its computer security posture). This gives the DAA the latitude to approve operational use of individual components of a computer system as they are being developed. Full implementation requires final accreditation. Interim approval is only granted for specific time periods

and must not exceed a cumulative total of **two years** (reference AFR 205-16 and AFSSI 5026). Interim approvals should be reviewed semiannually to make sure satisfactory progress is being made toward final approval.

Final Approval. Accreditation is specific to the site and is dependent on local security measures and procedures. After a computer system receives its final approval to operate, it must be subject to the normal re-accreditation process. According to AFR 205-16 and AFSSI-5026, the AIS needs re-accrediting every **three years**. (Reference AFR 205-16 and AFSSI-5026 for exceptions.)

CERTIFICATION AND ACCREDITATION **(C&A) PLAN:**

A C&A Plan, also called the Security Certification and Accreditation Plan (SCAP), is a step-by-step description of all the activities to be accomplished to at least fulfill the requirements agreed upon by the accreditor and certifier in the DAA Accreditation Plan. It describes the risk analysis, certification, and accreditation processes. It identifies all relevant tasks and the responsible organizations and time frames for completing the tasks. Upon completion of these tasks, a judgment or a statement of opinion on the security and assurance of the program will be produced.

This plan addresses all aspects of security, including physical, administrative, personnel, COMPUSEC, communication security (COMSEC), operations security (OPSEC), and emanations security (TEMPEST) for the program's C&A effort. It identifies and describes the tasks associated with assessing and certifying the system against:

- DODD 5200.25 STD criteria (DOD,1985)
- Program security requirements
- AFR 205-16 (USAF, 1989)
- Supplements to AFR 205-16

It also incorporates risk analysis, certification, and accreditation requirements from: AFR 205-16, Computer Security Policy (USAF, 1989), which applies to all USAF activities using AISs; DOD Directive 5200.28, Security Requirements for Automated Information Systems (DOD, 1988b), which is mandated for all DOD sensitive, classified, or critical AISs; and DOD 5200.28-STD (DOD, 1985b), Department of Defense Trusted Computer System Security Evaluation Criteria, as it pertains to product evaluations.

SYSTEM SECURITY POLICY:

The system security policy will interpret DODD 5200.28, AFSSI 5001, and AFR 205-16 as they apply to the specific system by defining the protection and access control requirements to be enforced by the various security provisions, including COMPUSEC features. The security policy should describe the classification of the information and any needs for special handling of the information. It should include any special interpretation of DOD security requirements and the rationale for any deviation of security standards or practices. The traditional system security policy is a translation of downward directed requirements (DOD Directives and AFR) and the system security requirements into security statements that must be met by the system design and implementation. The advantage to this traditional way is it closely correlates to the system requirements and, therefore, the certification team has a standard to measure the test results against. The disadvantage is the end users have a hard time correlating the policy to their day-to-day mission/function. A new end user balanced system security policy is a merger between a strongly reflected end user's day-to-day mission/function, directed requirements (DOD Directives and AFR), and system security requirements. The advantage to this new way is it closely correlates to end user's day-to-day mission/function and, therefore, the end user not only can understand but implement the policy at their level. The disadvantage is the responsible agent for producing the policy has to spend more time with the end users to thoroughly understand the end user's day-to-day mission/function. Experience has shown the more the end

user understands the system security policy, the more likely it will get implemented and the certification team should use the system security requirement as the standard to measure the test results against. Either way the system security policy is done, it must be sufficiently broad so as to not force a specific design. It should not be so detailed as to mandate that a requirement be met by a hardware solution as opposed to administrative controls. It will be a living document that requires updating throughout the life cycle. The certifier should ensure that the system security policy, as an interpretation of DOD security policy, is adequate for system design purposes and that there are no internal inconsistencies within the stated policy.

An MOA is accomplished when two AISS connect (AFR 205-16). That MOA should be included as an appendix to each of the System Security Policies as exceptions. This explanation of the connectivity not only amends the System Security Policy, but also defines circumstances for which information can be exchanged between the two AISS. When two AISS intersecting have the same DAA, a memoranda of record (MOR) needs to be produced to document the System Security Policy exceptions and should be made part of each policy.

SITE SECURITY TEST AND EVALUATION (ST&E):

A site ST&E is done to measure the environment in which the AIS is to operate. Requirements specified in the site ST&E include administrative, physical, personnel, OPSEC, COMSEC, and TEMPEST. For the most part, these are the requirements required for site certification. That is, those issues that deal with the security safeguards and controls applicable to the operating environment. This information is used in setting up countermeasures for vulnerabilities found during the AIS ST&E.

The scope of a site ST&E may extend to include operational considerations not covered in other tests. For example, the adequacy of physical security measures such as guards and cipher locks; the effectiveness of operational procedures such as downgrading of outputs, review of audit logs, and incident reporting; and the adequacy of

emergency procedures for events (i. e., fires or floods). This ST&E can be performed at anytime up to the point of assigning countermeasures in the Risk Analysis Report.

Administrative Security. Involves establishing and managing procedures that implement security policy apart from the computer and network system. An administrative structure is necessary to ensure that technical, personnel, and physical security measures are present to enforce non-AIS safeguards. Administrative procedures are usually established for personnel clearances, password management, handling classified information, security training, reporting security violations, etc. The policies governing administrative security are prescribed in DODD 5200.1R and AFR 205-1.

Physical Security. Includes installation access controls as well as the physical security of facilities. It should be commensurate with the minimal requirements for the open storage of the highest classification of information processed in the system. Plans for the protection of personnel, equipment, and physical property against vulnerabilities and threats should be described. Physical security is used to prevent unauthorized access to equipment, facilities, material, and information. The policies governing physical security are prescribed in DODD 5200.8 and AFR 207-1.

Personnel Security. Is the set of procedures established to ensure that access to classified information within the system has been granted only after a determination of a person's trustworthiness has been established and a valid need-to-know exists. It should provide that an individual has the proper security clearance and need-to-know before granting him or her access to classified information. The personnel security program is defined by DODD 5200.2R and is implemented for the AF by AFR 205-32.

Operations Security (OPSEC). Includes those measures needed to ensure the continued security of the activity worked to accomplish its mission. It involves the protection of information concerning sensitive military activity. The methods for guaranteeing OPSEC include, but are not confined to,

administrative, physical, COMSEC, and COMPUSEC. The governing directive for OPSEC is DODD 5205.2 and for AF collateral information is AFR 55-30.

Communications Security (COMSEC). Is defined in DODD 5200.5 as "protective measures taken to deny unauthorized persons information derived from telecommunications of the U.S. Government related to national security and to ensure the authenticity of such communications. Such protection results from the application of security measures (including cryptosecurity and transmission security) to electrical systems transmitting national security or national security-related information. It also includes the application of physical security and other measures to COMSEC information or materials." Other significant COMSEC regulations include AFKAG-1 and AFR 56-50.

COMSEC is used to ensure secure communications links between AIS components or between separate AISs. That is, to deny unauthorized persons information derived from classified telecommunications and to ensure the authenticity of those communications. AF COMSEC policy requires systems transmitting classified information and sensitive unclassified information which could affect national security interests be secured with NSA-approved equipment. COMSEC includes the use of encryption, transmission security, and physical security of COMSEC material and information.

Emanations Security (TEMPEST). Is the study and control of decipherable electronic signals unintentionally emitted from equipment. It involves measures to control compromising emanations from the AIS and communications equipment. Failure to use TEMPEST countermeasures can result in the DAA denying processing approval. Significant TEMPEST directive/regulations are DODD S-5200.19, AFR 56-16, and several NSA pamphlets.

BUILDING THE AUTOMATED INFORMATION SYSTEM (AIS):

The AIS should be built according to specifications which fulfill the users operational requirements as well as the security requirements which will allow the operation to run with risk being

properly managed. The paper "Structure and Portability of the MLS/GDSS Application Security Kernel (ASK)" on the Air Mobility Command (AMC) Multilevel Security (MLS) Global Decision Support System (GDSS) at Scott AFB IL is an example of the building of an AIS.

TESTING THE AUTOMATED INFORMATION SYSTEM (AIS):

The testing tasks described in this section are an important step in the risk analysis process and a key activity in the certification process. There are four types of tests that will be considered during certification: Development Test and Evaluation (DT&E), Operational Test and Evaluation (OT&E), Integrated System Testing (IST), and ST&E.

Development Test and Evaluation (DT&E). The requirements that are tested during DT&E include those from the system specification (DT&E System Tests), as well as all the allocated requirements that the developer has included in the authenticated B-level specifications. The developer is responsible for generating test plans and procedures and conducting DT&E. In order to facilitate security certification, it is advisable that the developer consolidates security-related plans and procedures in specific sections of the DT&E documentation.

Operational Test and Evaluation (OT&E). With system developer support, OT&E is conducted by the government at the end of the development phase, with its primary focus towards determining the system suitability and effectiveness for its intended mission. The security aspects of OT&E are related to the operational or user point of view. It will include the impact and effect of administrative and operating procedures as well as other environmental factors. OT&E tests are not limited to the requirements stated in the system specification.

Integrated System Testing (IST). A special type of operational test, IST, is performed for those system components that will be integrated into a larger system such as local area network/wide area network (LAN/WAN) systems. The purpose of the IST is to confirm that the newly developed system integrates properly with the other

components, and does not have adverse effects on them.

If, as is frequently the case, the system being developed will be integrated into a larger configuration, OT&E must include planning for IST, which tests the interoperability of all system components. The objective of this test is to show that newly developed components can integrate into the existing system and do not cause any degradation on the overall performance or functionality. The certifier must assess the adequacy and competence of the IST so it properly tests the security features of the new system in its larger environment, as well as its potential impact on the security mechanisms of the other existing system components.

Security Test and Evaluation (ST&E).

ST&E is conducted by the government (or a government-selected independent verifier) in order to determine the adequacy of the security mechanisms. ST&E may be conducted as an adjunct to DT&E and OT&E, as a separate independent test, or both. The range and extent of ST&E is dependent on the type of information that the system will process, the operating security mode, the criticality of the information, and constraints on the testing schedule. The simplest ST&E consists of witnessing pertinent DT&E and OT&E activities and conducting a paper review of the results. A more rigorous ST&E will include limited, specialized but short tests in conjunction with OT&E, while a still more complete ST&E will include a distinct separate period for the sole purpose of conducting dedicated security tests, up to and including sophisticated penetration tests. Due to the normal schedule conflicts and funding concerns at this stage of final development, such specialized security tests can only be justified for those systems where security is a major concern.

SYSTEM CONCEPT OF OPERATIONS (CONOPS):

The CONOPS document and Security CONOPS document should be accomplished earlier in the program, but may be completed at anytime up to the initial ST&E. Earlier in the program, both CONOPS, at best, consist of one or two paragraphs

describing how operations will use this AIS to support the user's mission.

The system CONOPS document should be read and generally understood by the system certifier. It will provide valuable insight into the expected data flows into, out of, and within the system that the COMPUSEC features are expected to mediate. This document and/or its counterpart, the Security CONOPS, greatly influence the security requirements.

SECURITY CONCEPT OF OPERATIONS (CONOPS):

Some AF systems currently under development have produced a separate Security CONOPS document that describes at a conceptual level how the data will be protected in the course of system operation. It defines the assumptions concerning the AIS, its mission, and its environment that have potential risks to the information in the system or the success of its mission.

The Security CONOPS document should describe how the operations of the system takes place from a security point of view, the system criticality, and the impact of system failure. For the information to be stored and processed by the system, the document should address the classification range and expected volumes of data, the number and clearance level of system users, the user interface expectations and limitations, a summary of the security policy to be enforced by the system, and a description of the procedural and physical protection methods that will augment the security features of the system itself. The expected connections to other systems and the classification range of data imported and exported over those connections should be included. Sufficient information should be provided to perform a risk assessment and to determine a predicted residual security risk.

The Security CONOPS information directly influences the system certification and accreditation requirements. The certifier should evaluate the Security CONOPS, not only to assure that the appropriate scope of information is provided, but also to evaluate the feasibility of the operating concepts presented in the document with respect to AF security

regulations and the operational limitations of current security technology.

**INITIAL AUTOMATED INFORMATION
SYSTEM (AIS) SECURITY TEST AND
EVALUATION (ST&E):**

The initial AIS ST&E is accomplished prior to and in support of IOC. Final AIS ST&E, also called residual testing, is accomplished after IOC as part of the residual testing which is done in support of FOC. The thoroughness of the initial and/or final AIS ST&E is depended on the time and resources available. The initial AIS ST&E, due to time constraints, may just be to review of the other testing being accomplished (i.e., DT&E and OT&E) and the development of a thorough final AIS ST&E. (NOTE: A mockup of the system to test against would aid the AIS ST&E and the penetration testing.)

The purpose of this task is to ascertain that ST&E, a critical step in the risk analysis process, has been performed and performed correctly, that all potential vulnerabilities have been evaluated, and that the results have been properly documented.

The certifier must have a close involvement with ST&E from its early planning stages and be an active participant on the decisions regarding the overall scope of the tests and the entities or agency that will perform them. ST&E must be strongly justified by both the users and the certifier, since it will take place during the latest stage of development, when the program environment is normally impaired by the cumulative effect of previous delays and cost increases. The rationale for separate tests from OT&E must be clear and powerful enough to justify the extra cost and delays that ST&E will entail. The responsibility for the independent security testing of the system must also be assigned early in the project. The government must identify and task a suitable government representative or specialized contractor to plan and perform the independent tests. While it is important that security features are evaluated by other than the system developers, actual testing may be performed by members of the development team provided that all tests are witnessed by the independent testers or the users.

The next task for the certifier is to evaluate the ST&E planning documentation and testing approach. Depending on the degree of concern about security, the range of tests performed during ST&E varies greatly. In its most complete form, the Security Test Plan and associated test procedures must be designed to measure the system's penetration resistance, security fault tolerance, and internal security boundary effectiveness. The results of the component security tests (if any) and previous document reviews are used to identify areas requiring thorough security testing. Known vulnerabilities must be tested to determine the difficulty of exploitation and potential resulting damage. Testing should be designed to detect malicious hardware and software. For systems with stricter security requirements, the planning of more sophisticated security penetration testing is needed.

The certifier must play an active role in the performance of ST&E by providing representatives who will witness and monitor the execution of the tests. By necessity, there may be the need to deviate from the specific approved procedures in order to explore security flaws in unanticipated areas. It is important for the certifier to ascertain that the proper efforts and impromptu tests were performed to determine the extent of the vulnerabilities, the difficulties in exploiting them, and the potential resulting damage.

Finally, the certifier's review of the ST&E report should ascertain that it details all tests performed and includes the unsuccessful ones. The report must also identify all **perceived** vulnerabilities, provide an assessment of potential damages, and recommend new security measures and requirements that will mitigate their impact.

NOTE: Testing for assurances may not be practical but testing for compliance can be accomplished.

DRAFT CERTIFICATION PACKAGE:

Based on AFR 205-16, Figure A7-7, the DAA accreditation package includes certification letters and checklists. The needed documentation for IOC should be listed in the DAA accreditation

plan. The draft certification package is from the reviewing authority to the DAA with a recommendation to approve or disapprove system processing, which risks to assume, and proposed dates to correct deficiencies. Extracts from supporting documents provide the information the DAA needs to make the decision. The draft certification package has also been called the security certification and accreditation report. The purpose of this task is to complete and document the process of risk assessment that was initiated during the conceptual phase. Before the system becomes operational, this process is correctly referred to as risk analysis. The analysis, based on the initial assessment, should be an on-going activity during the development of the system. The conclusions of the risk analysis should incorporate the findings of the penetration testing and should also consider additional risks to the system such as natural disasters, structural failures, and intentional hostile attacks. In either case, the certifier should be in agreement with these results as they, in effect, form the basis for system certification and accreditation.

The results of this task should be a documented assessment of the security risks to the system including:

- The methods and personnel used to perform the assessment,
- A description of the system and its safeguards,
- An identification of potential threats and threat scenarios,
- An identification of high-risk areas,
- An analysis of the countermeasures and procedural safeguards considered and those added to the system,
- A summary of the security test and evaluation results, and
- Conclusions on the residual risks to the system and some estimate of the level of severity of the residual risks.

AFR 205-16, Chapter 4 and Attachment 8, provides guidance for the performance

of risk analysis. The major command supplements to AFR 205-16 provide further specific information and documentation requirements. The important conclusion, however, is that the risk analysis provides the basis for system accreditation. If significant risks are not identified and countered by appropriate system security provisions, then approved operation may leave sensitive information at risk of compromise. The certifier should be convinced that the risk analysis was performed adequately and that the conclusions of this work are comprehensive.

INITIAL OPERATIONAL CAPABILITY (IOC) REQUEST LETTER:

The IOC Request Letter is based on AFR 205-16, Figure A9-3. The DAA Accreditation package includes Certification Letters and Checklists. It is from the reviewing authority to the DAA with a recommendation to approve or disapprove system processing, which risks to assume, and proposed dates to correct deficiencies. Extracts from supporting documents provide the information the DAA needs to make the decision.

FINAL AUTOMATED INFORMATION SYSTEM (AIS) SECURITY TEST AND EVALUATION (ST&E):

The Final AIS ST&E, also called Residual Testing, is accomplished after IOC and is part of the residual testing which is done prior to and in support of FOC. The thoroughness of the final AIS ST&E is depended on the time and resources available. Due to time constraints in the initial AIS ST&E, the final AIS ST&E may have to be extremely thorough.

PENETRATION TESTS:

The purpose of this task is to determine if the penetration tests, and the circumstances in which they were conducted, were adequate for the security level of the AIS. It would be ideal to accomplish this testing prior to IOC because penetration testing could disrupt operations and could identify flaws that would prevent FOC. But experience has shown there is usually not enough time to do an adequate job. For that reason, this task is shown as part of the residual testing due to the allotted time given

for testing and updates to the AIS between IOC and FOC. (NOTE: A mockup of the system to test against would aid the AIS ST&E and the penetration testing)

The certifier must first conduct a thorough review of the penetration test plan, which must identify the approach to be used in attempting to penetrate the security mechanisms of the system. Penetration tests must be performed by government personnel, or by an independent verifier appointed by the government. The object of the penetration tests is to discover design and implementation flaws that could allow for compromise of classified data. Due to the "no holds barred" nature of the test, the plan can only indicate a general methodology and initial steps to be used. The plan must include, however, a thorough description of the ground rules to be used, potential implications to the test environment (if the test will be done in conjunction with other operational, development, or test activities), potential impacts to other systems (if the testing configuration will have any kind of external connectivity), and the method for protecting results of successful penetrations.

The penetration tests efforts shall use the flaw hypothesis or equivalent security testing methodology. They should include a thorough analysis of the design, documentation, source code, and object code relevant to security protection, as well as "hands on" attempts to disable or circumvent the security protection features. Any found vulnerability should be properly documented including the extent of the penetration, the circumstances in which it took place (either in a fixed or dynamic situation), and the potential vulnerabilities of repeating or expanding the attempt.

The penetration tests report should also propose countermeasures for each successful penetration, and include the potential impact of each of the recommended countermeasures.

CONTINGENCY PLANNING:

No AIS is exempt from potential failure. Therefore, Computer Systems Managers (CSMs) and Network Managers (NMs) with functional Office of Program

Responsibility's (OPR) coordination, must develop plans to ensure the survival and timely recovery of the systems for which they are responsible.

Contingency and recovery plans must describe the actions necessary to ensure continuity of operations in the event of a disaster, or to restore operations in the event of a system failure. The scope and contents of these plans will vary depending on the criticality of the system. However, all contingency plans must outline actions required to prepare for emergencies, interim processing requirements, software and data backup strategy, and equipment maintenance concept. The CSM, NM, or functional OPR must ensure that contingency plans are reviewed annually and updated as necessary.

Life Cycle Report:

The purpose of the life cycle report is to determine the measures which are needed to be put in place to assure the continued life cycle certification of the AIS. This report needs to address many items such as how software and hardware changes are certified to maintain the AIS's certification and how reaccreditations are handled.

FINAL CERTIFICATION PACKAGE:

Final certification package requirements are basically the same as those for the draft certification package, but updated with all testing finalized and documentation completed. The needed documentation for FOC should be listed in the DAA accreditation plan.

FINAL OPERATIONAL CAPABILITY (FOC) REQUEST LETTER:

The FOC request letter is based on AFR 205-16, Figure A9-3. The DAA accreditation package includes certification letters and checklists. It is from the reviewing authority to the DAA with a recommendation to approve or disapprove system processing, which risks to assume, and proposed dates to correct deficiencies. Extracts from supporting documents provide the information the DAA needs to make the decision.

CONCLUSIONS:

It should be stressed that the responsible personnel factor mentioned earlier in this paper is the largest factor which will determine not only the thoroughness but also the domain of testing that may be performed. This test domain is a reflection of the negotiated agreement between the wants (assurances) and needs (directives and regulations) of the "DAA" and the capabilities and commitment of the "test team". It is obvious that the test team must be a strong one. The team must have the ability to analyze the combined securities as an environment, for it is both the physical and the electronic security elements which together makeup the environment which is to be trusted.

This C&A approach provides AIS programs with a methodology as well as an understanding of the C&A activities throughout the AIS's life cycle.

List of References

- Chiles, C. M., et al., May 1988, *Computer Security Certifications of Air Force Applications*, WP-27827, The MITRE Corporation, Bedford, MA.
- Defense Communications Agency (DCA), 1988a, *Model Command Center Security Architecture (Draft)*, DCA Technical Report TR-xx, Defense Communications Agency, Washington, DC.
- _____, 1988b, *Security Concept of Operations for the Military Airlift Command's Command and Control System (Draft)*, DCA Technical Report TR-xx, Defense Communications Agency, Washington, DC.
- _____, 1988c, *Security Policy for the Military Airlift Command's Command and Control System (Draft)*, DCA Technical Report TR-xx, Defense Communications Agency, Washington, DC.
- _____, 1989, *DOD C2MLS Prototype Program Management Plan*, Draft, DCA Technical Report TR-xx, Defense Communications Agency, Washington, DC.
- Department of Defense (DOD), December 1985, *Department of Defense Trusted Computer System Evaluation Criteria*, DOD Directive 5200.28-STD, Department of Defense, Washington, DC.
- _____, 29 February 1988a, *Military Standard - Defense System Software Development*, DOD-STD-2167A, Department of Defense, Washington, DC.
- _____, 22 March 1988b, *Security Requirements for Automated Information Systems (A/ISs)*, DOD Directive 5200.28, Department of Defense, Washington, DC.
- Kannel, Muriel, et al., 1992, *Security Certification and Accreditation Plan (SCAP) tool*, Air Force Contract F19628-89-C-0001, McLean, VA., The MITRE Corporation.
- Military Airlift Command (MAC), 12 January 1989, *MAC Supplement 1 to AFR 205-16*, Military Airlift Command, Scott AFB, IL.
- _____, August 1990, *MAC GDSS MLS Prototype Security Classification Guide*, Military Airlift Command, Scott AFB, IL.
- _____, January 1992, *Security Policy for the MLS GDSS*, Military Airlift Command, Scott AFB, IL.
- National Bureau of Standards (NBS), 1983, *Guideline for Computer Security Certification and Accreditation*, Federal Information Processing Standards Publication (FIPS PUB) 102, National Bureau of Standards, Washington, DC.
- Neugent, H. William, et al., 1987, *WIS Security Certification and Accreditation Plan*, MTR-86W00038, McLean, VA., The MITRE Corporation.
- National Security Agency (NSA), October 1991, *Information Systems Security Products and Services Catalogue*, National Security Agency, Washington, DC.
- United States Air Force (USAF), 1985, *Guidance for Performing a Risk Analysis*, ADPSEC Guideline #4, Air Force Computer Security Program Office (AFCSPPO), Department of the Air Force, Washington, DC.
- _____, 1 October 1987, *S1100 System Security and Classified Processing: P104K/JZ*, Air Force Manual 171-110, Volume V, Department of the Air Force, Washington, DC.
- _____, 28 April 1989, *Computer Security Policy*, AFR 205-16, Department of the Air Force, Washington, DC.

**SOCIAL PSYCHOLOGY AND INFOSEC:
PSYCHO-SOCIAL FACTORS IN THE IMPLEMENTATION OF INFORMATION SECURITY POLICY**

M. E. Kabay, Ph.D.

Director of Education, National Computer Security Association
Carlisle, PA

President, JINBU Corporation
P. O. Box 509 / Westmount, QC
H3Z 2T6 Canada
Internet: 75300.3232@compuserve.com

INTRODUCTION

Security policies and procedures affect not only what people do but also how they see themselves, their colleagues and their world. Despite these psychosocial issues, security personnel pay little or no attention to what is known about social psychology. The established principles of human social behaviour have much to teach us in our attempts to improve corporate and institutional information security.

Information security specialists concur that security depends on people more than on technology. Another commonplace is that employees are a far greater threat to information security than outsiders.

It follows from these observations that improving security depends on changing beliefs, attitudes and behaviour, both of individuals and of groups. Social psychology can help us understand how best to work with human predilections and predispositions to achieve our goals of improving security:

- o research on social cognition looks at how people form impressions about reality (knowing these principles, we can better teach our colleagues and clients about effective security);
- o work on attitude formation and beliefs helps us present information effectively and so convince employees and others to cooperate in improving security;
- o scientists studying persuasion and attitude change have learned how best to change people's minds about unpopular views such as those of the security community;
- o studies of factors enhancing prosocial behaviour provide insights on how to foster an environment where corporate information is willingly protected;
- o knowledge of the phenomena underlying conformity, compliance and obedience can help us enhance security by encouraging compliance and by protecting staff against social pressure to breach security;
- o group psychology research provides warnings about group pathology and hints for working better with groups in establishing and maintaining information security in the face of ingrained resistance.

The following discussion is based on well-established principles of social psychology. Any recent introductory college textbook in this field will provide references to the research that has led to the principles which are applied to security policy implementation. In this paper, references are to Lippa, R A (1990). *Introduction to Social Psychology*. Wadsworth (Belmont, CA). ISBN 0-534-11772-4.

SOCIAL COGNITION

Schemas are self-consistent views of reality. They help us pay attention to what we expect to be important and to ignore irrelevant data. They also help us organize our behaviour [Lippa, p. 141]. For example, our schema for relations at the office includes polite greetings, civil discussions, written communications, and businesslike clothes. The schema excludes obscene shrieks, abusive verbal attacks, spray-painted graffiti and colleagues dressed in swim suits. It is the schema that lets people tell what is inappropriate in a given situation.

Security policies and procedures conflict with most people's schema. Office workers' schema includes sharing office supplies ('Lend me your stapler, please?'), trusting your team members to share information ('Take a look at these figures, Sally'), and letting your papers stay openly visible when you have to leave your desk. Unfortunately, sharing user IDs, showing sensitive information to someone who lacks the appropriate clearance, and leaving work stations logged on without protection are gross breaches of a different schema. Normal politeness dictates that when a colleague approaches the door we have just opened, we hold the door open for them; when we see a visitor, we smile politely (who knows, it may be a customer). In contrast, access policies require that we refuse to let even a well-liked colleague piggy-back their way through an access-card system; security policies insist that unbadged strangers be challenged or reported to security personnel. Common sense tells us that when the Chief Executive Officer of the company wants something, we do it; yet we try to train computer room operators to forbid entry to anyone without documented authorization--including the CEO.

Schemas influence what we perceive [Lippa, p. 143]. For example, an employee refuses to take vacations, works late every night, is never late, and is never sick. A model employee? Perhaps, from one point of view. From the security point of view, the employee's behaviour is suspect. There have been cases where such people have actually been embezzlers unable to leave their employment: even a day away might result in discovery. Saint or sinner? Our expectations determine what we see.

Schemas influence what we remember [Lippa, p. 145]. When information inconsistent with our preconceptions is mixed with details that fit our existing schemas, we selectively retain what fits and discard what conflicts. When we have been fed a diet of movies and television shows illustrating the premise that information is most at risk from brilliant hackers, why should we remember the truth--that carelessness and incompetence by authorized users of information systems cause far more harm than evil intentions and outsiders ever do.

Before attempting to implement policies and procedures, we should ensure that we build up a consistent view of information security among our colleagues. In light of the complexity of social cognition, our usual attempts to implement security policies and procedures seem pathetically inept. A couple of hours of lectures followed by a video, a yearly ritual of signing a security policy that seems to have been written by Martians--these are not methods that will improve security. These are merely lip service to the idea of security.

According to research on counter-intuitive information, people's judgement is influenced by the manner in which information is presented. For example, even information contrary to established schemas can be assimilated if people have enough time to integrate the new knowledge into their world-views [Lippa, p. 148]. It follows that security policies should be introduced over a long time, not rushed into place.

Preliminary information may influence people's responses to information presented later. For example, merely exposing experimental subjects to a list of words such as 'reckless' or 'adventurous' affects their judgement of risk-taking behaviour in a later test. It follows that when preparing to increase employee awareness of security issues, presenting case-studies is likely to have a beneficial effect on participants' readiness to examine security requirements.

Pre-existing schemas can be challenged by several counter-examples, each of which challenges a component of the schema [Lippa, p. 153]. For example, prejudice about an ethnic group is more likely to be changed by contact with several people, each of whom contradicts a different aspect of the prejudiced schema. It follows that security awareness programs should include many realistic examples of security requirements and breaches. Students in the NCSA's *Information Systems Security Course* have commented on the unrealistic scenario in a training video they are shown; a series of disastrous security breaches occur in the same company. Based on the findings of cognitive social psychologists, the film would be more effective for training if the incidents were dramatized as occurring in different companies.

Judgements are easily distorted by the tendency to rely on personal anecdotes, small samples, easily available information, and faulty interpretation of statistical information [Lippa, p. 155-163]. Basically, we humans are not rational processors of factual information. If security awareness programs rely strictly on presentation of factual information about risks and proposed policies and procedures, they will run up against our stubborn refusal to act logically. Security program implementation must engage more than the rational mind. We must appeal to our colleagues' imagination and emotion as well. We must inspire a commitment to security rather than merely describing it.

Perceptions of risks and benefits are profoundly influenced by the wording in which situations and options are presented [Lippa, p. 163]. For example, experimental subjects responded far more positively to reports of a drug with '50% success' than to the same drug described as having '50% failure.' It follows that practitioners should choose their language carefully during security awareness campaigns. Instead of focusing on reducing failure rates (breaches of security), we should emphasize improvements of our success rate.

BELIEFS AND ATTITUDES

Psychologists distinguish between beliefs and attitudes. 'A *belief*... refers to cognitive information that need not have an emotional component....' An attitude refers to 'an evaluation or emotional response....' [Lippa, p. 238]. Thus a person may *believe* that copying software without authorization is a felony while nonetheless having the attitude that it doesn't matter.

Beliefs can change when contradictory information is presented, but some research suggests that it can take up to a week before significant shifts are measurable. Other studies suggest that when people hold contradictory beliefs, providing an opportunity to articulate and evaluate those beliefs may lead to changes that reduce inconsistency.

These findings imply that a new concern for corporate security must be created by exploring the current structure of beliefs among employees and managers. Questionnaires, focus groups, and interviews may not only help the security practitioner, they may actually help move the corporate culture in the right direction.

An attitude, in the classical definition, 'is a learned evaluative response, directed at specific objects, which is relatively enduring and influences behaviour in a generally motivating way' [Lippa, p. 221]. The advertising industry spends over \$50B yearly to influence public attitudes in the hope that these attitudes will lead to changes in spending habits--that is, in behaviour.

Research on classical conditioning suggests that attitudes can be learned even because of simple word association [Lippa, p. 232]. If we wish to move our colleagues towards a more negative view of computer criminals, it is important not to portray computer crime using positive images and words. Movies like *Sneakers* may do harm indirectly by associating pleasant, likeable people with techniques that are used for industrial espionage. When teaching security courses, we should avoid praising the criminals we describe in case studies.

One theory on how attitudes are learned suggests that rewards and punishments are important motivators. Studies show that even apparently minor encouragement can influence attitudes. A supervisor or instructor should praise any comments that are critical of computer crime or which support the established security policies. Employees who dismiss security concerns or flout the regulations should be challenged on their attitudes, not ignored.

PERSUASION AND ATTITUDE CHANGE

Persuasion--changing someone's attitudes--has been described in a terms of communications [Lippa, p. 258]. The four areas of research include

- o communicator variables: who is trying to persuade?
- o message variables: what is being presented?
- o channel variables: by what means is the attempt taking place?
- o audience variables: at whom is the persuasion aimed?

Attractiveness, credibility and social status have strong effects immediately after the speaker or writer has communicated with the target audience; however, over a period of weeks to a month, the effects decline until the predominant issue is message content. We can use this phenomenon by identifying the senior executives most likely to succeed in setting a positive tone for subsequent security training. We should look for respected, likeable people who understand the issues and sincerely believe in the policies they are advocating.

Fear can work to change attitudes only if judiciously applied. Excessive emphasis on the terrible results of poor security is likely to backfire, with participants in the awareness program rejecting the message altogether. Frightening consequences should be coupled immediately with effective and *achievable* security measures.

Some studies suggest that presenting a balanced argument helps convince those who initially *disagree* with a proposal. Presenting objections to a proposal and offering counter-arguments is more effective than one-sided diatribes. The Software Publishers' Association training video, *It's Just Not Worth the Risk*, uses

this technique: it shows several members of a company arguing over copyright infringement and fairly presents the arguments of software thieves before demolishing them.

Modest repetition of a message can help generate a more positive response. Thus security awareness programs which include imaginative posters, mugs, special newsletters, audio and video tapes and lectures are more likely to build and sustain support for security than occasional intense sessions of indoctrination.

The channel through which we communicate has a strong effect on attitudes and on the importance of superficial attributes of the communicator. 'Face-to-face persuasion often proves to have more impact than persuasion through the mass media.... [because they] are more salient, personal and attention-grabbing, and thus they often stimulate more thought and commitment to their persuasive messages' [Lippa, p. 264]. Security training should include more than tapes and books; a charismatic teacher or leader can help generate enthusiasm for--or at least reduce resistance to--better security.

Workers testing cognitive response theory [Lippa, p. 289] have studied many subtle aspects of persuasion. For example, experiments have shown that rhetorical questions (e.g., 'Are we to accept invasions of our computer systems?') are effective when the arguments are solid but counter-productive when arguments are weak.

In comparing the central route to persuasion (i.e., consideration of facts and logical arguments) with the peripheral (i.e., influences from logically unrelated factors such as physical attractiveness of a speaker), researchers find that the central route 'leads to more lasting attitudes and attitude changes....' [Lippa, p. 293].

As mentioned above, questionnaires and interviews may help cement a favourable change in attitude by leading to commitment. Once employees have publicly avowed support for better security, some will begin to change their perception of themselves. As a teacher of information security, I find that I now feel much more strongly about computer crime and security than I did before I created my courses. We should encourage specific employees to take on public responsibility for information security within their work group. This role should periodically be rotated among the employees to give everyone the experience of public commitment to improved security.

PROSOCIAL BEHAVIOUR

Studies of how and why people help other people have lessons for us as we work to encourage everyone in our organizations to do the right thing. Why do some people intervene to stop crimes? Why do others ignore crimes or watch passively? Latane and Darley (Lippa, p. 493) have devised a schema that describes the steps leading to prosocial behaviour:

- o People have to notice the emergency or the crime before they can act. Thus security training has to include information on how to tell that someone may be engaging in computer crime.
- o The situation has to be defined as an emergency--something requiring action. Security training that provides facts about the effects of computer crime on society and solid information about the need for security within the organization can help employees recognize security violations as emergencies.

- o We must take responsibility for acting. The *bystander effect* comes into play at this stage. The larger the number of people in a group confronted with an emergency, the slower the average response time. Larger groups seem to lead 'to a diffusion of responsibility whereby each person felt less personally responsible for dealing with the emergency' [Lippa, p. 497]. Another possible factor is uncertainty about the social climate; people fear 'appearing foolish or overly emotional in the eyes of those present.' We can address this component of the process by providing a corporate culture which rewards responsible behaviour such as reporting security violations.
- o Having taken responsibility for solving a problem, we must decide on action. Clearly written security policies and procedures will make it more likely that employees act to improve security. In contrast, contradictory policies, poorly-documented procedures, and inconsistent support from management will interfere with the decision to act.

Another analysis proposes that people implicitly analyze costs of helping and of not helping when deciding whether to act prosocially. The combination of factors most conducive to prosociality is low cost for helping and high cost for not helping. Security procedures should make it easy to act in accordance with security policy; e.g., there should be a hot-line for reporting security violations, anonymity should be respected if desired, and psychological counselling and followup should be available if people feel upset about their involvement. Conversely, failing to act responsibly should be a serious matter; personnel policies should document clear and meaningful sanctions for failing to act when a security violation is observed; e.g., inclusion of critical remarks in employment reviews and even dismissal.

One method that does *not* work to increase prosocial behaviour is exhortation [Lippa, p. 513]. That is, merely lecturing people has little or no effect. On the other hand, the general level of stress and pressure to focus on narrow tasks can significantly reduce the likelihood that people will act on their moral and ethical principles. Security is likely to flourish in an environment that provides sufficient time and support for employees to work professionally; offices where everyone responds to self-defined emergencies all the time will not likely pay attention to security violations.

Some findings from research confirm common sense. For example, guilt motivates people to act more prosocially. This effect works best 'when people are forced to assume responsibility....' Thus enforcing standards of security using reprimands and sanctions can indeed increase the likelihood that employees will subsequently act more cooperatively. In addition, mood affects susceptibility to prosocial pressures: bad moods make prosocial behaviour less likely, whereas good moods increase prosociality. A working environment in which employees are respected is more conducive to good security than one which devalues and abuses them. Even cursory acquaintance with other people makes it more likely that we will help them; it thus makes sense for security supervisors to get to know the staff from whom they need support. Encouraging social activities in an office (lunch groups, occasional parties, charitable projects) enhances interpersonal relationships and can improve the climate for effective security training.

CONFORMITY, COMPLIANCE AND OBEDIENCE

Turning a group into a community provides a framework in which social pressures can operate to improve our organization's information security. People respond to the opinions of others by (sometimes unconsciously) shifting their opinion towards the mode. Security programs must aim to shift the normative values (the sense of what one *should* do) towards confidentiality, integrity and availability of data. As we

have seen in public campaigns aimed at reducing drunk driving, it is possible to shift the mode. Twenty years ago, many people believed that driving while intoxicated was amusing; today a drunk driver is a social pariah. We must move towards making computer crime as distasteful as public drunkenness.

The trend towards conformity increases when people within the group like or admire each other [Lippa, p. 534]. In addition, the social status of an individual within a group influences that individual's willingness to conform. High-status people (those liked by most people in the group) and low-status people (those disliked by the group) both tend to more autonomous and less compliant than people liked by some and disliked by others [Lippa, p. 536]. Therefore the security officers should pay special attention to those outliers during instruction programs. Managers should monitor compliance more closely in both ends of the popularity range. Contrariwise, if security practises are currently poor and we want allies in changing the norm, we should work with the outliers to resist the herd's anti-security bias.

'The norm of reciprocity holds that we should return favours in social relations' [Lippa, p. 546]. Even a small, unexpected or unsolicited (and even unwanted) present increases the likelihood that we will respond to requests. A security awareness program that includes small gifts such as a mug labelled 'SECURITY IS EVERYONE'S BUSINESS' or an inexpensive booklet such as the *Information Systems Security Pocket Guide* (available from the NCSA) can help get people involved in security.

The 'foot in the door' technique suggests that we 'follow a small initial request with a much larger second request' [Lippa, p. 549]. For example, we can personally ask an employee to set a good example by blanking their screen and locking their terminal when they leave their desk. Later, once they have begun their process of redefinition of themselves ('I am a person who cares about computer security'), we can ask them for something more intense, such as participating in security training for others (e.g., asking each colleague to blank their screen and lock their terminal).

GROUP BEHAVIOUR

Early studies on the effects of being in groups produced contradictory behaviour; sometimes people did better at their tasks when there were other people around and sometimes they did worse. Eventually, social psychologist Robert Zajonc [Lippa, p. 572 ff.] realized that 'The presence of others is arousing, and this arousal facilitates dominant, well-learned habits but inhibits nondominant, poorly-learned habits.' Thus when trying to teach employees new habits, it is counter-productive to put them into large groups. Individualized learning (e.g., computer-based training, video tapes) can overcome the inhibitory effect of groups in the early stages of behavioural change.

Another branch of research in group psychology deals with *group polarization*. Groups tend to take more extreme decisions than individuals in the group would have [Lippa, p. 584]. In group discussions of the need for security, polarization can involve deciding to take more risks--by reducing or ignoring security concerns--than any individual would have judged reasonable. Again, one-on-one discussions of the need for security may be a more effective approach to building a consensus that supports cost-effective security provisions than large meetings.

In the extreme, a group can display *groupthink*, in which a consensus is reached because of strong desires for social cohesion [Lippa, p. 586 ff.]. When groupthink prevails, evidence contrary to the received view is discounted; opposition is viewed as disloyal; dissenters are discredited. Especially worrisome for security professionals, people in the grip of groupthink tend to ignore risks and

contingencies. To prevent such aberrations, the leader must remain impartial and encourage open debate. Experts from the outside (e.g., respected security consultants) should be invited to address the group, bringing their own experience to bear on the group's requirements. After a consensus has been achieved, the group should meet again and focus on playing devil's advocate to try to come up with additional challenges and alternatives.

CONCLUSIONS

By viewing information security as primarily a management issue, we can benefit from the mass of knowledge accumulated by social psychologists. We can implement security policies and procedures more easily by adapting our training and awareness techniques to correspond to human patterns of learning and compliance.

SUMMARY OF RECOMMENDATIONS

1. Before attempting to implement policies and procedures, we should ensure that we build up a consistent view of information security among our colleagues.
2. Security policies should be introduced over a long time, not rushed into place.
3. Presenting case-studies is likely to have a beneficial effect on participants' readiness to examine security requirements.
4. Security awareness programs should include many realistic examples of security requirements and breaches.
5. We must inspire a commitment to security rather than merely describing it.
6. Emphasize improvements rather than reduction of failure.
7. A new concern for corporate security must be created by exploring the current structure of beliefs among employees and managers.
8. Do not to portray computer crime using positive images and words.
9. Praise any comments that are critical of computer crime or which support the established security policies.
10. Employees who dismiss security concerns or flout the regulations should be challenged on their attitudes, not ignored.
11. Identify the senior executives most likely to succeed in setting a positive tone for subsequent security training.
12. Frightening consequences should be coupled immediately with effective and *achievable* security measures.
13. Presenting objections to a proposal and offering counter-arguments is more effective than one-sided diatribes.
14. Security awareness programs should include repeated novel reminders of security issues.

15. In addition to tapes and books, rely on a charismatic teacher or leader to help generate enthusiasm for better security.
16. Encourage specific employees to take on public responsibility for information security within their work group.
17. Rotate the security role periodically.
18. Security training should include information on how to tell that someone may be engaging in computer crime.
19. Build a corporate culture which rewards responsible behaviour such as reporting security violations.
20. Develop clearly written security policies and procedures.
21. Security procedures should make it easy to act in accordance with security policy.
22. Failing to act in accordance with security policies and procedures should be a serious matter.
23. Enforcing standards of security can increase the likelihood that employees will subsequently act more cooperatively.
24. A working environment in which employees are respected is more conducive to good security than one which devalues and abuses them.
25. Security supervisors should get to know the staff from whom they need support.
26. Encourage social activities in the office.
27. Pay special attention to social outliers during instruction programs.
28. Monitor compliance more closely in both ends of the popularity range.
29. Work with the outliers to resist the herd's anti-security bias.
30. Include small gifts in your security awareness program.
31. Start improving security a little at a time and work up to more intrusive procedures.
32. Before discussing security at a meeting, have one-on-one discussions with the participants.
33. Remain impartial and encourage open debate in security meetings.
34. Bring in experts from the outside when faced with groupthink.
35. Meet again after a consensus has been build and play devil's advocate.

TRUSTED SYSTEMS: APPLYING THE THEORY IN A COMMERCIAL FIRM

Ernest C. Charles
Donna A. Diodati
Walter J. Mozdierz

Information Technology Security & Business Continuity
Aetna Life & Casualty
Mail Stop C14N
151 Farmington Ave.
Hartford, Connecticut 06156

ABSTRACT

Aetna's information technology resources include mainframe systems running in several computer centers; mid-range processors; local area networks; and personal computer workstations. Most resources located outside the computer centers are connected to at least one mainframe environment. The company's security mechanisms and procedures are plentiful and diverse.

In reviewing the information technology security issues which most organizations face, Aetna's Information Technology Security and Business Continuity department found the following to be most significant:

- One enterprise-wide information technology security policy endorsed by senior management ensures consistency in procedures, standards, practices, training, and audits.
- The business manager is accountable for protecting corporate information. Under this principle, business managers decide who can access the information for which they are responsible, and what the people granted access can do with the data.
- The trusted computing base (TCB)--hardware, software, and procedures--is defined so that security and audit professionals should only be concerned with *changes* to the TCB. If the TCB is not defined, it is subject to interpretation whenever a system is audited.
- The features and procedures necessary to protect the TCB must be clearly defined, implemented, and maintained. When security maintenance procedures are inconsistent and lack a foundation in trusted systems, the procedures for *maintaining the environment* are inconsistent and incomplete.
- Inconsistent, "home-grown" security features embedded in the applications inhibit standardization. It is better to use the host operating system, or a trusted systems rated security package, where practical.
- Published baseline standards against which to audit security compliance are critical.

- If monitoring and investigations of security-relevant events are inconsistent, it is unknown how much security a system has, or how well it works.

To address these factors, Aetna's information technology security group proposed a trusted systems framework that provides for business manager accountability for data; measurements of security effectiveness; and standards for auditing. The Aetna Trust and Assurance Requirements (ATAR) were created to provide minimum standards for security features and functions. Business managers evaluate existing systems against the ATAR, while new systems are designed with ATAR in mind. Experts in applications development and information technology security may guide business managers through this process, but the decisions are the business managers' alone.

Through a careful, integrated implementation of the trusted systems framework, Aetna is providing appropriate access to information, anytime, anywhere via systems with an assured level of trust. This paper describes how Aetna is translating the theory of trusted systems into practice.

INTRODUCTION

Many people view information technology security as expensive, cumbersome, and difficult to administer. This reputation is justified: until recently, there were few standards, with many mechanisms available to enforce them. Without the information technology security equivalent of generally accepted accounting principles, there are inconsistencies among information technology security practices, and subjective audits of their effectiveness.

Several national and international groups have addressed these issues in the last 20 years. The 1970s saw many "standards" emerging, while the 1980s brought clearer standards and metrics. Under such volatile conditions, early commercial implementations of trusted systems met with mixed results. The reasons for this included:

- The U.S. military's emphasis on access control and data classification over auditability and other features which are important to the commercial sector;
- The National Security Agency's difficulty in marketing the concept effectively to commercial entities; and
- The National Security Agency's assumption that all government systems needed A-1 (highest level) features, and that the commercial sector would, too.

Widely accepted international standards for information technology security have emerged in the last few years. The U.S. government's Trusted Computer System Evaluation Criteria (The "Orange Book") has been a key part of the emerging international standards. With cost containment vital to their survival in the 1990s, companies such as Aetna are adopting this framework to ensure their competitiveness in the global marketplace. Although many commercial firms have implemented trusted systems, we believe that Aetna is the first insurance company to do so.

AETNA'S BUSINESS ENVIRONMENT

Founded in 1853, Aetna is a leading provider of insurance and financial services to corporations, public and private institutions, and individuals. With assets of \$89.9 billion, Aetna ranks as the nation's largest stockholder-owned insurance and financial services organization, and is the 14th largest U.S. company.

On the basis of written premiums, Aetna is one of the nation's largest insurance company providers of group health and life benefits; one of the largest underwriters of commercial property-casualty coverage; and the sixth largest underwriter of personal property-casualty products. Measured by assets under management, Aetna is among the ten largest managers of pension assets in the United States.

Aetna's operating components are divided into 18 "strategic business units" that are supported by several "strategic service units," including Aetna Information Technology (AIT). AIT includes the Information Technology Security and Business Continuity department, which sets corporate policy and standards in its arena.

Approximately 15,000 of Aetna's 43,000 U.S. employees work at the home office in Hartford, Connecticut, with the remaining 28,000 employees based in 600 domestic field offices. International employees are based in Bermuda, Canada, China, Hong Kong, Korea, Malaysia, Mexico, Taiwan, and the United Kingdom, among other places. Most employees use computers in their work, and the number of trading partners using Aetna's technology is growing yearly.

AETNA'S TECHNOLOGY ENVIRONMENT

Aetna's information technology resources include mainframe systems running in several computer centers; mid-range processors; local area networks; and personal computer workstations. Most resources located outside the computer centers are connected to at least one mainframe environment.

The company's information technology security architecture has a mainframe orientation, with a commercially available product as the foundation. This product has been certified as meeting the "Orange Book" C2 level of trust, when properly implemented in an operating environment such as Aetna's. Nonetheless, Aetna's information technology security policies, procedures, and mechanisms are plentiful and diverse.

In reviewing the information technology security issues which most organizations face, Aetna's Information Technology Security and Business Continuity department found the following to be the most significant:

- One enterprise-wide information technology security policy endorsed by senior management ensures consistency in procedures, standards, practices, training, and audits.

- The business manager is accountable for protecting corporate information. Under this principle, business managers decide who can access the information for which they are responsible, and what the people granted access can do with the data.
- The trusted computing base (TCB)--hardware, software, and procedures--is defined so that security and audit professionals should only be concerned with *changes* to the TCB. If the TCB is not defined, it is subject to interpretation whenever a system is audited.
- The features and procedures necessary to protect the TCB must be clearly defined, implemented, and maintained. When security maintenance procedures are inconsistent and lack a foundation in trusted systems, the procedures for *maintaining the environment* are inconsistent and incomplete.
- Inconsistent, "home-grown" security features embedded in the applications inhibit standardization. It is better to use the host operating system, or a security package rated as a trusted system, where practical.
- Published baseline standards against which to audit security compliance are critical.
- If monitoring and investigations of security-relevant events are inconsistent, it is unknown how much security a system has, or how well it works.

If standardization and strategic planning are missing, these problems arise:

- Information technology security practices become more fragmented;
- The existing issues grow more significant as the number of people with systems access increases; and
- The information on which an organization's business relies could be exposed to unknown and potentially unacceptable risks.

Figure 1 summarizes the differences between environments with and without the trusted systems framework.

AETNA'S TRUSTED ENVIRONMENT

In early 1992 Aetna's several information technology security units were centralized. The new department head quickly conducted customer satisfaction surveys, and met with top management of the business units. By September the department had reviewed the current security environment, and recommended the trusted systems framework to close existing gaps.

A standards document called the Aetna Trust and Assurance Requirements (ATAR) was the first tangible product of this effort. The ATAR represents the minimum security features for Aetna's business systems. It is a composite of the C2 class requirements of the Trusted Systems Evaluation Criteria (Orange Book); the F2 standard from the International Standards

Organization (ISO) (Green Book); and others. This blend was used to ensure that Aetna's trusted systems framework would meet international standards.

The ATAR specifies consistent security features and functions, which represent Aetna's trusted systems baseline. Existing systems will be evaluated against the ATAR, while new systems are designed with ATAR in mind. Business managers conduct these evaluations using self-assessment tools, and then decide whether to comply with ATAR. Systems that comply with ATAR will have the following control features:

- Identification and authentication of everyone who accesses an Aetna information resource;
- Controlled access to data;
- Accountability for the exchange of information;
- Auditing and monitoring of security-relevant events;
- Assurance reviews of the product's trust level;
- Risk assessment with corresponding disaster backup and recovery plans; and
- Controlled and uncorrupted electronic data exchange.

Aetna's main objectives in using the ATAR are to ensure:

- A well-defined, comprehensive, effective and efficient information technology security architecture spanning all strategic platforms, software systems, and application environments;
- Rules for a consistent user interface with a company-wide accessor standard ("user ID") that is independent of the hardware or software components involved;
- A security system that accepts and enforces company policy decisions and protection requirements, with mechanisms provided for automatic detection and reporting of deviations and integrity exposures;
- An information technology security framework which is flexible enough to satisfy protection requirements ranging from minimum standard (baseline) through extremely high levels; and
- Standard options and default values that allow for security implementation in simple migration steps.

The ATAR addresses general purpose multi-user business systems, with the expectation that they will be used in a wide variety of applications. These requirements apply to multi-user workstations, minicomputers, and mainframes. They do not address any security requirements that are specific to a particular type of computer system. Workstation-specific requirements, for example, are not addressed.

The ATAR is not meant to recommend or endorse any particular vendor's products; impose further restrictions on the development and delivery of information systems; or comprise a complete set of security design specifications. The ATAR assumes the existence of a routine, well-managed operational environment maintained through assurance reviews and independent audits.

Since attackers might be able to gain nominal access to a system, the business manager must determine the need for, and apply, appropriate types of controls in the environment over such nominal access. The ATAR is a "reasonable first-line defense." However, in high-payoff circumstances, highly motivated persons may exert the effort to circumvent it. Under such circumstances, a system designed to meet the ATAR would be inappropriate. A system that has been designed and developed in total compliance with the ATAR can and will contain vulnerabilities to higher levels of attack.

IMPLEMENTATION

Through a careful, integrated implementation of the trusted systems framework, Aetna is providing appropriate access to information, anytime, anywhere via systems with an assured level of trust. The trusted systems framework is an integral part of Aetna's information technology architecture. Its implementation requires the Information Technology Security and Business Continuity group to:

- Gain Aetna management commitment to the trusted systems approach;
- Provide enterprise-wide information systems security and business continuity standards;
- Deliver education and awareness and self-assessment materials and presentations for management, employees, and other interested parties;
- Consult on all phases of information systems security and business continuity planning, implementation, and ongoing assessments;
- Provide on-site support in the Home Office, field, or international locations, where appropriate;
- Administer mainframe security software systems;
- Coordinate periodic tests of the computer centers' disaster recovery plans; and
- Ensure that senior management in each strategic business unit or service unit:
 - Examines its requirements for information systems security and business continuity;
 - Assesses their business functions and information resources to determine which are critical or sensitive;
 - Maintains appropriate information systems security controls and business continuity plans;
 - Assures compliance with applicable Aetna policies and federal, state, and local laws and regulations concerning information integrity, confidentiality, and availability;

- Designs new systems to comply with the ATAR; and
- Evaluates regularly the risks to which their businesses are exposed, and the adequacy of protective measures to minimize these risks.

If an existing business system does not have the security features and functions that the ATAR specifies, the business manager will document that:

- The system does not need the control due to the asset's value; or
- The control does not apply to this environment; or
- The system has cost-effective compensating or equivalent controls; or
- The business manager has considered the control and decided to accept the risk of non-compliance for now.

Rather than act as enforcers, Information Technology Security and Business Continuity consultants have collaborated with other departments that have a stake in Aetna's information technology security. These include Internal Audit, Law, business managers, and technology applications development units.

One such effort launched a company-wide information technology security and business continuity policy. This policy was introduced in conjunction with a corporate initiative aimed at streamlining company policy development and implementation. This ensured the policy's endorsement by senior management, and consistent implementation throughout the company.

The security group also obtained the Internal Audit department's approval of the ATAR, and their agreement to audit systems in light of the business manager's risk assessment decisions. There would be no more "surprise" audit comments, if management maintained the agreed-upon security level, including administrative controls.

Another collaboration involved the Corporate Controllers' internal controls documentation program. This initiative requires Aetna business managers to show that they continually assess the risks to which their operations are exposed, and that they use appropriate controls to mitigate them. The information technology security self-assessment packages, when completed by a business manager, partially fulfill these internal controls documentation requirements.

Such cooperation has been key to successful implementation of trusted systems at Aetna. Senior management, auditors, security professionals, business managers, and others have cooperated to ensure a world-class information technology security environment.

REFERENCES

1. *Information Technology Security Evaluation Criteria (ITSEC)*, Harmonised Criteria of France, Germany, The Netherlands, United Kingdom, Version 1.2, June 1991, published by the Commission of the European Communities ("Green Book").
2. *Trusted Computer System Evaluation Criteria*. Department of Defense Standard 5200.28-STD, 1985 ("Orange Book").

INFORMATION TECHNOLOGY SECURITY ENVIRONMENT

BUSINESS NEED	WITHOUT TRUSTED SYSTEMS	WITH TRUSTED SYSTEMS
Business Manager's Accountability for Information Security	Not well understood by most business managers or technology managers.	Business managers evaluate systems against standards (ATAR); assess the costs and benefits of any exposures; and decide which security features and controls are appropriate.
Security Measures & Metrics	Do not exist in any meaningful way.	Information technology security department monitors clearly defined levels of trust through the proper selection and implementation of trusted products.
Minimum Security Baseline	Are incomplete, and inconsistent where applied.	Information technology security department defines the standards. Application developers implement and maintain the standards at business manager's request.
Auditing	Are subject to interpretation, with no indication of risk.	Business managers routinely audit for selected security-relevant events. Internal auditors review whether the agreed-upon controls are in place.
Standards & Procedures	Are lacking or inconsistent.	Information technology security department defines security standards and maintenance procedures, and assists in their implementation across the enterprise.
Security Features & Functions	Are inconsistent, often "home-grown" and undocumented. Maintenance is difficult.	Business managers and technology managers use the product evaluation and rating process developed by the U.S. government, available at no direct cost.

FIGURE 1

HOW TO MARKET THE INFORMATION SYSTEMS SECURITY PROGRAM

Author: David Eakin, CISSP

Organization: Navy Ships Parts Control Center
Code 0444
5450 Carlisle Pike
Mechanicsburg PA 17055-0788

Phone Number: (717) 790-6949
FAX: (717) 790-2876

Introduction

In a recent issue of InfoSecurity News, the results of a nationwide survey were published. Of the 840 respondents to the survey, 67% identified a lack of awareness on the part of management and end users as the greatest obstacle to achieving adequate levels of information security [1]. Information system (IS) security is not a new data processing discipline; most large organizations have had IS security programs in existence for over ten years. Many organizations started their programs as the result of some adverse action (e.g., external audit review, new legislation, extensive damage/embarrassment) as opposed to a visionary need for resource accountability. If management has already acknowledged the need for an IS security program and has dedicated some amount of resources to this program, why is it so hard for IS security professionals to gain management support for the continuity of the program? Several answers come to mind that could explain this situation. Hoping that the personal integrity of IS security professionals would preclude suspicion of fraud, misuse of funds, or dereliction of duty, it would seem that the IS security professional has not done an adequate job in marketing the security program within the organization. This paper attempts to explain how the IS security program can be better marketed within an organization.

Discussion

Business has only two basic functions - marketing and innovation."
- Peter Drucker

Most IS security professionals understand the need for and means of security awareness training. This usually involves informing different groups of IS users (e.g., clerk/typists, data entry, business analysts, mid-upper management) of their security responsibilities in the use of company IS resources as defined in organizational policies. Security professionals understand that different training sessions must be developed to address the specific responsibilities of each IS user group. What is not generally known is that to gain acceptance at any level, the security program must be marketed, and marketed by the security program manager. Marketing in the context of this paper is not selling. Selling is exchanging a product or service for money or its equivalent. Using this thought line, the services of any IS security professional who is employed by an organization have already been "sold". For the purposes of this paper, marketing is synonymous with advertising.

Good marketing is a process whose primary goals are to raise awareness and interest. The object of marketing is not to get clients, but rather to educate them; to get their commitment to support your organization and goals. The goal of marketing the IS security program internally is to promote a general understanding and appreciation of what the IS security program is all about and how it can be utilized as a partner to help meet business goals. Generally accepted hallmarks of a good marketing program are that it is dynamic; it is a long-term process; it is focused on new opportunities; it is a process that shows off the product or service's qualities (if a product or service is bad, a good marketing campaign will help people to become aware of its bad points that much faster.) Marketing should be thought of as the best way of positioning the IS security program for the future.

"The business of America is business." - Calvin Coolidge

IS security helping to meet business goals may seem like a foreign concept. After all, the organization invested resources in IS security staff, training, computer hardware and software. Doesn't that mean that effective security practices speak for themselves? Not if the recommended practices detract (or are perceived to detract) from the main goals of the organization. Enhancing the IS security program is difficult because every enhancement requires a change in attitude and behavior. Most people view the IS security program to be theoretical and concerned more with "the sky is falling" than with likely events. Very few individuals (including many security professionals) have ever been directly involved in or affected by a major IS security incident [2].

As arcane as IS security may be to the layman, everyone can understand that implementing IS security safeguards have adverse aspects (e.g., added costs, slower response time, inconvenience, discouraging workplace environment, etc.) Garnering additional resources for IS security countermeasures will only happen when there is a net profit as measured by the goals of the organization. Sometimes a goal of an organization is just to stay out of trouble. It is worth noting that the banking industry's increased investment in IS security countermeasures was prodded by law and regulation, and in the communications industry because of lost revenues due to piracy of services. In the current business climate of cost cutting, new or expanded IS security enhancements can seem like an impediment to meeting the business objectives.

Our plans miscarry because they have no aim. When a man does not know what harbor he is making for, no wind is the right wind." - Seneca

IS security has to be viewed by the security professional as a means to an end, not an end in itself. The end goal of all business operations (including IS security) is to contribute to the growth and profitability of the organization and to ensure that the organization can effectively accomplish its goals [3]. A proposal to any layer of an organization without regard to the completion of its goals is to invite disaster. The most important task of IS security is ensuring the availability of reliable information, and assuring that the intended meanings are intact and meaningful.

The trouble with this task is that it is an intangible item. Because it is an intangible, it's more difficult to appreciate the benefits and often conveys the perception that it is something that the customer can do without. As a rule, intangibles get lower priority. The more intangible the product or service, the more important it is for you to have specific marketing goals and targets [4].

The image IS security professionals have of themselves and their IS security programs is not necessarily the same image that the rest of the organization has. When the rest of the organization sees you and your program as part of "the team", the IS security program will be functioning as an integral part of the organization's strategy. As stated previously, enhancing the IS security program is difficult because every change in the program requires a change in attitude and behavior. The logical starting place for change is with the IS security professional who has responsibility for the IS security program.

Consider yourself an internal consultant to the organization. Developing a viewpoint that you are an internal consultant means actively perceiving the people and groups in the organization as clients rather than "users". A good consultant's client orientation means that you do not generally work out the solution, but you help your clients to work it out. An attitude of business awareness is the first step towards this change. Most IS security professionals are at a disadvantage here because of the technical nature of the IS security program. Many (if not all) of the current crop of IS security professionals graduated from one of the technical ranks (e.g., data processing, audit, software engineer, etc.) Few IS security professionals have an extensive business, marketing or functional background.

Clearly, the first priority is to develop a sensitivity to what the business is, and how it is accomplished. To do this, stop talking and listen. When you attend staff meetings (assuming you do attend), listen to what "pushes the buttons" of the different levels of people in attendance. Discover what the five most pressing business issues are within your organization. Using Tom Peter's notion of Management by Wandering Around (MBWA), listen to what the general workforce has to say about the IS security program. Just as security awareness training has to be tailored to each specific audience, developing a "business sensitivity" is dependent on the specific organizational group.

Successful marketing of the IS security program comes down to interpersonal skills and how to get something done when you have no authority to do it [5] [7]. More important to other people than your technical skills are your interpersonal skills - what kind of human being you are. Are you someone that others can work with? Someone who is a "team player"? Someone reliable, credible, trustworthy? Someone who can improve situations by virtue of mutual interaction? Paul Ouellette's advice on how to improve this area is condensed into the acronym HEAR:

"It is a curious fact that of all the illusions that beset mankind none is quite so curious as that tendency to suppose that we are mentally and morally superior to those who differ from us in opinion." - Elbert Hubbard

The objective of marketing the IS security program is to market the business benefits that the staff, products, and services provide - not to market themselves. The development of a "we" attitude and a team approach that spreads with each encounter is essential to success. [4] One problem that crops up is ego-building, which is viewed not as an attempt to help clients or the corporation, but to boost one's career. A second problem that surfaces is to focus on the IS security program instead of your audience. Remember that your clients are only interested in what you can do for them. A third problem area is deciding on a solution before you analyze the problem. This usually takes the form of picking a marketing communications vehicle before you think through the factors involved.

All marketing communications vehicles can work - and work well. That doesn't mean that they are all equal. In a given situation, what makes one work while another crashes is the match-up between the medium, message, budget, and audience. When you do it right, creating the right marketing plan offers the IS security professional the best credibility, visibility, and publicity.

References

- [1] Infosecurity News editors. 1993. "The Outlook for Infosecurity," Infosecurity News. January/February
- [2] National Research Council. 1991. "Computers at Risk", National Academy Press, 2101 Constitution Avenue, N.W., Washington, DC 20418
- [3] DeMaio, Harry B. 1992. "You Can't Sell Security If You Don't Talk Business," Information Systems Security. Summer, Vol 1, No 2
- [4] Ouellette, L. Paul. 1992. "How To Market The I/S Department Internally", American Management Association, 135 West 50th Street, New York, NY 10020
- [5] Kay, Russell. 1992. "Alan Krull on Getting Things Done: The Art of Productive Politics," INFOSecurity Product News. July/August
- [6] David, Bernard J. 1993. "Building a Business Case for Information Technology," Beyond Computing. March/April
- [7] Crouse, Harlan W. 1993. "How to Influence Users and Boost Security," Infosecurity News. May/June
- [8] Carey, Cameron. 1991. "Building a Successful Career in Infosecurity," INFOSecurity Product News. November/December
- [9] Isaacson, Gerald I. 1990. "Security Awareness: Making It Work," ISSA Access. 4th Quarter

Related Articles

DeMaio, Harry B. 1993. "Building Enterprisewide Alliances,"
Information Systems Security. Winter, Vol 1, No 4

Beaudry, Mark H. 1992. "Can We Keep Up with the Changing Times?"
Security Management. May

Fisher, Patricia A. P. 1992. "How Can We Add to the Bottom Line?"
Security Management. June

Gale, Stephen and Charles H. Davidson. 1992. "A Model for
Security's Bottom Line," Security Management. September

Browne, Judith A. 1990. "Critical Success Factors for Managing
Information Security," INFOSecurity Product News. June/July

Quotations

Peter, Dr. Laurence J. 1977. "Peter's Quotations - Ideas For Our
Time", Bantam Doubleday Dell Publishing Group, Inc., 666 Fifth
Avenue, New York, NY 10103

H - Hear. Physically hear exactly what is said. Block out external noise and internal arguing. This isn't an opportunity to impress the other person by figuring out how you are going to respond to what the other person is saying, it is an exchange of ideas.

E - Empathize. Have feeling for what the other person is trying to explain. People do what they do for reasons. Their reasons may be rational, emotional, or completely off the wall, but they do have reasons. Understanding the motive of why the other person is telling you something. Understanding that motive is empathy. Understand their motivation and you will have a better chance of being able to help them.

A - Analyze. Take time to reason it out. Don't decide on the answer before you even know the question. Widen your horizons to look at the problem from all angles.

R - Respond. Answer all the questions and cover all the bases. Always respond, even when the answer is "I don't know." Don't feel pressured into responding too quickly. In normal interactions, it has been estimated that there can be a buffer time of up to two minutes of pure listening before the conversation reaches the awkward stage. It is much better to say it right than to say it quickly and superficially [4].

"For who of you, wanting to build a tower, does not first sit down to figure out the expense, whether he has enough to complete it?"
- Luke 14:28

An IS security marketing project should fulfill one or more of the following tasks:

a. Increase awareness of the IS security program's capabilities - let them know what you can do and how it can help others in their business.

b. Generate enthusiasm to have a significantly positive impact on getting problems solved.

c. Provide incentives for the use of your services - IS security products and services must be accepted from within, not imposed from without.

d. Optimize client control and decision making - if your clients feel that they have had a say in the decision making process, that their input received serious consideration, and that they have control over their circumstances, acceptance increases astronomically [4].

All marketing starts with market research. The first thing you have to consider when undertaking a market research program is cost. You should concentrate on increasing the value of the IS security program while maintaining or utilizing the resources you already have. The old maxim "there's no such thing as a free lunch" also applies to marketing the IS security program. There is a cost involved in anything having to do with a business. There is also a cost in not doing something. The good point here

is that all business costs are relative to each other.

The next step is to develop a marketing strategy. The basic elements of a marketing strategy are:

Target + Focus + Impact Factors = Strategy [4].

Targeting your marketing campaign starts with understanding your audience concerns by organizational placement. IS security program clients generally fall within five areas; senior executive staff, senior management, middle management, end user community, data processing staff. Each group has its own needs and concerns. Top management wants to see technology's impact on the goals of the organization; middle managers want information technology to improve business efficiency; people at lower levels in the organization generally want routine tasks to be made easier or automated so they can move on to more interesting duties [6].

In determining which audience you want to address first, consider some important points. Who are the key decision makers that need to be convinced? Which group of clients will have the greatest impact on the success of the organization as a whole? Who are the key players? Remember the "golden rule" (whoever has the gold rules). Even when the budget is controlled by a single manager, this person will frequently seek a wide range of opinions before making a decision. You will have to understand and communicate with such influential users and managers.

After you have targeted your audience, you need to develop a focus. Are you trying to create awareness of or educate clients about something new? Are you merely promoting something different (not something new, just a change)? Is your intent to clarify an issue because your client is confused or misinformed?

Once target and focus have been identified, the marketing impact factors remain to be considered. Determine what the benefits/drawbacks are to the audience. Identify potential obstacles. What indicators do you have that will tell you if you are successful or not? Identify the corporate culture of your organization (conservative, aggressive, etc.). Is there any history that could affect the marketing effort? Are some organizational groups more affected by your recommendations than others?

The way to get resources, win commitments, and gain understanding usually has little to do with the organization's formal structure. However, the marketer must be extremely sensitive to organizational structures and the human personalities that make up those structures.

An example

"Never put off till tomorrow what you can do the day after tomorrow." - Mark Twain

One recent example of a marketing project conducted at the Navy Ships Parts Control Center can be described in the context of the thirteen steps Paul Ouellette recommends in developing a marketing plan [4]:

1. Set goals and objectives. We wanted to stage an event that would clearly show upper-management's advocacy of the IS security program.

2. Establish a measurable outcome. We felt that the amount of personal involvement of upper-management in the event would grade the success of the event.

3. Prepare your audience profile. We targeted the general base-wide workforce as our audience. Our base is quite diverse in geographic layout, with a "core" of administrative buildings housing the majority of computer users/customers. Ages vary from 20's to 60's. We wanted something that would take approximately 10 minutes to see, but provided more for those people who could afford the time.

4. Identify positive forces that apply to the plan. We recently had a change of command and the new Commanding Officer expressed support of the IS security program.

5. Identify negative forces that apply to the plan. Upper-management time is a rare commodity to obtain.

6. Focus on primary concerns. Little funding was available for such an event. Few opportunities for wide exposure existed. Few staff members were available to help with such an event.

7. Decide on your approach. We decided on a seed planting/soft sell approach given the audience and constraints. We decided to be a "gung ho" participant in International Computer Security Day.

8. Choose a theme. The theme of 1992's International Computer Security Day was "Working Together".

9. Find partners. We invited the IS security managers from the other base tenant activities to participate (in whatever capacity they could afford). We also enlisted the assistance of our Command's graphic artist.

10. Design your tactical plan. We planned to have a series of display tables at the entrance to our main cafeteria. Each table would be showing something different. We would have IS security managers/staff present at as many tables as possible to answer any questions. The event would be attended by our Commanding Officer and Executive Officer. Publicity before and after the event would be maximized.

11. Review your plan. Several iterations of a suitable central theme poster were developed. One version of a Computer Aided Instruction software package had to be substituted because it was not "self running". One of the base tenants did not follow through in assisting the planning phases.

12. Execute it. We advertised the event several weeks before hand in our base-wide newspaper and in electronic mail. On the day of the event, we had our public affairs office photograph the Commanding Officer, Executive Officer, and the IS security managers who were participating. These photographs were used in a

follow-up article in the base-wide newspaper. We had three large folding tables (with three different PCs), five easels, a large-screen TV/VCR, and a large cork bulletin board set up to display various posters, hand-outs, virus infection demonstrations, anti-virus safeguard awareness, and a general-awareness video.

13. Evaluate it. The event was judged to have met our objectives. Upper-management's personal appearance and permission to use the facilities showed their commitment to the program. We have received fewer complaints that the "lower echelon (i.e., organizational points of contact) was trying to push something that the management structure was not behind."

Conclusions

To be a success in business, be daring, be first, be different." - Marchant

Communications is the bridge between the marketing plan and the audience. It's the creation of information vehicles to get your message across to the audience. The process involves both form and content. Once you know who the audience is and what the message is, there are numerous ways of communicating it. All marketing communications vehicles can work if done right. Two criteria apply:

1. Use as many vehicles as possible. Just as in real estate the three most important factors are location, location, location; the three most important factors in marketing are exposure, exposure, exposure.

2. Use vehicles that are appropriate to your audience, message, budget, image, and organization norms. [8]

There are three basic types of marketing techniques: mass marketing ("shotgunning"), indiscriminate marketing ("run it up the flagpole and see who salutes"), and target marketing (specific audiences). There are two basic strategies under the three types: positive (aiming for a positive appreciation of you and your program) and negative (trying to reverse an opinion already established). Considering the resources available to most IS security professionals, a positive, targeted marketing campaign is usually the best approach.

Some suggested ways you can "get the message out" are:

- International Computer Security Day
- Senior management briefings
- Departmental presentations
- New employee training programs
- Organization or security newsletters
- Posters
- PC-based training programs
- Videos and movies
- Security conferences
- Pamphlets and brochures
- Novelty items like coffee cups and coasters
- "Brown bag" seminars
- Technology/productivity fair [9]

THE OECD GUIDELINES FOR THE SECURITY OF INFORMATION SYSTEMS:
A LOOK TO THE FUTURE

Christine Axsmith, Esq.
ManTech Strategic Associates, Ltd.
12015 Lee Jackson Highway
Suite 700
Fairfax, Virginia 22033-3338

INTRODUCTION

The Organization for Economic Cooperation and Development (OECD) established "The Guidelines for the Security of Information Systems" (Guidelines) in an attempt to set a common international framework for computer security. The goal of these Guidelines is to establish a common set of principles from which many nations can begin their computer security awareness and practices to foster the proliferation of international trade. The OECD decided that computer security needed to be approached in a manner which started from the same basic building blocks, and these Guidelines resulted. This paper introduces and explains the role and ramifications of the Guidelines by discussing several issues: the Digital Signature Standard, computerized information as evidence, and extradition for international computer crimes. The United States should enact legislation to allow for online contracting, the admissibility of computer information as evidence, and sign Mutual Legal Assistance Treaties for international prosecution of computer crime.

Information turns the wheel of international commerce. Security of information ensures that the international markets and computerized transactions will retain confidence and integrity required for their maximum utility. The end result, if executed properly, will be the proliferation of international trade by confident and quick usage of information systems.

Transborder data flows are increasingly important to the economies of all countries. As the flow of information across borders becomes a cornerstone of economic survival, protection of the information being transferred becomes paramount to effective operations.

Several stumbling blocks stand in the way. One is the international scope of some computer crimes, and the inability of some legal systems to cope with the admissibility of information resident on computers as evidence. These concerns are based on questions regarding the integrity of computer processing and data. Beyond mere admissibility is the issue of accepting the validity of the electronic information as unaltered truth. Another issue is extradition, where, as in the Hanover Hacker case, the criminal and the crime are in two separate countries. Mutual Legal Assistance treaties are used to extradite an alleged criminal from a country to which she or he has fled. Extradition treaties for computer crimes lag behind more traditional crimes such as fraud or drug trafficking. The OECD Guidelines will offer a foundation for future extradition for international computer crimes.

For years, problems with inconsistent standards have made integration of security efforts in different countries nearly impossible. Extradition is very difficult, and information from information systems themselves is not available as evidence in trials in many countries.

ROLE OF THE GUIDELINES

In 1992, the OECD drafted the Guidelines, which attempt to set a framework for common international standards for information security, that would eventually become common to all information systems. More importantly, it calls for legislative action to foster international

enforcement of criminal laws, extradition, and use of computerized information in courts of law.

The United States had considerable input in the drafting these standards. These Guidelines are an important first step to an international framework, and will emphasize to government and business alike the heightened priority that must be placed on information security in a global marketplace.

The aims of the OECD as an organization are "sound economic expansion" and "contribution to the expansion of world trade on a multilateral, non-discriminatory basis."¹ Although OECD Guidelines are not legally binding, these particular guidelines are very important to the information security professional. As the world becomes a smaller place with the advent of new technologies, establishing protocols to handle computer crime is becoming increasingly important, especially as crimes can now be committed without ever setting foot within the country in question. How will these issues be dealt with in the future? Currently nothing allows for the type of rigorous prosecution necessary to follow an international computer criminal from the scene of a crime back to her or his home country and allow for prosecution in the country where the compromised computer resides.

The ultimate value of these Guidelines is in law enforcement, beginning a framework for international prosecution of information security related crimes. The strength of the Guidelines is not their legal enforceability, because there is none, but their political and moral suasion. Sometimes on the international front, treaties and moral suasion can mean very little. In passing an OECD recommendation, the individual countries either sign on to it or they don't, thereby determining the applicability of these Guidelines and enforcement procedures to themselves. This leaves the way clear for a country which does not see itself as able to comply with a method of avoiding compliance. If they do sign, the OECD has procedures to review a country's compliance with the Guidelines, which involve the participation of the noncomplying country in question. In closed door discussions, an agreement is made leading to mutual assistance, or elaboration of the meaning or intent of the Guidelines, or the application of different norms to meet a unique situation. The end result is usually compliance.² The ultimate binding force is a political one.³

Nonetheless, OECD Guidelines in other areas have been mentioned in court decisions, despite lacking any "legal" imperative. In the end, political and moral suasion is the force and effectiveness of the Guidelines, providing conformity to a generalized standard. In the Australian Law Journal⁴ the OECD Guideline on TransBorder Data Flows of Personal Data are cited as a major influence in the area in personal privacy internationally. There is great political pressure on a country to act consistently with an OECD document that they have signed. Not doing so results in embarrassment to the country in question due to inconsistency. Most countries are not eager to be seen in this light. In this manner, the purpose of the OECD Guidelines on information security do fulfill their purpose, namely to increase overall awareness of the need for information security, and the need to set a common framework from which to set working international standards in this area.

OVERVIEW OF THE GUIDELINES

The Recommendation of the Council Concerning Guidelines for the Security of Information Systems describes the aims of the document. They are intended to be general and to promote a general framework from which member nations⁵ can develop standards. More specifically, the Guidelines are intended to raise awareness of risks and appropriate safeguards in information security, to create a general framework from which to develop information security measures, to foster confidence in information systems and facilitate their development, and promote international cooperation in achieving security of information systems. Part of information security is the availability of means by which computer criminals can be extradited and prosecuted. Another part is the confidence placed in information systems by its users. The level of confidence must be high enough to allow for the integration of sophisticated systems into the international marketplace on an even greater level, which in the end would foster economic development globally.

Technologies of the future require security to utilize their full potential. A goal of the Guidelines is establishing a structure that will outlive existing technologies. The aims of these Guidelines reflect that forward vision. The methods for implementing these aims within the charter of the OECD are the exchange of information between member states, consultation, studies, joint projects, close cooperation and coordinated action. Guidelines as set forth by the OECD are primarily statements of goals.⁶

Part one of the Guidelines discusses its aims: to foster a common framework with which countries can communicate their computer security structures to one another, to promote co-operation between the public and private sectors, to foster confidence in information systems, and to promote international co-operation in achieving security of information systems. Part of that process lies in the implementation of these Guidelines, specifically, adjustments and fine tuning in the area of international aspects of criminal law.

Part two of the Guidelines describe very briefly the scope of their application, which broadly encompasses the public and private sectors, and all information systems they contain. They do not supersede existing OECD Guidelines on the protection of Privacy and Transborder Flows of Personal Data. The development of separate information security systems for national security and other information systems is discouraged.⁷ These Guidelines are not intended to be inflexible. It was recognized that deviation might be required "in the areas of national security and maintenance of the public order."⁸ Exceptions should be in the area of implementation, rather than deviation from the principles discussed later, and the Guidelines call for public disclosure of any exceptions. The specifics of implementation of the Guidelines are discussed later in this paper.

Part three provides definitions for: data, information, information systems, availability, confidentiality, and integrity.

In part four, certain principles are proposed to begin discussion of information security issues. The underlying objective is explained as the protection of the interests of those relying on information systems from harm resulting from failures of availability, confidentiality, and integrity

Part five states the underlying principles in connection with the security of information systems:

Accountability Principle - the responsibilities and accountability of parties using an information systems should be explicit.

Awareness Principle - users and owners of information systems should be made aware of basic information security practices.

Ethics Principle - "The rights and legitimate expectations of others should be respected."

Multidisciplinary Principle - Emphasizes that information security development should consider the perspective of all interested parties.

Proportionality Principle - Information security should reflect reliance on the systems and the potential harm resulting from compromise of that information.

Integration Principle - Differing aspects of information security should be integrated.

Timeliness Principle - Parties adopting these Guidelines should establish mechanisms for quick response to challenges to the security of information systems.

Reassessment Principle - Information security should be reevaluated periodically.

Democracy Principle - The security of information systems should not impede the free flow of information in a democratic society.⁹

Most of these principles are familiar to information security professionals. The concept behind these principles is to have member countries working from the same general computer security scheme. The object is to create a seamless web of security and to have the transition from one to another transparent to the user, that the same set of standards would apply, eliminating loopholes between systems and countries.

IMPLEMENTATION OF THE GUIDELINES

Part six of the Guidelines discusses implementation in the areas of policy development, education and training, enforcement and redress, exchange of information, cooperation on international and national levels.

Copyright 1993 Christine Axsmith

The policy development portion of this section outlines issues that need to be addressed in national policies. "Worldwide harmonization" of technical security standards is one of these goals. In doing so, the Guidelines suggests that security solutions reflect the variety of information systems.

Promotion of expertise and "best practice" in the field of information security is another goal of the Guidelines implementation. Again, the specifics of each program would vary according to the needs of the organization and its users.

Three major issues emerge from the Guidelines and their Explanatory Memorandum. They are the call for a digital signature standard and the framework for the resolution of international legal issues including extradition and evidence. Effective implementation of the Guidelines depends on the resolution of three major areas: legal acceptance of the Digital Signature Standard, expanding the use of computerized evidence in court, and extradition treaties.

Digital Signature Standard

Contracts, and the ability to verify electronic "signatures" to such contracts are placed as a high priority by the Guidelines. The drafters of the Guidelines see that such contracting capability is essential to the future development of global information systems. This essentially means the formation of contracts entirely on automated information systems.

An example of this scenario is that a seller of goods would post contracts on a computer accessible by modem link. A buyer could sign on to that computer and purchase some of those goods, forming a legally binding contract on the computer. Such a scenario is possible with a legally acceptable electronic signature standard, and the drafters of the Guidelines see it as an economic inevitability. Its implementation would do a great deal to foster economic development and the increased integration of information systems in the international business world. Processing time and costs of contracting would be dramatically reduced.

Many issues must be resolved before this idea can be implemented, such as enforcement and liability of such contracts,¹⁰ rules regarding electronic signatures and their validity, when such contracts are formed, whether they are unilateral contracts, and what are the changes to U.S. commercial law.

As a signer to the Guidelines, the U.S. is far ahead of most countries. Still, work needs to be done. The Uniform Commercial Code defines "signed" as "includes any symbol executed or adopted by a party with the present intention to authenticate a writing." "Authenticate" is not defined in the Uniform Commercial Code in Articles One or Two. This lack of definition has created confusion in the legal community.¹¹

An electronic signature could be included in the Uniform Commercial Code definition of "signature," but its acceptance depends in large part upon the legitimacy given by the courts.

This area of the law is very undefined. Considerable support exists, but a court has yet to adopt it.¹²

A change to the Uniform Commercial Code to legally include the Digital Signature Standard as a "signature" valid to form a contract is an action which will foster its use. Otherwise, any business would be taking a risk if a contract "signed" online came to litigation later on.

The drafters of the Guidelines foresaw a time when the contracting process would be completely automated. As a result, other legalities in terms of when a contract is formed, and legitimacy of the agreeing signatures, are key issues that need addressed.

Evidence

The second issue raised by the OECD Guidelines is establishing mechanisms for the use of information on computers in courts of law as evidence, and as evidence in administrative hearings, worldwide. Adjustments need to be made on the national level to laws regarding jurisdiction and evidence. This change will allow for the use of computer evidence in another country, aiding in the international prosecution of computer criminals.

An example of this is where a computer criminal breaks into a computer in another country. The Guidelines call for the establishment of procedures whereby the criminal can be tried in the country where the computer sits. For some countries, in the area of computer crime, this need has required an adjustment of current laws to forge an exception for computer crime.

In the U.S., computer records are admissible as an exception to hearsay¹³ under the reasoning that if the information was reliable enough for a business to depend on it in the course of business, then the computer process is reliable enough for its results to be seen by a court in making its decision. The element of unusual reliability of business records is supplied by systematic checking, by regularity, and by continuity which produce habits of precision, and by actual experience of business in relying upon them.¹⁴ What the computer printout demonstrates is that the printout is a correct reflection of what is in the machine, rather than the assertion that the information itself is correct.¹⁵

Presuming that printouts are accepted into evidence as accurate reflections of machine contents, a separate issue is whether the information on the computer is accurate.¹⁶ Currently, the computerized information may be admissible, but then is subject to attack for its validity. The OECD Guidelines call for changes in evidence law as well as accepted use of the Digital Signature Standard to rectify the situation.

Another related issue is the use of information contained on a computer for use later at a trial or administrative hearing. Here, evidence resident on an audit trail could be used in court, if proper identification and authentication standards have been met. That is where the challenges lies. An adequate method of determining what information was on the computer without having been changed must be established and accepted by the U.S. court system.

This could be affected through legislation accepting the Digital Signature Standard as a method to guarantee that the information remains the same, in court.

Effective policy coordination also requires the allocation of risks and liability in the event of an information security breach?¹⁷ Which party bears the risk in cases of fraud? Further development of electronic contracting will require addressing these contracting issues in a computerized context.

Sanctions are essential to the enforcement of any rule. Here, "sanctions for the misuse of information systems" should be embedded into current legislation dealing with computer crime. A "core" of computer related offenses has developed due to legislative actions on the part of the OECD member countries in recent years. The Guidelines do not list the "core offenses". In the U.S., the Computer Fraud and Abuse Act makes the unauthorized entry into a computer a crime in the case of national security or government computers, and a crime to change information resident in a financial system. Recently, a change to the copyright law has made illegal copyright infringement a felony, which applies to software piracy.¹⁸

The effect of the Guidelines is to greatly increase awareness of information security and its importance to the functioning of information systems in terms of user confidence and trust. User confidence is the basis of the functionality of any information system.

International Prosecution of Computer Crimes

The third issue to be addressed and coordinated between countries relates to jurisdiction. Some nations do not necessarily claim jurisdiction where the crime took place, and as a result cases involving computer criminals operating from another country are not available to the courts of that country. Other countries will not extradite a national to another country. Both of these situations will have to be clarified according to the individual nation before international cohesion is attained in the area of extraterritorial jurisdiction. One such method is Mutual Legal Assistance Treaties, which enable law enforcement authorities to obtain evidence in a form admissible in our courts.¹⁹

With differing systems of admitting evidence into courts of law, mechanisms such as Mutual legal Assistance treaties must be in place to facilitate a commonality in the prosecution of computer criminals and civil suits. Establishment of a common framework would allow each nation to adjust its own statutes to work with existing structures in other countries. The eventual aim is to utilize electronically stored information in courts and administrative hearings between each and every country signing the OECD Computer Security Guidelines.²⁰

The Guidelines also mention education to "increase awareness at every level of society" about information security, including ethics.²¹ Training is related to education in the Guidelines, which calls for training various computer professionals, such as ADP auditors, law enforcement, and users or owners of information.

Cooperation is key to setting workable standards that mesh with those of other countries. Countries should report on the information security activities of its territories to harmonize them on a national and international level.

CONCLUSION

On the whole, the security presented in the Guidelines is not new to U.S. information security professionals. The issues that are raised present new legal challenges. The United States should make legislative changes if they are to move into full compliance with the OECD Guidelines for the Security of Information Systems. Adoption of the Digital Signature Standard as a legal signature under the Uniform Commercial Code would be a healthy start for U.S. compliance with the new OECD Guidelines. Recognition of basic computer security techniques as a method of accepting the accuracy of the information resident on computer systems in court would increase the reliance on computers in the business world. Another suggestion is to adopt Mutual Legal Assistance Treaties with other OECD nations signing the Guidelines for the Security of Information Systems to enhance the effectiveness of current laws in bringing computer criminals to justice.

If these actions are taken, the United States will be at a competitive advantage by virtue of fostering the use of computers to facilitate business and cut overhead costs relating to the contracting process.

- 1 Audretsch, *Supervision in the EEC, OECD, and Benelux*, v.36 Intl. Comp. LQ, p.844 (1987). Preamble to the Convention and Article One, OECD.
- 2 Audretsch, *Supervision in the EEC, OECD, and Benelux*, v.36 Intl. Comp. LQ, p.844, 848 (1987).
- 3 Audretsch, *Supervision in the EEC, OECD, and Benelux*, v.36 Intl. Comp. LQ, p.844, 849 (1987).
- 4 v.59 Australian Law Journal p. 683
- 5 Recommendation of the Council Concerning Guidelines for the Security of Information Systems, p. 7
- 6 Audretsch, *Supervision in the EEC, OECD, and Benelux*, v.36 Intl. Comp. LQ, p.844, 845 (1987).
- 7 Explanatory Memorandum to Accompany the Guidelines for the Security of Information Systems, OECD doc OCDE/GD(92)190 p. 25.
- 8 Explanatory Memorandum to Accompany the Guidelines for the Security of Information Systems, OECD doc OCDE/GD(92)190 p. 25.
- 9 Guidelines for the Security of Information Systems, OECD doc OCDE/GD(92)190 pp. 8-10.
- 10 Explanatory Memorandum to Accompany the Guidelines for the Security of Information Systems, OECD doc OCDE/GD(92)190 p. 31.
- 11 Baum, Linking Security and the Law of Computer -Based Commerce, p.4 (11/10/92) NIST workshop on Security Procedures for the Interchange of Electronic Documents.

- 12 American Bar Association, Section of Science and Technology, Resolution affecting Electronic Commerce and Information Security, # 115, approved 8/19/92.
- 13 Fed. R. Evid. Rule 803(6).
- 14 Fed. R. Evid. Rule 803(6).
- 15 US v. Vela, 673 F.2d 86 (5th Cir. 1982).
- 16 E. Cleary, McCormick on Evidence, (3rd ed. 1984) p. 886.
- 17 Explanatory Memorandum to Accompany the Guidelines for the Security of Information Systems, OECD doc OCDE/GD(92)190 p. 32.
- 18 Criminal Penalties for Copyright Infringement, Pub. L. No. 102-561, 106 Stat 4233.
- 19 Kreczko, *US Practice: Contemporary Practice of the United States Relating to International Law*, v.86 American Journal of International Law P. 548, 550, July 1992.
- 20 Explanatory Memorandum to Accompany the Guidelines for the Security of Information Systems, OECD doc OCDE/GD(92)190 p. 34.
- 21 Explanatory Memorandum to Accompany the Guidelines for the Security of Information Systems, OECD doc OCDE/GD(92)190 p. 35.

THE DRAFT FEDERAL CRITERIA AND THE ITSEC: PROGRESS TOWARDS ALIGNMENT

Julian Straw

Secure Information Systems Ltd (SISL)
Sentinel House, Harvest Crescent, Ancells Park
Fleet, Hampshire, GU14 8UZ, England. Tel. +44 252 811818

Abstract

This paper draws comparisons between the European ITSEC and the draft Federal Criteria, and assesses the problems in achieving criteria alignment and mutually acceptable evaluations. A representation of the ITSEC in terms of the Federal Criteria assurance components is presented.

Introduction

Governments in both North America and Europe have successfully operated schemes for the independent certification of information technology products for some years. The growing body of certified products enables purchasers to select products which have been tested against published standards, and to specify standards in procurement which vendors must meet.

The Trusted Computer System Evaluation Criteria [TCSEC], was the first standard for such evaluations to achieve widespread acceptance. This was established for the evaluation of secure operating systems, although interpretations for networks and databases have been produced and applied. The need to evaluate systems for government use, and a wider variety of products, led a number of countries in Europe to develop their own national criteria, for example the German Federal Criteria and UK CESG Confidence Levels. In 1989 an initiative by France, Germany, the Netherlands and the United Kingdom led to the harmonisation of these criteria into the Information Technology Security Evaluation Criteria [ITSEC], now published under the aegis of the European Commission. These criteria introduced a number of concepts, notably the separation of functionality and assurance, the division of assurance into correctness and effectiveness, and the use of a security target document to define the basis for an evaluation.

The ITSEC have now achieved widespread acceptance in Europe, and are the basis of commercially operated evaluation schemes in Germany and the United Kingdom. Market demands are now leading many vendors to seek product evaluations against both the ITSEC and the TCSEC in Europe and the US, respectively. There is a strong desire within both vendors and procurers of IT products and systems to establish a single set of criteria, evaluations against which would be accepted in all markets. This would avoid the need for duplicate evaluations, and would facilitate the transfer of security technology between nations.

Against this background the National Institute of Standards and Technology (NIST) and the National Security Agency (NSA) have now published the draft Federal Criteria [FC1 and FC2] for comment. This document is intended to evolve into a new Federal Information Processing Standard, for use by the US Federal Government, and by others as desired and appropriate. Inevitably this new document will be compared with the TCSEC and the ITSEC, to analyse comparative strengths, and to assess its suitability for adoption as an international standard.

This paper offers an initial comparison of the Federal Criteria with the ITSEC in terms of its component parts, and provides a basis for comparison of the assurance levels in the two sets of criteria.

Structure of the Federal Criteria

The Draft Federal Criteria are published in two volumes. Volume I (FC1) defines the criteria, and specifies the requirements to be met for a successful product evaluation. Volume II (FC2) illustrates how the criteria may be used to construct specifications for evaluations. A full description of the criteria is beyond the scope of this paper, but a brief outline is given below.

Requirements for an evaluation under the Federal Criteria are defined in a protection profile. This document describes the functionality and assurance requirements to be met by products intended to satisfy specific security needs. The protection profile does not relate to a specific product, but rather defines a standard which products must meet. In addition it presents a rationale based on assumptions concerning threats, environment and intended usage. Each profile is analysed for technical correctness, completeness and consistency, before being registered as a suitable basis for evaluations.

Functionality is expressed using a selection from sixteen components (e.g. Access Control), each of which is divided into levels. The functionality specified in a protection profile is constructed from one or more of these building blocks, taking into account any recognised interdependencies.

Assurance is divided into development and evaluation components. Development assurance specifies criteria which a developer must satisfy in the construction of a product, and in its development process. Evaluation assurance defines tasks which must be undertaken during an independent evaluation process. Example assurance classes (T1 to T7) have been constructed for use in profile construction.

Comparison with TCSEC

ITSEC	TCSEC	Federal Criteria	
		Protection Profile	T Level
F-C1,E1	C1	-	-
F-C2,E2	C2	CS-1	T1
F-B1,E3	B1	LP-1	T2
-	-	CS-2	T2+
-	-	CS-3	T3+
-	-	-	T4
F-B2,E4	B2	LP-2	T5
F-B3,E5	B3	LP-3	T6
F-B3,E6	A1	LP-4	T7

TABLE 1

The ITSEC contain a table (paragraph 1.39) showing intended correspondence of the example functionality classes with the TCSEC classes. This comparison, shown in the two left hand columns of Table 1, has been the subject of previous analysis [BRANSTAD] [VDMA] and is well known, although by no means universally accepted. Recent ITSEC evaluations of products designed to meet TCSEC requirements have provided support for the comparison, although above B1 differences in

deliverable requirements, notably in the areas of semiformal and formal expression and traceability of requirements, may cause difficulties.

This comparison is extended in the Federal Criteria which provides example protection profiles to supersede the TCSEC levels above C1. Each of these profiles incorporates one of the defined T assurance levels, as shown in Table 1. However, no analysis is provided to support the comparability of the new protection profiles with the TCSEC levels.

Basis for Comparison with ITSEC

There is widespread agreement that alignment of security evaluation criteria will yield benefits for both vendors and procurers of secure products. A comparison between the Federal Criteria and the ITSEC is required for two reasons. Firstly, to establish differences which represent only variations in approach or emphasis, and those which are potential barriers to international harmonisation. Secondly, given that evaluations may be conducted under the new criteria, some basis for comparison with ITSEC evaluation levels has to be established.

Looking first at differences in approach, some key issues can be identified for examination:

Approach to definition of an evaluation

Protection profiles define the basis for evaluations conducted using the Federal Criteria. In this respect they fulfil a similar role to that of an ITSEC security target. If the criteria are to be aligned then a means of mapping evaluations from one to the other needs to be found, and in order to do this the different approaches must be examined.

The most obvious distinction between a protection profile and a security target relates to the degree of product independence. A security target refers to a specific product, and completely defines the target of evaluation. A protection profile does not (and indeed must not) relate to a specific product, but rather represents a benchmark against which specific products will be assessed.

The Federal Criteria require categorical and descriptive information to uniquely identify, register, and cross reference a protection profile in a registry of profiles. The protection profile must also include a description of the information protection problem to be solved.

A security target must provide a description of the Target of Evaluation (TOE), providing sufficient detail for unique identification, and giving an indication of purpose and function. No registration process is necessary, and the assessment process is performed during an evaluation. Before an evaluation can commence the national Certification Body determines whether an evaluation against a proposed security target would be a suitable basis for certification.

The descriptive requirements of the two sets of criteria are similar, although detailed differences will arise insofar as a protection profile covers a class of products rather than a specific system or product. The need for protection profile registration is a major difference, although the analysis performed during the registration process is similar to that carried out by the UK Certification Body and the Commercial Licensed Evaluation Facility (CLEF) before and during an evaluation.

A Federal Criteria protection profile must include a justification for its existence, which must list assumptions about threats, intended environment and method of use. It must also indicate what support is provided for organisational security policies.

A security target must provide the necessary information for a prospective purchaser to decide whether it will help satisfy his system security objectives, and defines what else must be done to

meet those objectives. It should identify the intended environment and method of use, and the assumed threats. The security target does not need to provide a specific justification for its existence, since it does not undergo a registration process.

The content of the rationale sections of the two documents are similar, but the need to describe the relationship between the functionality provided and the security objectives is absent in the protection profile. The guidance on production of profiles requires that this be considered, but the analysis does not need to be included in the document. This is borne out by an examination of LP-1, which does not relate the functionality specified to the identified threats, although more of this material is provided in the commercial profiles supplied. This requirement is present in the ITSEC in both the security target rationale and in the suitability analysis.

The Federal Criteria requirement is to establish the boundary of responsibility for information protection which must be provided by an IT product, such that the expected threats to information within this boundary are countered. Assessment of this requirement is done during the analysis phase, when a profile is submitted for registration. The Federal Criteria state that "pre-specified or unique" functional components may be used, corresponding to the ITSEC's optional use of functionality classes. However, this option does not appear to be discussed elsewhere.

The ITSEC impose few restrictions on the specification of functionality. To date few sponsors have used the predefined functionality classes, either because they were inappropriate to the type of product, or because they wished to claim unique features. Reliance is placed upon the evaluators to ensure that the functionality is clearly expressed.

The ITSEC require a semiformal specification of security enforcing functions to be used at E4 and E5, with a formal specification at E6. This has no parallel in the Federal Criteria, which must presumably exercise very careful control of the expression of functionality. Attempts to express TCSEC functionality semiformally have revealed ambiguity in these criteria (see for example [LAMP]).

An ITSEC security target must state the minimum strength of mechanism applicable to the TOE. This requirement is not covered by the Federal Criteria.

The Federal Criteria anticipate that protection profiles will be produced by consumers or producers within the government or the private sector in response to a specific need for information protection. Following production, the profile will be analysed using the guidelines contained in [FC1, 3.5] before being approved as a basis for evaluations. The specific details of profile registration are currently under development, although it is anticipated that different types of profile registration will be possible as follows:

- a) A complete protection profile;
- b) A functional package;
- c) An assurance package.

The likelihood of success of this route is difficult to assess, but there is little precedent for the market led generation of security standards. Users of security products are generally not well placed to do this, other than through an existing body with a remit to establish standards in other areas. Vendors will endeavour to ensure that the profile favours their own product, and agreement may only be possible when all participants already provide the same features. Development of the TCSEC Trusted Database Interpretation (TDI) took place over a period of years, and there is little reason to expect rapid, spontaneous development of new protection profiles.

If the Federal Criteria become widely accepted there may be more than one body with responsibility for approval of protection profiles. In this case some form of international agreement would be necessary to maintain standards, and to avoid duplication and overlap.

Responsibility for the production of ITSEC security targets rests with the sponsor of a product or system evaluation. In practice this has led to a variety of approaches and styles. Reliance is placed upon guidance contained in the ITSEC, and consultancy provided by CLEFs to ensure consistent quality. As a result security targets for similar secure operating systems or databases may be difficult to compare. On the other hand conformance of two products to a single protection profile does not allow the different features of the products to emerge, and some security features may not be evaluated.

The protection profile approach provides functional standards against which products may be judged. Unless a security target references a predefined functionality class, it does not allow for the same type of comparison. However, a security target enables all of a product's security features to receive evaluation, whereas a protection profile, unless a perfect match can be obtained will only allow a subset to be evaluated.

The level of skill required for analysis of protection profiles will be considerable. The process requires an understanding of user requirements, the capabilities of products in the relevant market area, and the criteria themselves. The assessor must consider not only technical soundness, consistency and evaluation capability, but is also required to consider usefulness. Under the ITSEC, responsibility for assessment of the security target is split between the Certification Body and the CLEF, and the criteria for approval are more limited than for a protection profile. Assessment is aimed at ensuring that a minimum standard is met, rather than achieving high standards.

Range of products which can be evaluated

The ITSEC were designed to apply to a wide range of systems and products, and a variety of evaluations have now been carried out which has helped to validate this concept. It is contended that any internationally accepted criteria must allow a similarly wide variety of evaluations if significant benefits are to be gained. The Federal Criteria, developing a concept from the Canadian Criteria, provide a set of functionality building blocks [FC1, 4] which have been applied in the construction of the example protection profiles in [FC2]. The extent to which this approach will permit an acceptable range of evaluations needs to be assessed.

The Federal Criteria state [FC1, 4.1] that the functional components defined "allow the definition of protection profiles that closely capture the functional characteristics of IT products evaluated under the existing standards". Unfortunately, all the existing standards used are based on the TCSEC, which was designed to meet the requirements for monolithic operating systems, and this has resulted in a strong bias of the defined functional components in this direction. Even the new availability and TCB ease of use components are strongly influenced by this.

Many secure products are not general purpose, and are designed to provide very specific security functionality. Examples of such products are commercial encryption devices, domain separation products, privacy enhanced mail products and smart cards. In order to evaluate the security features of such products in a worthwhile manner a method must be found of expressing their functionality precisely. Any approach using a restricted set of building blocks, even if this set were expanded, will constantly encounter new products for which the existing set is inadequate.

The ITSEC approach allows vendors to specify arbitrary functionality. It is assumed that this approach has been rejected in the Federal Criteria on the grounds that comparisons would be made difficult. The ITSEC approach has worked well in practice, and the flexibility which this allows would not be given up willingly by vendors. Many of the concepts introduced for protection profiles,

for example the functional building blocks, can usefully be applied to the production of ITSEC security targets. At present experience in the generation of protection profiles is limited. The Federal Criteria contain seven profiles, three of which are aimed at the commercial market, and four for the protection of nationally classified information. All of these profiles address monolithic operating systems. The extent to which the concept can be applied to other types of product, for example databases and encryption devices, has yet to be demonstrated.

Use of criteria in system evaluations

The Federal Criteria [FC1, 1.2] are designed to meet the requirements for product evaluations only, and do not address the issues related to evaluation of systems. The ITSEC are intended to meet the evaluation of both systems and products, contending that this will make it both easier and cheaper to evaluate systems containing products which have already been evaluated [ITSEC, 1.6].

The ITSEC contend that from the point of view of security, the main difference between systems and products lies in what is certain about their operational environment. A system is contained in a defined real world environment, and is designed to meet the requirements of a specific group of end users. For a system the security threats are known, and do not have to be assumed. From the point of view of evaluation, a system is treated as a specific instance of a product, or collection of products, with the assumptions concerning the operational environment replaced by facts. During a system evaluation the operational environment will be examined.

The Federal Criteria make a similar distinction between products and systems. Systems are generally constructed from a number of hardware and software components, with some level of customisation and integration. A principal distinction is identified as the difference in what is certain about the operational environment. The composition of multiple products into a system is judged to be beyond the scope of the standard, and is to be addressed in future publications [FC1, 1.2].

The approach taken in a protection profile is similar to that of an ITSEC security target. The ITSEC security target covers both systems and products, and is similar in content to a protection profile (see section below), and there appears to be no reason why the techniques used in construction of protection profiles could not be adapted to construct the basis for a system evaluation. The main reason for exclusion of systems resides in the problem of composability: the combination of evaluated products to provide a system suitable for certification.

In the UK security targets for systems are usually based upon a System Security Policy (SSP) and a System Electronic Information Security Policy (SEISP). Acceptance of these documents as a suitable definition of the security requirements for a system is an accreditation issue, and is undertaken prior to commencing the evaluation process. This approach has parallels with the proposed system for approval of protection profiles. One problem which would affect the construction of system protection profiles is the limited set of functionality components currently defined.

The UK Scheme encourages the use of evaluated products to build secure systems, and the basis for this approach has been documented [SMITH]. The process is not without difficulties, and the ITSEC effectiveness criteria provide an important contribution. It must be conceded that to date little experience has been gained in the evaluation of complex or high assurance systems using the ITSEC, and their applicability to such systems remains to an extent untested. A supply of evaluated products allows the construction of cost-effective secure systems, and the issue of composability must be tackled as a priority if alignment is to be achieved.

Approach to assurance requirements

The Federal Criteria distinguish assurance derived from the development process and that from evaluation. This approach differs from the ITSEC division into correctness and effectiveness, which provides six levels of assurance each of which is indivisible. The Federal Criteria provide twenty-eight components from which assurance profiles can be constructed. The ITSEC have been criticised in some quarters for providing too many assurance levels, and the complexity of the Federal Criteria approach may cause concern.

The idea that assurance derives from the quality of the development process and from a subsequent independent evaluation provides a useful basis on which to build assurance requirements, and this idea is embodied in the ITSEC as well as the Federal Criteria. The ITSEC approach is to include evaluator actions after each assurance requirement, and this layout makes determination of the complete requirement easy.

The Federal Criteria extract the evaluation assurance requirements into a separate chapter [FC1, 6]. This approach can lead to confusion, and the extent to which new requirements are added is unclear. For example, this chapter contains separate requirements for functional and penetration testing by the evaluator, which build on the requirements for developer testing in the previous chapter. Testing requirements could usefully be presented in one place. Some of the sections (e.g. 6.3.5 DA) simply require the evaluator to check that the developer has met the requirements of the development assurance section. Whilst Chapter 6 contains a good summary of the activities to be carried out during an evaluation, the concept is not yet fully developed, and the usefulness of this division into development and evaluation assurance is open to question. The assurance component Flaw Remediation (FR) is an interesting new area not considered by the ITSEC, which looks at provisions for support of the product during its operational life.

Effectiveness Analysis

The ITSEC divide assurance requirements into effectiveness and correctness, an approach which has received widespread acceptance in Europe. The Federal Criteria provide a new division, between development and evaluation assurance. The concepts used in effectiveness analysis have proved useful, and the extent to which these are incorporated in the Federal Criteria requires investigation.

Suitability

The ITSEC address suitability in two different contexts. The security target for a product or system must correlate the security enforcing functions to the intended method of use, and the suitability analysis must determine whether the security enforcing functions and mechanisms counter the identified threats.

The Federal Criteria contain no explicit requirement to document in the protection profile the relationship between functional requirements and expected threats or intended method of use. However, this operation does form part of the activities conducted during protection profile analysis and construction. This analysis forms part of the registration process for protection profiles, and parallels the inspection of an ITSEC security target conducted by a CLEF and confirmed by the Certification Body. Suitability is covered by the Technical Soundness assessment, which checks that the protection profile is "technically sound and reasonably balanced, considering the profile rationale (threat, usage and environment assumptions), and the functional and assurance requirements".

Whereas the ITSEC suitability examination is based upon a product specific security target, that done for the Federal Criteria will be at the abstract protection profile level, with a further stage of matching the protection profile against a particular product specification. This is a valid approach.

Binding

An ITSEC binding analysis investigates the ability of the security enforcing functions and mechanisms of a TOE to work together in an effective and mutually supportive manner. The current consensus is that it should be performed using information at the architectural level, unless the absence of sufficient information at this level makes necessary consideration of the detailed design. The concept is particularly useful in the analysis of systems built from certified components.

A suggested comparison with the Federal Criteria relates to the dependency analysis carried out during protection profile construction. On first consideration this comparison appears invalid, since the relevant dependency analysis is a requirements level analysis, and is not performed at the architectural level. However, many of the Federal Criteria functional components are architectural constraints, rather than user functionality, and the comparison may have some validity.

This would imply that the binding analysis does not form part of the evaluation process, but is conducted during protection profile registration. As a consequence it is performed in a product independent manner. The comparability of such an analysis, which does not consider the product itself, with an ITSEC binding analysis is open to doubt. The test analysis component of development assurance (TA) provides for an investigation of possible ways to bypass protection mechanisms at the higher levels. This could be considered to cover some aspects of binding. However, beyond this and dependency analysis, no direct parallel with ITSEC binding analysis was found in the Federal Criteria during this examination.

Strength of Mechanisms

Elements of the strength of mechanisms concept may be found in the development assurance penetration analysis (PA). Such analysis is not introduced until T3, which is above the B1/E3 level, and would thus not apply to the majority of evaluations currently performed. The basic analysis at this level would not provide the same level of analysis as the ITSEC requires, as strength is *tested* in the Federal Criteria rather than *analysed*. PA-1 is also limited to testing of unprivileged user access, such testing being derived from system reference manuals rather than design documentation.

The evaluation assurance components independent testing (IT) and implementation analysis (CI) do not contain requirements which relate directly to strength of mechanisms analysis. The former relates to correctness, and corresponds with the ITSEC implementation requirements. The latter deals with the analysis of samples of source code, checking for adherence to standards, and searching for defects.

The test analysis component (TA) refers to an analysis of ways to "bypass...or otherwise defeat the product's TCB", but this is also not used until T3.

Construction Vulnerability Analysis

The requirement for a sponsor to provide penetration analysis corresponds with the ITSEC concept of construction vulnerability analysis. However, as stated above this concept is not employed until T3, and so would not apply until higher level commercial or B2 equivalent evaluations. The use of flaw hypothesis testing in the Federal Criteria provides a structured approach to testing which more closely parallels the structured ITSEC analysis.

In the ITSEC, vulnerability analysis is begun by the developer and continued during the evaluation. This corresponds with the development assurance (PA) and evaluation assurance (IT) elements in the Federal Criteria.

Operational Vulnerability Analysis

The development assurance component penetration analysis (TA) provides a close parallel with the ITSEC operational vulnerability analysis. In the Federal Criteria the developer is required to conduct analysis to discover such vulnerabilities. In this sense the requirement is more onerous than under the ITSEC, where the developer is merely required to analyse known vulnerabilities. Again, however, penetration analysis is only required at T3 and above.

Ease of Use

The comparison of ease of use requirements is interesting as the Federal Criteria cover this aspect under functionality, and the ITSEC under effectiveness assurance.

In both sets of criteria the emphasis is on administration. The ITSEC requirement is very weak, requiring that the entry of the TOE into an insecure state when an administrator or user may reasonably believe otherwise should be detectable. A TOE cannot fail evaluation on ease of use grounds unless an exploitable vulnerability exists.

Treatment of ease of use as functionality allows more specific requirements to be imposed which contribute to what is essentially the same objective in both criteria. The Federal Criteria emphasise the importance of well defined functionality and default settings for security parameters, which must be fail-safe at the higher levels. This makes it clearer what the developer must do, and what the evaluator must check. The existence of fail-safe defaults (which are assumed to mean defaults set to the most secure option) is a powerful means of ensuring that the ITSEC ease of use criteria is met.

The Federal Criteria ease of use concept is more restrictive than that of the ITSEC, but may represent a useful improvement.

Comparison of ITSEC and Federal Criteria Assurance Levels

To conduct a more detailed analysis of the relationship between the ITSEC and the Federal Criteria it is necessary to provide comparisons at a more detailed level than contained in Table 1 above. Comparisons will inevitably concentrate on assurance, rather than functionality. It is unlikely that a given T Level will correspond directly with an ITSEC E Level, and a comparison in terms of each separate component must be performed.

There is more than one method of achieving this. The analysis provided here takes each assurance component of the Federal Criteria and relates each of the described levels described to a particular E level. This was performed in a reasonably objective manner, taking the level of each Federal Criteria component required to meet the ITSEC requirement at each level. This analysis proved to be easier with some components than others, and subjective judgement was applied in some cases. Only in the case of FR Flaw Remediation could no comparable concept be found in the ITSEC. In some cases relationships could be identified at some levels only. In these cases some interpolation was performed. The following table provides a first attempt to express the ITSEC E levels in terms of the Federal Criteria assurance components.

DEVELOPMENT ASSURANCE COMPONENTS¹

	E1	E2	E3	E4	E5	E6
DEVELOPMENT PROCESS						
TCB Property Definition	PD-1	PD-1	PD-2	PD-3	PD-3	PD-4
TCB Design						
TCB Element Identification		ID-1	ID-1	ID-1	ID-2	ID-2
TCB Interface Definition	IF-1	IF-1	IF-2	IF-2	IF-2	IF-3
TCB Modular Decomposition		MD-1	MD-2	MD-3	MD-3	MD-3
TCB Structuring Support		SP-1	SP-2	SP-2	SP-3	SP-3
TCB Design Disciplines				DD-1	DD-2	DD-2
TCB Implementation Support	IM-1	IM-3	IM-4	IM-4	IM-4	IM-4
TCB Testing & Analysis						
Functional Testing	FT-1	FT-1	FT-2	FT-2	FT-3	FT-3
Penetration Analysis				PA-1	PA-2	PA-3
Covert Channel Analysis				CCA-1	CCA-2	CCA-3
OPERATIONAL SUPPORT						
User's Security Guidance	UG-1	UG-1	UG-1	UG-1	UG-1	UG-1
Administrative Guidance	AG-1	AG-1	AG-2	AG-2	AG-3	AG-3
Flaw Remediation						
Trusted Generation	TG-1	TG-1	TG-1	TG-1	TG-1	TG-1
DEVELOP. ENVIRONMENT						
Life Cycle Definition			LC-2	LC-2	LC-2	LC-2
Configuration Management			CM-1	CM-3	CM-4	CM-4
Trusted Distribution		TD-1	TD-1	TD-1	TD-1	TD-1
DEVELOPMENT EVIDENCE						
TCB Protection Properties	EPP-1	EPP-1	EPP-1	EPP-3	EPP-3	EPP-4
Product Development		EPD-1	EPD-2	EPD-3	EPD-4	EPD-5
Product Testing & Analysis						
Functional Testing		EFT-1	EFT-1	EFT-1	EFT-2	EFT-3
Penetration Analysis				EPA-1	EPA-2	EPA-3
Covert Channel Analysis				ECA-1	ECA-2	ECA-2
Product Support		EPS-1	EPS-1	EPS-2	EPS-2	EPS-2

EVALUATION ASSURANCE COMPONENTS²

TESTING						
Test Analysis	TA-2	TA-2	TA-3	TA-4	TA-4	TA-5
Independent Testing		IT-2	IT-2	IT-3	IT-3	IT-4

¹ There are no explicit requirements in the ITSEC for a developer to perform penetration or covert channel analysis. Such analysis is, however, normally required at E4 and above, partly to meet effectiveness requirements for construction and operational vulnerability assessment, and partly to meet the detailed design requirement to demonstrate minimisation of the potential for violations of security.

² The comparison of evaluation assurance components is especially difficult. These do not vary greatly with level in the ITSEC or the Federal Criteria, and the variations are often quite subtle (e.g "small" changes to "moderate". The type of information required also varies greatly and the comparison is more qualitative than for development assurance.

REVIEW						
Development Environment		DER-2	DER-2	DER-2	DER-2	DER-2
Operational Support	OSR-1	OSR-1	OSR-1	OSR-1	OSR-1	OSR-1
ANALYSIS						
Design	DA-1	DA-1	DA-1	DA-1	DA-1	DA-1
Implementation			CI-1	CI-2	CI-2	CI-3

TABLE 2

The next step is to relate these sets of assurance components to the sets used to construct the T Levels. Since E3 and B1 are the most common target evaluation levels for operating systems, ITSEC E3 will be used as a basis for comparison. The table below compares ITSEC E3 with FC T2 (T2 is used to construct LP-1). The results, shown in Table 3, indicate only two areas in which the T2 requirement is more onerous than for E3. In a number of cases, initial inspection suggests the E3 requirement to be considerably in excess of T2.

The T3 assurance class provides a closer match with E3 requirements, but even T3 does not introduce any analysis of implementation (CI). The closest match appears on initial investigation to be between E3 and T4. The root of this problem lies in the claimed compatibility between B1 and LP-1, rather than in any major difference between the ITSEC and the Federal Criteria.

	ITSEC E3	FC T2	COMPARISON ³
DEVELOPMENT PROCESS			
TCB Property Definition	PD-2	PD-2	=
TCB Design			
TCB Element Identification	ID-1	ID-2	-
TCB Interface Definition	IF-2	IF-1	+
TCB Modular Decomposition	MD-3	-	+++
TCB Structuring Support	SP-2	SP-1	+
TCB Design Disciplines	-	-	=
TCB Implementation Support	IM-4	-	++++
TCB Testing & Analysis			
Functional Testing	FT-2	FT-1	+
Penetration Analysis		-	=
Covert Channel Analysis	-	-	=
OPERATIONAL SUPPORT			
User's Security Guidance	UG-1	UG-1	=
Administrative Guidance	AG-2	AG-1	+
Flaw Remediation	-	?	?
Trusted Generation	TG-1	TG-1	=
DEVELOP. ENVIRONMENT			
Life Cycle Definition	LC-2	-	++
Configuration Management	CM-1	-	+
Trusted Distribution	TD-1	-	+
DEVELOPMENT EVIDENCE			

³ Each + in this column indicates the number of levels within a component by which the ITSEC E3 requirement exceeds that of T2. Each - shows the extent to which the T2 requirement exceeds E3. A = indicates broad equality.

TCB Protection Properties	EPP-1	EPP-2	-
Product Development	EPD-2	EPD-1	+
Product Testing & Analysis			
Functional Testing	EFT-1	EFT-1	=
Penetration Analysis	-	-	=
Covert Channel Analysis	-	-	=
Product Support	EPS-1	EPS-1	=

TESTING			
Test Analysis	TA-3	TA-1	++
Independent Testing	IT-2	IT-1	+
REVIEW			
Development Environment	DER-2	-	++
Operational Support	OSR-1	OSR-1	=
ANALYSIS			
Design	DA-1	-	+
Implementation	CI-2	-	++

TABLE 3

Conclusions

Publication of the draft Federal Criteria represents a significant step forward for both the US evaluation process and international harmonisation. In adopting a modular approach to the definition of evaluation requirements the Federal Criteria will enable a wider variety of products to be evaluated than was possible under the TCSEC. A number of features have been incorporated from the ITSEC, notably the separation of functionality and assurance, and the protection profile, which is similar in many ways to an ITSEC security target.

However, a number of differences remain relating both to the structure of the criteria and to the evaluation process. There is clearly a much stronger desire in the US than in Europe to control the nature of evaluations performed. This is reflected in the approval process for protection profiles, the control over functionality components, and the desire to establish classes of evaluated products, rather than leaving the definition of functionality to vendors (subject to ratification). The European market requires a single set of criteria for the evaluation of both systems and products, an issue which has yet to be fully addressed by the Federal Criteria. The ITSEC concept of Effectiveness has yielded useful results in evaluations, and although the Federal Criteria have improved the Ease of Use aspect, other parts of the analysis do not appear to be addressed to the same extent.

The authors of the Federal Criteria have drawn on the extensive US experience in the evaluation of operating systems, and this is evident in the definitions of functional components, and in the requirements for assurance. The extent to which this strength in operating systems will prove a handicap in providing criteria suitable for the whole range of secure products is difficult to assess at this stage, and is a concern.

The extensive range of Federal Criteria assurance components makes comparisons with the ITSEC complex. However, the limited analysis undertaken in this paper indicates that it is possible to use these components to build assurance profiles broadly equivalent to ITSEC assurance levels; and in this respect alignment will be facilitated.

The draft Federal Criteria is a major achievement in terms of new ideas clearly presented and supported by a thorough rationale. Whilst a number of issues have yet to be resolved before alignment of criteria between Europe and North America can be achieved, both parties are now moving in the same direction, and a basis for mutually acceptable evaluations should be attainable.

References

- [FC1] Federal Criteria for Information Technology Security, Volume I, Protection Profile Development, Version 1.0, National Institute of Standards and Technology and National Security Agency, December 1992
- [FC2] Federal Criteria for Information Technology Security, Volume II, Registry of Protection Profiles, Version 1.0, National Institute of Standards and Technology and National Security Agency, December 1992
- [TCSEC] Department of Defense Trusted Computer System Evaluation Criteria, DoD 5200.28-STD, December 1985
- [ITSEC] Information Technology Security Evaluation Criteria, Provisional Harmonised Criteria, Version 1.2, CEC, June 1991
- [BRANSTAD] "Apparent Differences between the US TCSEC and the European ITSEC" Martha Branstad, David Brewer, Christian Jahl and Helmut Kurth, 14th National Computer Security Conference, October 1991
- [LAMP] "Specifying ITSEC Functionality Classes using the Claims Language" Richard Lampard, NPL Report DITC 186/91, August 1991
- [SMITH] "Re-Use of Evaluation Results" Jonathan Smith, 15th National Computer Security Conference, October 1992
- [VDMA] "Towards Mutual Recognition of Security Evaluations" VDMA/ZVEI Working Group on IT Security, October 1991

IT-Security: - a Quality Aspect !

Quality Assurance in the ITSEC-Evaluation Environment in Germany

(Authors: K. Keus, W. Kurth, D. Loevenich)

Bundesamt für Sicherheit in der Informationstechnik (BSI) ¹

German Information Security Agency (GISA)

Department: Scientific Fundamentals and Certification

Division: Accreditation and Standardisation

Edited By: Pat Toth, NIST

Bundesamt für Sicherheit in der Informationstechnik

Godesberger Allee 183; 5300 Bonn 2; Germany

Telefon: (0049) 228 / 9582-0

Fax: (0049) 228/ 9582-400

Table of Contents:

1. Motivation / Background / Objectives
2. Summary
3. Relevant Criteria, Prescriptions and their application
4. A first common approach
5. A german example: "the V-Model"

Abstract

The field of IT-Security is influenced by far-reaching requirements concerning quality aspects. These requirements are implemented according to several standardized criteria and instructions which are responsible for different aspects and phases during the life-cycle of an IT-product. Fundamental Basic Criteria as the ISO Standard set 9000 (quality assurance in design / development and production), the EN/DIN 45001 (installation and the behavior of an evaluation facility (ITSEF)) or the EN/DIN 45011 (installation and behavior of a Certification Body (CB)) build the basic for a quality oriented evaluation environment. The IT-Security Specific Criteria as the ITSEC and the ITSEM define the specific requirements for the development, the evaluation and certification of IT-Security products and IT-Security systems.

In this paper we present a brief survey of the dependencies of relations between the Fundamental Criteria and the IT-Security Specific Criteria with respect to evaluation and certification of IT-products including IT-Security requirements.

1. Motivation / Background / Objectives

IT-Security is embedded in a large framework of quality requirements. The quality of IT-Products is based on the whole development process and is checked during the evaluation and certification. The requirements used are based on standards, prescriptions, criteria and recommendations.

In this paper the term *Quality* denotes the compliance of *the measure of assurance of correctness and effectiveness of a target of evaluation (TOE)*.

With respect to security objectives of an IT-Product five steps have to be considered:

- the design phase
- the development process
- the evaluation
- the certification
- the requirements for operation.

These different steps linked together build the complete range in the life-cycle of a product. Each of these detailed steps is represented by several criteria (e.g. ITSEC [ITSEC], ITSEM [ITSEM], ISO 9000 set [ISO 9001/4], EN/DIN 45000 set [EN45011/2/3/11]).

Many manufactures have installed a quality assurance system, e.g. based on the requirements of the ISO 9000 set.

A number of different quality requirements defined in the specific IT-Security Criteria are met by the conditions of the installed quality system in a more general way.

Our intention for this paper was a first approach to combine the different quality assurance aspects in the criteria mentioned above with respect to the roles of the different involved parties (developer, evaluator and certifier).

2. Summary

The set of quality aspects may be distinguished in the following categories:

- a) the different involved partners (e.g. manufacturer, developer, evaluator, certifier)
- b) the relevant general criteria (Fundamental Basic Criteria) for establishing and maintenance of the different responsibilities of the involved partners (e.g. the set of ISO 9000, EN/DIN 45001, EN/DIN 45003, EN/DIN 45011)
- c) the evaluation criteria (IT-Security Specific Criteria) which have to be used (e.g. ITSEC, ITSEM).

A first approach of global consistence can be demonstrated by several detail relations between the Fundamental Criteria and the IT-Security Specific Criteria (here: ITSEC).

As a result this first check has demonstrated the problems of different used terminology and different interpretations.

The general existence of a quality assurance system and its usage does not replace the need of a detailed product evaluation and certification in the field of IT-Security. But a well installed quality system may reduce the requirements for the evaluation and certification phase (time and efforts).

Assumption is a first approach concerning common terminology and an enhancement of the Fundamental Criteria for the behalf of IT-Security.

In the field of accreditation of laboratories (ITSEFs) and certification bodies (CBs) (EN/DIN 45001/11) these requirements are solved in a first approach.

The relevant Fundamental Criteria are improved by IT-Security specific interpretations which build the foundation for each evaluation and certification.

The specific IT-Security Criteria may be interpreted as a specific enhancement of the general quality criteria (IT-Security as a special application of quality assurance).

3. Relevant Criteria, Prescriptions and their application

The first European harmonised IT-Criteria (ITSEC) and the relevant Evaluation Manual ITSEM are building the heart of the IT-Security requirements, improved by a framework (including the set of the ISO 9000, the set of the EN/DIN 45000 or ISO guide 25 [ISO 25]).

A set of detailed descriptions and in case of the ITSEC a lot of detailed prescriptions are defined. Major parts of these criteria are aspects in the field of

attributes of the product, its development process, its documentation and its evaluation / certification itself.

In this presented scheme of different responsibilities and competences the involved partners have to fulfil different aspects and roles.

According to their different roles GISA has to cover several roles: as the *Accreditation & Licencing Body* (ALB) and as the *Certification Body* (CB).

The role as the Accreditation & Licencing Body is embedded in the requirements of EN/DIN 45003.

The quality of the Certification Body is based on the use of EN/DIN 45011. As considered in the EN/DIN 45003 GISA has to meet all the conditions concerning the structure, organization and personnel required in the EN/DIN 45011.

Seperated divisions have to ensure the impartiality and responsibility of each role in the scheme.

In Germany the evaluations are performed by so called *IT-Security-Evaluation-Facilities* (ITSEF). The "Accreditation & Licencing" of these independent laboratories are performed by GISA. Basic rules are defined in the EN/DIN 45001.

The whole process of "Accreditation & Licencing" has to be separated into two different phases. The first phase called "Accreditation" is improved by additional IT-Security specific enhancements and interpretations. The second phase of the "Accreditation and Licencing" scheme is defined by the technical phase, the socalled "Licencing".

During the first phase many quality assurance aspects are considered. The "Leagal Identity", its "Impartiality", its "Independance" and its "Integrity" are more general requirements.

Requirements concerning "Technical Competence" include quality aspects comparable to the requirements in the ITSEC and ITSEM as "Personnel roles" or "sufficient qualified personnel", which shall have the "necessary education, training, technical knowledge and experience in evaluation in IT-Security or comparable fields. The personal shall be bound to observe professional secrecy with regard to all information gained in carrying out its tasks (personals confidentiality).

The needed "Test environment" has to be adequate and available. The "Premises", the "Surrounding and the Equipment" have to fulfil the requirements in the sense of "Reproducability and Repeatability". It has to be assured that the evaluation results are independent from outside influence.

Main quality aspects have to be met by the conditions of the "Working Procedures". The used "Test Methods and Procedures" are defined by the ITSEC and further relevant Security Criteria. Furthermore the testing laboratory shall have "adequate documented instructions on the use and operation of all relevant equipment, on the handling and preparation of test items according to the relevant criteria" (e.g. ITSEM and special additions defined by the CB).

The laboratory shall operate a systemmatically and periodically reviewed "Quality System" appropriate to the type, range and volume of work performed. The elements of this system shall be documented in a Quality Manual which is available for use by the personnel.

The Quality Manual should be based on the requirements of ISO 9004, at least it shall contain :

- a quality policy statement
- the structure of the ITSEF (e.g. organizational chart)
- the operational / functional activities pertaining to quality (concerning the limits and extents of each responsibility)
- general quality assurance procedures
- reference to quality assurance procedures specific for each evaluation
- where appropriate, reference to proficiency in testing, use of reference material, etc.
- procedures for dealing with complaints.

"Test reports" shall present each work in an accurate, clear and unambiguous way, according to the requirements of the ITSEM added by specific instructions of the Certification Body.

The ITSEF shall maintain a "Record System" to suit the particular circumstances in IT-Security Evaluation. Considering the need for repetition and reproducibility of the evaluation each record shall contain sufficient information.

Subcontracts are only allowed with ITSEFs licenced by GISA.

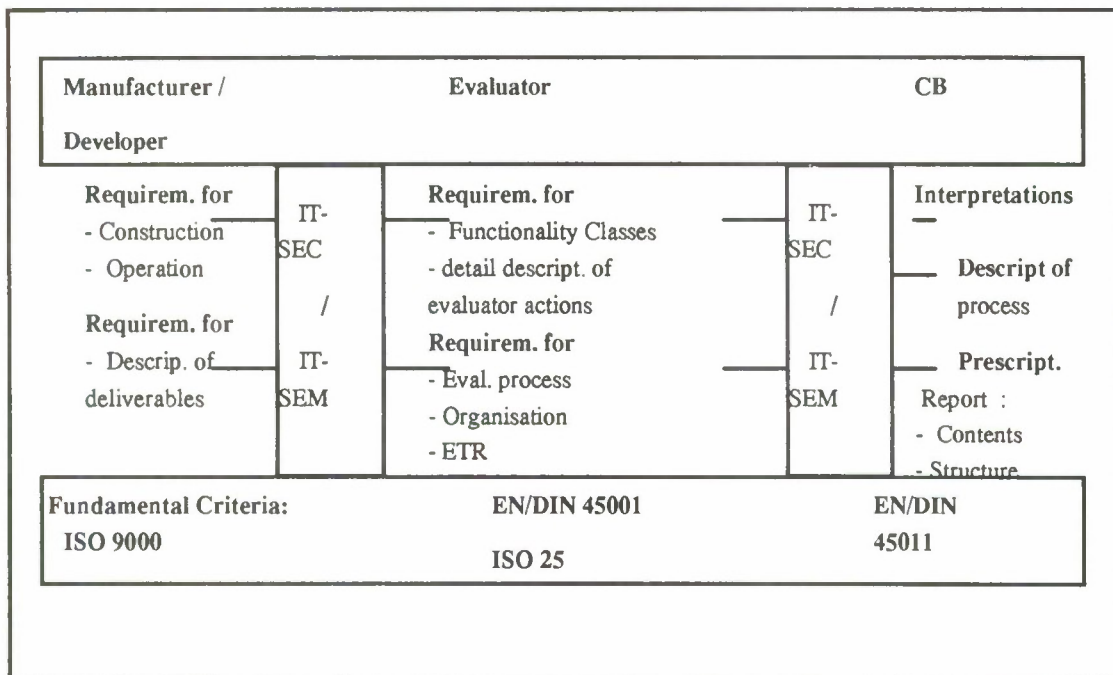
These formal aspects are improved by further additions and interpretations on specific IT-Security relevant aspects in the EN/DIN 45001.

During the "Licencing" the technical competence of the ITSEF is checked. The ITSEF has to perform an IT-Security test evaluation based on assesement modules. The ITSEF will get a TOE, prepared by GISA and available in a GISA database. The whole set of all relevant informations (e.g. the claimed deliverables from the sponsor, e.g. the documentation, parts of the source code, etc.) is given to the ITSEF to perform an evaluation. The ITSEF has to check the deliverables e.g. according to correctness, completeness and consistence.

The ITSEF will demonstrate his competence and will be trained in the behaviour of performing an evaluation based on the german scheme. This training includes the knowledge of the different roles which have to be met in the future.

4. A first common approach

These pictures (1)-(3) contain examples of the relationship between the partners and the specific IT-Security Criteria.



Picture (1)

Main Moduls during the SW-Development are:

- System Requirement Spezification
- System Design
- SW-Requirement Spezification
- SW-Design
- Coding
- Operating

Each step has to be considered in respect to the quality requirements.

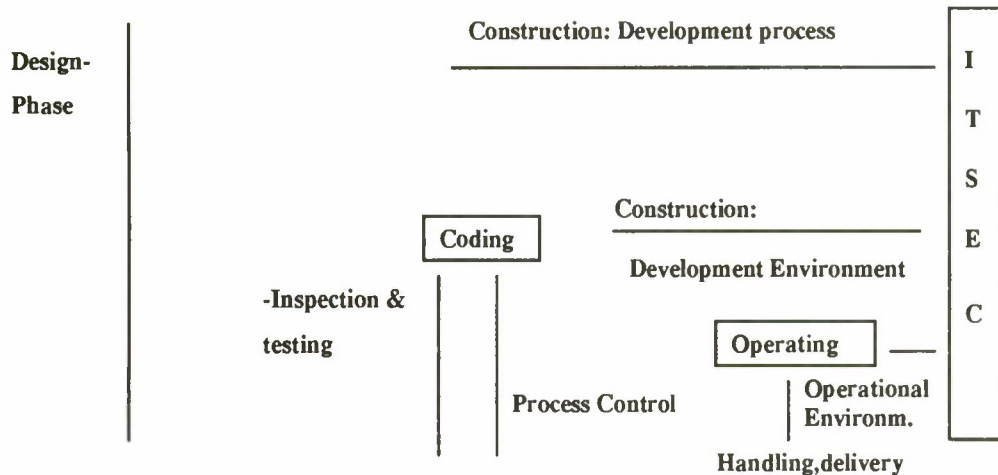
At the examples of

- System Design
- SW-Design
- Coding
- Operating

a first global mapping of the relationship between the Fundamental Criteria ISO 9001 and a specific IT-Security Criteria (ITSEC) is shown (picture (2)).

Details descriptions will be improved by (picture (3.1)-(3.5)).

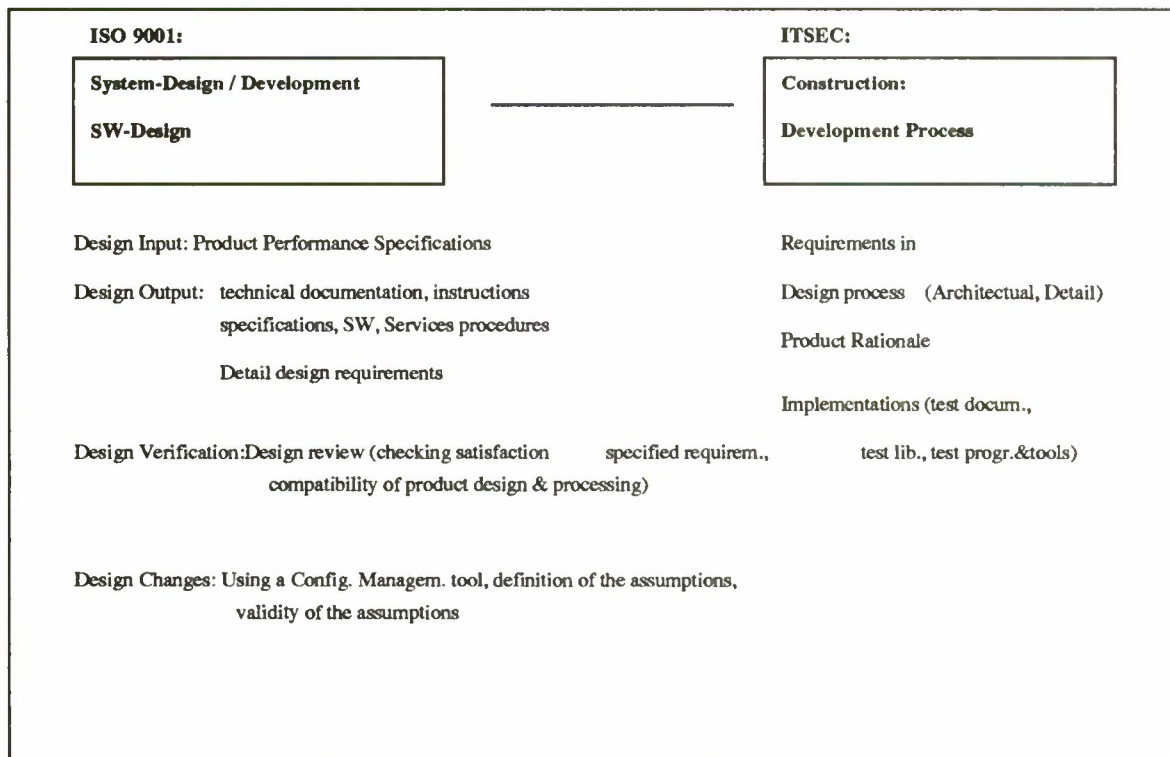
**System-Design
System-Requirements**



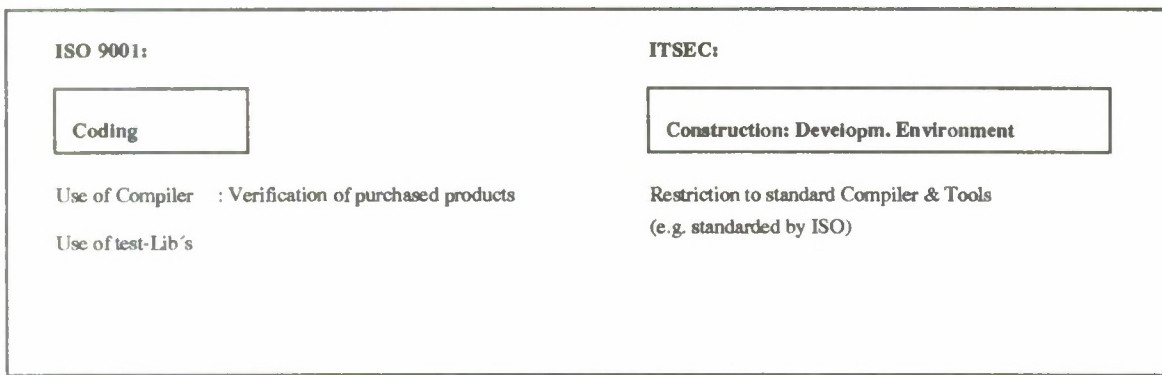
ISO 9001: Quality systems - a model for quality assurance in design / development, production, installation & servicing

Picture (2)

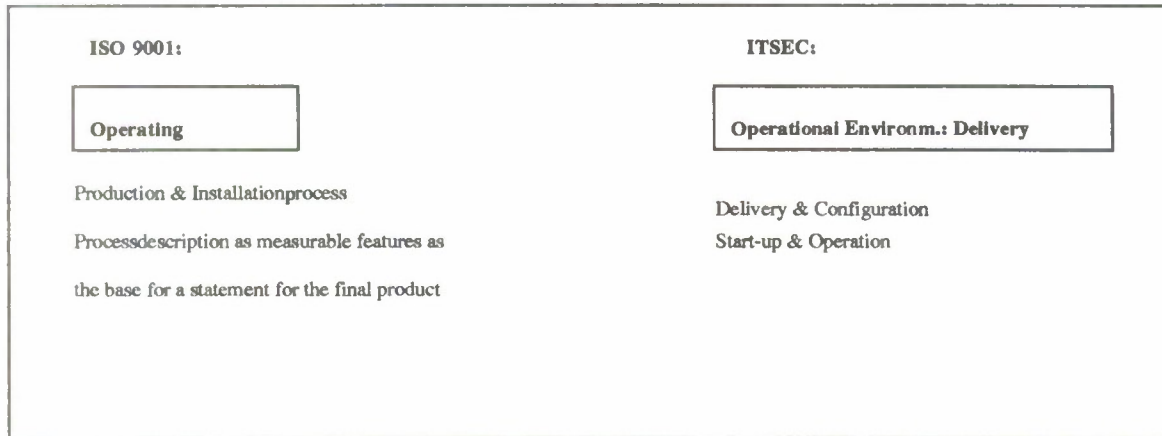
The Pictures (3.1)-(3.5) are a combination of picture (1) and picture (2). They show in detail several relations between the IT-Security Specific Criteria and the requirements based on the Fundamental Criteria, here ISO 9001.



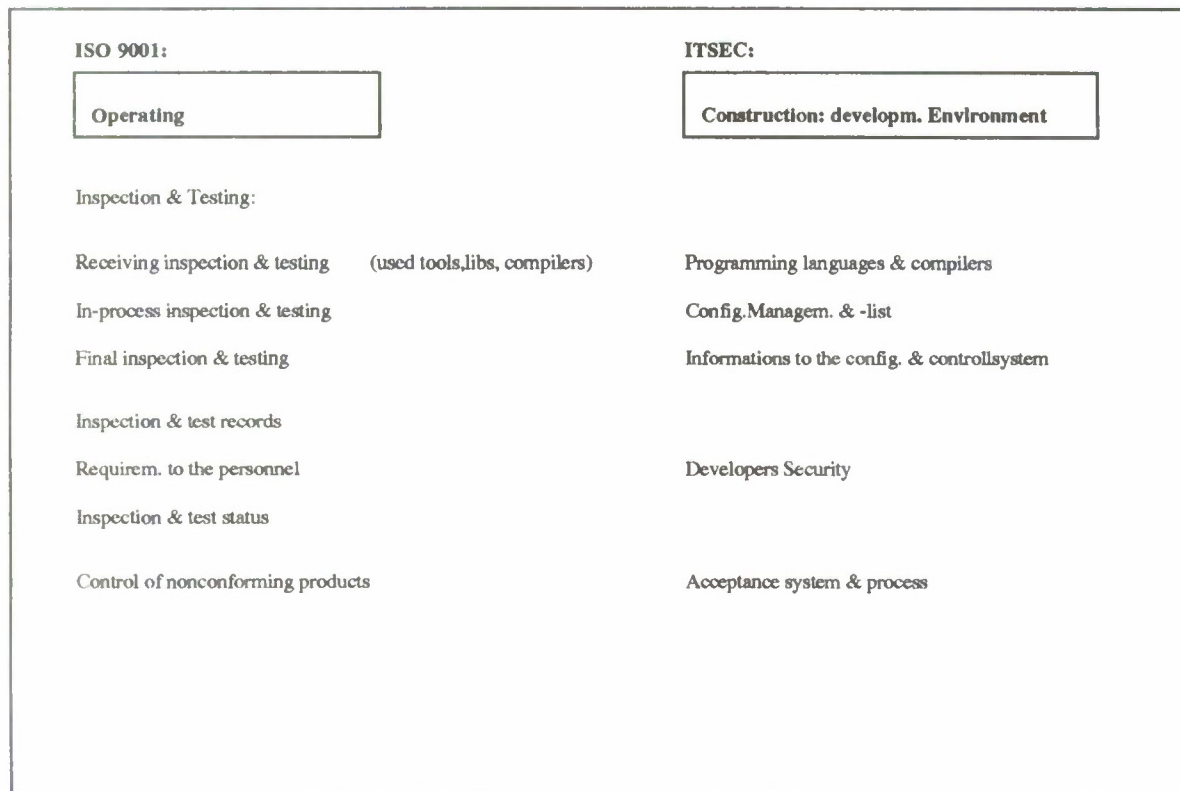
Picture (3.1)



Picture (3.2)



Picture (3.3)



Picture (3.4)



Picture (3.5)

5. An example: "the V-Model"

GISA has performed first steps to construct an IT-Security tailored quality assurance model called "V-Model" (process-model).

The general V-Model includes 4 submodules:

- Quality Assurance (QA)
- Project Management (PM)
- Software Development (SWD)
- Configuration Management (CM).

Depending on the chosen level of correctness (E-level in the ITSEC) different requirements are to be met, e.g. according to the rules of the PM (e.g. the organization, its activities) or to the requirements of SWD.

New rules have to be defined according the relevant IT-Security specific criteria and conditions.

Additionally it has to be distinguished between "evaluation after completion" (evaluation on available products) in comparison to "current evaluation" (i.e. evaluation parallel to development).

In the case of the "evaluation after completion" (case (1)) the conditions of the V-Model and the requirements of the IT-Security Specific Criteria (ITSEC /ITSEM) have to be met. In this case the tailoring is determined by the conditions based on the ITSEC. This tailoring would depend on the detail requirements in the ITSEC e.g. according to the target of the evaluation level.

Case (2) "current evaluation" is much more difficult. The current evaluation is defined by a lot of different interactions and relationships. An independent evaluation & certification block - maybe a submodel of QA- shall meet the possibility and permission for interaction with the other blocks.

References:

- | | |
|-----------|---|
| [ISO 9/0] | ISO 9000: Quality management and quality assurance standards |
| [ISO 9/1] | ISO 9001: Quality systems - Model for quality assurance in design / development, production, installation and servicing |
| [ISO 9/4] | ISO 9004: Quality management and quality systems elements - Guidelines |

[EN45/1]	EN/DIN 45001: General criteria for the operation of testing laboratories, 1989
[EN45/2]	EN/DIN 45002: General criteria for the assessment of testing laboratories, 1989
[EN45/3]	EN/DIN 45003: General criteria for laboratory accreditation bodies, 1989
[EN45/11]	EN/DIN 45011: General criteria for certification bodies operating product certification, 1989
[ISO 25]	ISO Guide 25: General requirements for the competence of calibration and testing laboratories
[V-MOD]	Anwendung des Vorgehensmodells unter Berücksichtigung anderer oder ergänzender Vorschriften, (Dr. Pütz, H. Hauff, BSI)
[ITSEC]	ITSEC: Information Technology Security Evaluation Criteria, Version 1.2, 1991
[ITSEM]	ITSEM: Information Technology Security Evaluation Manual, Draft-Version 0.2, 1992

SEVEN STRATEGIES FOR INFORMATION TECHNOLOGY PROTECTION IN THE 1990s

Thomas R. Malarkey
Department of Defense
P.O. Box 4388
Mountain View, CA 94040

ABSTRACT

The paper attempts to lay out several useful strategies for improving the U.S. posture in information technology (IT) security. The paper is divided into three main sections. The first briefly reviews the successes and failures of IT protection in the 1980s. The second provides a broad brush treatment of IT trends in an attempt to describe, at least at the conceptual level, the IT infrastructure that will need to be protected in the next decade. The last section describes seven suggested directions for government and industry attention. These include: establishing a national IT protection policy; promulgating a national policy on minimum IT protection requirements; unifying the security and safety constituencies into a single protection community; increasing emphasis on effective system security management; examining alternatives to the traditional in-depth evaluation process for IT security product quality control; improving the accountability features of IT products; and increasing our investment in security interoperability.

Keywords: Information protection policy, information technology security, development assurance, accountability, minimum protection requirements, system security management, interoperability of security features.

INTRODUCTION

There is a natural apprehension in writing a paper that proposes broad "strategies" for the future. For one thing, successful strategies ultimately need to be executed by real people and organizations. This paper is intentionally vague on implementation details, deferring the decisions on resource planning and institutional structures for subsequent efforts. Moreover, such a paper necessarily requires an attempt to predict what the technological (and to some extent, the political and sociological) future has in store. The massive technological shifts of the last decade are testimony to the need for caution and humility as we attempt to foresee what will transpire in the next. When we try to factor in social and political trends, the job becomes even more challenging. There may have been some prognosticators who could have predicted the PC and LAN revolutions ten years ago, but who would have been bold enough to predict the reunification of Germany, the breakup of the USSR, and the end of the cold war? Lastly, there has been a noticeable lull in major computer security incidents in the last few years. Is there really a burning need for new initiatives in IT protection? Are we prepared to either identify new investment resources or to reallocate existing resources in order to move out in the suggested directions? The answers to all of these questions are far from certain.

These considerations notwithstanding, there are good arguments for some reflection on potential strategic directions for the next decade. The pace of technological change that we saw in the 1980s is likely to continue at least at the same rate, if not faster. As our experience with nuclear technology illustrates, humankind has been very successful at developing new technologies, but much less successful in controlling and managing these technologies to meet the needs of the people of the world. Similarly, the challenges of protecting information systems will continue to exceed our ability to develop effective responses. Nevertheless, by the turn of the millennium, our society will likely be so dependent on information, and consequently on information technology, that unless we are willing to take the necessary political, social, and technological steps to protect IT, we will be in real danger of losing control of the very systems put in place to serve us.

This paper is organized in three sections. First, we review what the information technology (IT) security community has accomplished in the 1980s. Next, we provide a very broad brush treatment of IT trends in an attempt to describe, at least at the conceptual level, the IT infrastructure that will need to be protected in the next decade. In the final section, we describe some suggested directions for government and industry attention.

None of the ideas presented are particularly new. Most have been discussed in the community for some time, and many were explored in considerably more depth in the National Research Council (NRC) study that culminated in the publication of *Computers at Risk* [16]. Moreover, the seven strategies outlined here are in no way intended as "the" strategies for the 1990s. This paper is notably silent on many topics that will be very important in advancing our ability to protect information systems. Such critical topics as international IT security criteria and evaluation and the application of cryptography as an IT protection mechanism are virtually ignored. It is not that these topics are unimportant; it is just that what we have tried to do is choose some relatively neglected areas where small investments can potentially lead to large payoffs. In combination with other inputs, the seven strategies presented here can form the basis for something the U.S.* badly needs--an overall national IT protection strategy for the 1990s.

The paper is less an action plan than a generalized roadmap of what the author believes to be useful activities that can help us better protect information and the systems on which it is processed. Developing detailed strategies, plans, investment programs, and institutional arrangements will have to be a follow-on activity performed as a true government-industry partnership.

THE WAY WE WERE: IT SECURITY IN THE 1980s

We sometimes lose sight of the fact that IT security is still a very young discipline. The Anderson Report [1], more or less computer security's "Book of Genesis", was published only a little over twenty years ago; the DoD Computer Security Evaluation Center and its successor have been with us for a little more than

* The focus on U.S. activities throughout this paper should not be construed as reflecting a nationalistic preoccupation. Obviously, many aspects of IT security can only be dealt with in an international context. Addressing them from a national, perspective first, however, should may make getting started somewhat easier.

a decade. The early computer security activities were, of course, well rooted in the DoD and, especially, its intelligence community components. The information model that has driven most of the developments in computer security has been the model used for handling classified information in the paper world as embodied in Executive Order 12356 [9] and supporting procedures. While this information model and the mechanisms developed to implement it in automated information systems still form the basis for much of the work in the field, there was an increasing awareness in the latter part of the 1980s that new information models and technical solutions were needed to deal with the security problems confronting other parts of the government and the private sector. Clark and Wilson's seminal paper [4] was a major stride forward in identifying this need and in proposing a model for addressing it. Still, we have been more successful in stating the need to go beyond the Bell and LaPadula model [3] than in developing any practical prototypes that illustrate how it can be done. Our collective inability to even agree on a satisfactory definition for "integrity" is testimony to the fact that there is still a long way to go.

Not only does computer security have its roots in the DoD, it also owes some of its genealogy to the much more mature field of communications security (COMSEC). The COMSEC roots account, in part at least, for perhaps the most frequently and hotly discussed topic in the field during the 1980s--the emphasis placed on independent evaluation as a source of assurance[†]. It's no exaggeration to say that the evaluation process and its results were the principal preoccupations of many professionals in the computer security community during the last decade. The DoD Computer Security Evaluation Center, later to become the National Computer Security Center (NCSC), was established in 1981, as its name implies, primarily to evaluate products and publish the results in an Evaluated Products List (EPL). The *DoD Trusted Computer Systems Evaluation Criteria* (TCSEC) [13] and its international offspring were published not primarily as ends in themselves, but to form the basis for national evaluation programs. The success, or lack of success, of the NCSC evaluation program has probably been the most dominant discussion item on the computer security agenda. Evaluations are universally decried as taking too long and costing too much. The value of product evaluations to customers is being called into question. Nevertheless, vendors are still submitting products to be evaluated, and entries are still being added to the EPL.

We'll talk further about the role of evaluation later in the paper. But what can we say about the results of the evaluation program to date? Certainly, if the people who established the Center had been asked twelve years ago to predict how many and what types of products would be on the EPL in 1992, their projections would have been considerably higher than the actual results. Does this mean that the evaluation program has been a failure? Not necessarily. The number of products on the EPL is only one measure of success, and an imperfect one at that. The real aim of the evaluation program was not to perform evaluations, but to increase the availability of secure IT products; evaluation is only a means to that end. The fact is that commercial-off-the-shelf (COTS) IT products have considerably better security features and assurance built into them in 1992 than they did a decade ago. This

†. There were certainly other factors involved as well; the penetration activities of the early 1970s heavily influenced the thinking that led to the establishment of the DoD Computer Security Evaluation Center. Moreover, these penetration exercises led to a different evaluation paradigm for COMPUSEC than for COMSEC, the former emphasizing design analysis and the latter a search for exploitable vulnerabilities.

achievement is due in no small measure to the existence of the Center's evaluation program supported by government policy, i.e., NTISSP 200, often referred to as "C2 by '92". Whether or not the product evaluation process will be as useful in the 1990s, particularly in the commercial world, is a subject of debate, but it is an indisputable fact that the NCSC program has been a major factor in raising the general level of security in COTS products, and in heightening the security awareness of customers nationwide.

While the 1980s saw significant improvements in commercial product security, considerably less progress was made with regard to security of operational IT systems. In part, this is due to the way the notion of "systems" has evolved over the last ten years. In the early 1980s, there was a much closer relationship between a product and a system. The authors of the TCSEC, while recognizing the need for a separate system certification and accreditation (C&A) process following product evaluation, appear to have believed that evaluation would make C&A almost automatic. The argument went that the intensive technical work would only have to be performed once by a small team of highly-trained technical experts. The results could then be reused many times by the less-technically capable C&A teams. In the early part of the decade there was a sound basis for this argument. The certification of DOCKMASTER, for example, could be based almost exclusively on the B2 EPL rating for Honeywell's Multics system because DOCKMASTER and Multics were almost the same thing. In this case, little if any additional work was required to complete C&A. Later on, as systems began increasingly to be built from different products (LANs, workstations, servers), it became much more difficult to use the EPL rating as the sole basis for a C&A decision. In fact, even if the system were built solely from trusted products, there would still be a strong likelihood that the products wouldn't work well together, necessitating breakage of the individual TCBs, and making the certifiers look much more deeply at the resulting heterogeneous system.

How then can we summarize computer security progress in the 1980s? First, from a security perspective, we now have much better IT products than we had twelve years ago, but our ability to develop closed-form system security solutions is still woefully inadequate. The C&A of operational systems is still a mix of hard work, educated guesses, and prayer; much more an art than a science. Product evaluation results have not yet proven to be as valuable to C&A as was expected. Second, the need to develop information models and solutions that go beyond the national security paradigm is well recognized, but we have not yet come to grips nationally with the methodology or technology to do so.

WHERE WE ARE GOING: THE INFORMATION INFRASTRUCTURE OF THE NEXT DECADE

In the 1980s our IT infrastructure changed from a mainframe-centered, largely disconnected collection of systems to a client-server, LAN-based architecture where large segments are loosely or tightly coupled into distributed computing environments. The PC/workstation/LAN revolution has been the subject of much discussion. Even more dramatic, however, has been the almost universal acceptance, at least in the U.S., of the DoD suite of network protocols leading to an unparalleled growth in system connectedness. In 1980, the only significant computer network, the ARPANET, was made up of fewer than 100 host systems, largely in the research community. By the end of the decade, the successor to the ARPANET--the Internet--

served a literally uncountable number of subscribers from government, industry, academia, and other sectors. The almost organic nature of the Internet is exemplified by the fact that it's impossible to point to any single organization responsible for its management. It is especially ironic that, while the so-called Internet Worm of 1988, was denying service to large numbers of Internet subscribers, the network itself continued to perform admirably.

The trend toward almost universal connectedness over large data highways will certainly continue in the 1990s. For one thing, technology and economics will continue to push us in that direction. The convergence of technological advances in computing and communications coupled with the pressures to reduce costs in both public and private sectors will lead inevitably to the establishment of data network utilities patterned after the Internet. Political factors, exemplified by the strong technology thrusts of the new administration will further hasten the movement in this direction. One of the six broad initiatives proposed by the Clinton-Gore campaign [5] was to establish a "21st century technology infrastructure." It's fashionable these days to talk of a National Information Infrastructure (NII), in contrast to the view of a collection of individual, independent systems. By the beginning of the next century, as business becomes even more multi-national, it will probably be more realistic to speak about an International Information Infrastructure (I³).

Hanging off this information infrastructure will be a wide variety of physical and virtual enterprise systems serving government, industry, non-profit organizations, educational institutions, etc., all requiring the ability to adequately protect valuable information being processed, stored, and transmitted on the NII. The NII, like the timesharing systems of the 1970s, will have to support efficient transmission and sharing of information in an environment where many of its subscribers may be economic or political competitors. The job of providing adequate protection to this complex structure and the information it stores, processes, or transmits will be shared by the data highway and the individual systems or domains. Determining how such security is provided (i.e., what services does the NII provide, what services are left to the individual enterprise systems, how will the protection services be developed, implemented, and managed) will be an enormous task requiring close cooperation among all the parties involved.

HOW TO GET THERE: PROTECTING THE NATIONAL INFORMATION INFRASTRUCTURE

As the NRC study and others have observed, protecting our information assets will be a formidable job, requiring close cooperation among government, the IT industry, and other communities. To do the job right will undoubtedly require considerable work on the part of all players both to build the consensus required to stimulate the necessary investment and to plan and execute the supporting activities. It is also clear that there is not yet a national agreement on what needs to be done and how much investment we are willing to apply. Computer security professionals find themselves in the unenviable position of trying to sell the need for IT security largely based on the belief that, without it, bad things will happen to the country's information infrastructure in the future. Given the relatively few "smoking guns" uncovered to date, this is and will continue to be a hard sell. The end of the cold war has produced a noticeable decline in enthusiasm for IT security in the national security community, traditionally one of its foremost advocates, as

departments and agencies scramble to downsize and allocate scarce dollars and people in the most cost-effective way. Still, there appears to be a slowly growing recognition, especially in certain parts of the private sector, that we need to do the job, and hope that a technologically-grounded administration will provide the necessary leadership and support.

This paper does not lay out any grand plan; nor does it try to define specific tasks or organizations to carry them out. Rather, the attempt is to take a small step in the right direction by proposing a few broad approaches (again, none of them necessarily new) that can be debated, refined, and eventually form the basis for a national effort. Investment costs associated with these strategies should be relatively small; there are no big-ticket items on the list. Still, some investment (both public and private) will be required. In cost-conscious times like these, this will likely entail diverting money from some existing activities. An investment program is a reflection of an underlying value system. Consequently, it will be necessary to convince ourselves that the strategies described here are more worthy of increased attention and investment than some of the more traditional approaches. Only time will tell whether or not we will be willing to make the necessary adjustments.

While it may sound redundant, the need for this effort to be a true partnership between the public and private sectors can't be overemphasized. Some of these initiatives can only be accomplished through strong government leadership; others are necessarily more in the realm of the IT manufacturers and other parts of the private sector. With a clear set of goals and objectives and a spirit of cooperation it should be possible to make significant strides toward better protecting our information infrastructure.

1. Establish a national information and IT protection policy

In order to make significant progress in developing technical or procedural solutions to IT protection problems, the whole topic needs to be addressed in a much larger policy context, i.e., the United States needs a national policy on information and information technology protection. The closest thing we have to such a policy now is that which addresses the handling of classified national security information. Since the second world war, a relatively coherent set of statutes, executive orders, regulations, and procedures has been put in place to address the processing, storage, and handling of classified information. The definitions of the terms involved, proper handling procedures, personnel vetting procedures (i.e., the clearance process), and penalties for misuse have all been developed, promulgated, and generally accepted as necessary for protecting national security information in an era of global challenge[‡].

Given this general national agreement on the need to protect classified information and the methods for doing so, it should come as no surprise that it is precisely in this area where the greatest strides in automated information protection have been accomplished. The importance ascribed to classified information spawned the necessary investment that led to the penetration exercises of the 1970s, issuance of the Anderson Report, the DoD Computer Security Initiative, and the establishment of the NCSC. While the TCSEC's preoccupation with confidentiality

[‡]. There obviously are particular aspects of national security policy and procedures that have not been accepted by all citizens of the U.S. The emphasis here is on the word "generally."

has often been overstated, the fact remains that, especially at levels above B1, the Orange Book is focused on the goal of protecting national security information. The Bell and LaPadula model, the mathematical underpinning of mandatory access controls, is derived directly from the policies and procedures put in place to handle classified information in the paper world.

For more than forty years there has been an implicit national consensus that national security information is the nation's most valuable information asset, requiring the highest degree of protection. Events of the last several years, such as the end of the Cold War and the growing importance of geoeconomics relative to geopolitics, have called into question this fundamental assumption. In a post-cold war era is national defense information really more important to the nation's security than economic data? What kind of protective measures should be applied to personal information (medical records, for example)? Does proprietary information, formerly viewed as the province of individual enterprises, have importance to the country as a whole in an era of geoeconomic competition?

In order to address these and a host of related questions, a U.S. information protection policy that addresses all types of nationally-valuable information is sorely needed. Such a policy must articulate a national view that reflects an underlying consensus about the value of different types of information, just as the classification policy reflected a national consensus during the cold war era. This information protection policy should be developed in close concert with or as part of an overall national technology policy that is likely to form a major initiative of the Clinton-Gore administration. In establishing such a policy we can learn a great deal from our close neighbor to the north, the government of Canada. For several years, Canada has had policy and procedures for dealing with both classified information and what is called "designated" information [23]. The latter category deals with types of information that are roughly the equivalent of what the Computer Security Act of 1987 [6] considers "sensitive" information, including personal privacy and proprietary data. The Canadian government has established policy and procedures for dealing with designated information to include a set of personnel screening procedures less stringent than, but analogous to, the procedures for granting security clearances.

The Canadian policy and standards are directed at government institutions. A more useful national policy would have to be broad enough to address the protection of "nationally-important," i.e., classified or designated information, by private institutions as well. A person's medical records need to be protected by private physicians and hospitals as well as by the government. Building on the Canadian approach, however, it should be possible to construct a unified information protection policy encompassing both national security related information as well as the increasingly important economic and personal information. Such a policy can form an overall umbrella under which specific protection measures can be developed and implemented. It should encompass fundamental definitions, handling procedures, rules for information markings (both human and machine-readable), processes for vetting people who handle such information (if deemed appropriate), and penalties for unauthorized or improper use of nationally-valuable information. It should, moreover, address the statutory, regulatory, and enforcement structure necessary to carry it out. What, for example, is the legitimate role for law enforcement in the NII, i.e., how do we balance the legitimate needs of law enforcement agencies with the equally legitimate privacy rights of our citizenry. This issue has been hotly debated for several years, but recent dialogue in a variety of forums suggests that, while the issues will remain contentious, some middle ground may be possible. The policy need, not address every category of information in the country, only that defined to be

important to the country as a whole or to the rights and privileges of its citizens. Nevertheless, such a policy can serve as a model on which local information protection policies can be based.

National security information policy was developed in the context of the paper world, but proved somewhat less useful in handling automated information. A national INFOSEC policy for the next decade must go beyond the paper world and try to address the differences between a domain whose fundamental objects are paper documents and safes and one whose objects include EMAIL messages, relational database tuples, and executable computer programs. The development of this policy should also consider the information protection measures that should be provided to users of the NII. Just as users of the interstate highway system in the U.S. have learned to depend on a certain agreed-upon set of minimum features that all segments of the system provide, regardless in which states they are located, users of the information superhighway should be able to depend on a minimal set of protection services that the NII will provide to its users. By no means should the NII be expected to provide all, or even most, of the required protection services; some have to be performed in the individual enterprise islands connected to the superhighway. Nevertheless, a minimum set of confidentiality, integrity, and availability services can and should be provided. While the details of these protection services should not and need not be defined by the policy, the top level ideas certainly can be articulated so as to form the basis for the expectations that users of the NII will have with respect to protection of information assets.

Lastly, in developing this policy it will behoove us to reexamine the institutional structures in the public and private sectors that will be responsible for carrying out such a policy once it is put in place. For much of the 1980s, IT security in the federal government was the province of the NCSC. Since 1988, NIST has been charged with the major share of the responsibility for computer security outside the national security establishment, but has not received the resources required to execute this responsibility. Recognizing a number of problems with the existing institutional breakout, the NRC study proposed the establishment of a private organization called the Information Security Foundation (ISF). Whether the need for a new institution was never really understood, whether the concept stalled because of a lack of necessary seed money, whether it was felt that existing institutions were performing satisfactorily, or a combination of all three, the ISF really never got off the ground. It's not clear what the correct private-public sector institutional mix should be, but the issue should certainly be addressed as part of the policy-making agenda. A variety of options should be explored, including creation of a cooperative consortium along the lines of Sematech or the Microelectronics Computer Technology Corporation (MCC). Whatever institutional structure is finally agreed to, it must be adequate to put in place the activities necessary to ensure that the IT protection policy is faithfully carried out.

2. Promulgate a national policy on baseline protection requirements for IT products and systems

It has been common practice to make jokes about the controlled access protection policy promulgated in National Telecommunications and Information Systems Policy (NTISSP) 200 [17], known more commonly as "C2 by '92." It is true that the target year has passed, and not all systems in the federal government meet the letter of the C2 requirements. What such humor overlooks, however, is that this policy, perhaps more than any other action, was instrumental in raising the level of

security built into commercial IT products. NTISSP 200 established a market for IT security, and caused vendors to develop C2 products and submit them to the NCSC for evaluation. Of course, not all operational versions of rated products have been through the Center's ratings maintenance (RAMP) process; certainly, system managers sometimes, knowingly or unknowingly, turn off important security mechanisms. Still, I would assert that our systems are fundamentally better today than they were ten years ago, and that NTISSP 200 was a major contributor to this improvement.

NTISSP 200 was, in effect, shelved by the passage of the Computer Security Act of 1987, other than for national security systems. In 1990, NIST published non-mandatory guidance to Federal Agencies on the use of trusted systems [14]. That guidance notwithstanding, since 1988 we have had a practical policy vacuum in the area of baseline protection requirements for IT security; it's now time to raise the floor again. The first draft of the Federal Criteria (FC) [15] described two broad families of protection profiles. One of these, known as the national security family, consists of the traditional TCSEC classes B1-A1, renamed LP1-LP4 to emphasize that they all provide some degree of label-based protection. The other, called the commercial security family, consists of the TCSEC class C2 (renamed CS1) plus two new profiles, CS2 and CS3 that are extensions of C2. The commercial security profiles are intended to be applicable to large segments of the federal government as well as the private sector. While the individual requirements defined in these two new profiles will necessarily undergo some revision as the FC goes through public review, they will eventually become stable enough to use as guideposts for new national guidance. CS2, based on C2 but with significantly stronger accountability requirements (I&A, audit, system entry, etc.) ought to be the target for the middle of the decade (sorry, no catchy rhyming phrase). CS3 ought to be the target for the end. The national security community should examine its requirements as well, and set a goal of achieving a level of protection that can be provided by an improved B2 level (LP2 in the Federal Criteria).

These goals are reasonable and attainable; they may, in fact, be somewhat timid[§]. Even if we don't fully realize them, however, the mere fact that we set them as goals will help improve IT protection by stimulating consumer demand, helping to point producers in the right direction, and focusing our limited evaluation resources on IT products that address our important national requirements.

3. Promote increased cooperation between the security and safety communities

In this paper we have used the terms "protection" and "security" almost interchangeably, but we have used the former more often largely because it is more all-encompassing. *Computers at Risk* noted the similarity between requirements for security of information systems and those for reliability of safety-critical systems. In the United Kingdom, this similarity has been reflected in the publication of standards that could apply equally well to security or safety controls. In fact, many (but not all) of the requirements for securing information are nearly identical with those for any high-assurance endeavor. The requirement for stringent development practices, some type of product quality-assurance process (evaluation), and the need

§. The author has been told by several knowledgeable experts, for example, that systems with all the attributes of CS3 are available today.

for effective system management can just as easily characterize an application with demanding security requirements as one that requires a high degree of system safety. The term "Protection Profile" was adopted by the Federal Criteria as its fundamental construct for a variety of reasons, not the least of which was to avoid confusion with terms such as "security target" and "security profile." Nevertheless, the use of the more general term "protection" rather than "security" may make it easier to build bridges linking the security community with other activities that demand high assurance. The assurance requirements of the FC would not be out of place in a criteria specifying requirements for safety-critical systems.

Despite these similarities, the security and safety communities have not been as close as they should be. A major goal of IT protection policy in the 1990s ought to be to bring the multiple high assurance communities, including IT security, under a single umbrella. If nothing else, such a strategy will provide strength through numbers. Moreover, both communities can benefit from solutions that can be applied in a variety of applications. The security community's work on confidentiality and data and system integrity can have direct applicability in other high-assurance environments; reliability engineering results from the safety community might help security practitioners get a better handle on requirements and solutions for the relatively unexplored area of information availability. Whether or not we evolve into a single cohesive protection community is beside the point. Through joint workshops, conferences, research projects, and system developments both communities will be better able to deal with problems common to a variety of information processing applications.

4. Place additional emphasis on the importance of effective system security management

Some of the worst security "horror stories" of the last several years deal with situations where trusted products with adequate security controls were installed in an operational system, but rendered ineffective when some or all of the controls were disabled by the system administrator or site security officer. In 1989, a report from the President's Council on Integrity and Efficiency [19] noted that many government installations running large IBM mainframes had installed the C2-rated security package ACF2. When the operational configurations were examined, however, it was discovered that the ACF2 security controls were often ignored, bypassed, or misused. The result was a situation where a false sense of security engendered by the mere presence of a C2 product coupled with poor system management produced a condition that may have actually been worse than would have existed had the C2 product never been installed. Garfinckel and Spafford [10] cite example after example of where even the most sensible security safeguards can be rendered ineffective by inadequate attention to system security management and administration.

While the 1980s saw some important advances in security technology, there was considerably less improvement in overall system security management. The work of the Computer Emergency Response Team (CERT) and its clones have heightened the emphasis placed on good security management tools and practices. Nevertheless, the sense remains that sound security management is still a relatively neglected area; we have failed to put in the required work or resources to raise its importance.

Part of the problem can be addressed by a concentrated effort to encourage vendors to use security-supporting default options for their products. There are specific requirements in the Federal Criteria that address this need. Secondly, we can continue to develop tools to help the system security manager carry out his or her duties. A number of such tools are already available, although almost all are limited to the UNIX environment, and many require a good deal of in-depth systems experience to use properly.

If we limit our attention to ease-of-safe-use and improved management tools, however, we will be treating merely the symptoms rather than the problem itself. The fundamental issue here is that, with the exception of a few isolated instances, neither government nor industry has been willing to undertake the investment necessary to raise the overall quality of the people doing system security management. What is really needed is a concentrated program to increase the rewards (both monetary and psychic) that security managers can receive. There has to be increased prestige and authority associated with the work if we want to attract motivated and talented people to the field. Rather than being merely a component part of a computer operations organization, the security manager needs to be recognized as perhaps the most important player in protecting an enterprise's information assets. The educational and experience requirements need to be considerably expanded from what they are today, and the responsibilities of the position redefined in much broader terms. The irony is that we have been willing to invest fairly heavily in security technology in the last decade, while, at the same time, been reluctant to place similar resources into improving the prestige and skills of the people who manage this technology. A talented, experienced security manager can often work around limitations in security technology. On the other hand, all the AI systems in the world can't protect information if the managers don't do their jobs properly.

5. Develop alternatives to in-depth product evaluation as a source of product quality assurance

As noted earlier, the process of IT product evaluation dominated the agenda of computer security during the 1980s. The notion of an independent review that could serve as a *Consumer Reports* for the community was naturally attractive to consumers. Moreover, particularly after the issuance of NTISSP 200, manufacturers searching for product discriminators seized upon the EPL rating as an important mechanism for making their particular products attractive to potential markets. The only alternative to evaluation seemed to be what has been called "security by emphatic assertion," whereby a vendor would declare in glossy marketing brochures that its product had attained or was "designed to meet" some security level as defined in the TCSEC. This alternative has been generally viewed by both producers and consumers as wholly unsatisfactory.

Despite the potential attractiveness of an independent evaluation program, and despite hard work on the part of the evaluation community to improve the process of performing IT security evaluations, it is fair to say that the evaluation program has not been as successful as it was expected or needs to be as the principal quality assurance mechanism for IT product security. By a conservative estimate, the government has spent well over \$50 million to pay the salaries of government and contractor employees to perform evaluations since the mid-1980s. Vendor costs are harder to assess, although one vendor has claimed privately to have spent over \$25 million to develop a product and get it through the evaluation process. Certainly, attempts have been made to streamline the process, the most ambitious being the

NCSC-sponsored Process Action Team that applied principles of Total Quality Management (TQM) to the problem. Alternatives to government evaluation, notably the European programs that license commercial facilities to perform evaluations, have been attempted as a way to make the process more responsive to customer needs. The question as to whether vendors will be willing to pay relatively high costs for such commercial evaluations is still unanswered, as is the question of whether commercial evaluations can be profitable enough to generate sufficient interest by potential evaluation facilities. The nagging question persists, "is in-depth, independent evaluation a cost-effective method of performing quality control on commercial IT security products, especially for what are typically called low-assurance systems?" The corollary is, "If not, what can we do about it?"

One of the fundamental assumptions underlying the evaluation process is that an IT product can be sufficiently understood and analyzed by a small independent team in a reasonable period of time so as to add value for the consumers of the product. Given the size and complexity of most modern IT products, this assumption is questionable at best. An implicit premise on which the original evaluation program rested was that, as vendors began to build high assurance products, the security-relevant parts of the system, i.e., the TCB, would become closer to those of the idealized reference monitor. Unfortunately, the experiences of the last ten years belie this assumption. As manufacturers have added complex networking subsystems and graphic user interfaces to their products in response to customer demand, the ability of the developer's own staff to fully understand the product, let alone an independent team lacking product-specific knowledge, has been seriously curtailed. The value-added contribution provided by evaluation of such complex products is, therefore, difficult to assess.

Despite the uncertainty over the contribution made by evaluation, it seems that, particularly for products that promise high assurance, some type of independent quality assurance process will continue to be required. Even for low-assurance products some level of independent assessment may be necessary, even if it is a relatively cursory security testing of the product coupled with a review of vendor-developed evidence. We must avoid relying on "emphatic assertion" as the only evidence a consumer can use in selecting a product. What we really want to achieve is a situation where the depth of the evaluation is commensurate with the assurance required.

In an intriguing 1992 paper, Dorothy Denning [8] proposed a redefinition of the term "trust." As traditionally defined, trust has been considered an inherent property of a product or system, susceptible to being formally modeled, specified, and verified objectively by an independent evaluator. In Denning's view, trust is an assessment formed in a particular environment (market) based on personal experiences of the customers and the assessments of others who are trusted to be competent. These others could include manufacturers as well as independent evaluators. Since trust, defined in this way, is a subjective and dynamic concept, a variety of development and evaluation methodologies can be used to construct and assess a trusted system. While the utility and practicality of Denning's trusted systems paradigm have not yet been fully explored, some of the ideas she presented, e.g., the use of market-based criteria and standard security benchmarks, could be instrumental in producing an assessment process that is more adaptable to different customer sets.

There are several concepts introduced in the first version of the Federal Criteria that, when fully matured, may allow us to adopt a quality control process that relies

less on independent assessment and more on manufacturer-provided quality. Interestingly enough, the FC is the first set of IT security criteria not to use the word "evaluation" in its title. This doesn't mean it considers evaluation irrelevant, just that it at least leaves the door open for alternatives. The Federal Criteria divides assurance into two categories--development assurance and evaluation assurance. The assumption underlying this division is that a consumer can obtain confidence (assurance) about an IT product or system in several ways. First off, the mere inclusion of certain security-enforcing functions provides some sense of confidence that the product can enforce a security policy. More importantly, however, the consumer gains confidence by knowing how well the product or system was built (development assurance) and through some independent assessment (evaluation assurance). Different consumers will rely on varying mixes of these types of assurance. At present, many commercial products provide little in the way of development assurance. Consequently, the consumer is forced to rely on evaluation results in order to have any confidence that the security-enforcing functions perform as advertised. Assuming manufacturers improve their development practices, by adopting better system and software engineering procedures, the need for in-depth evaluation should be reduced, at least for some classes of products and consumers.

The FC, moreover, introduces a set of assurance components called flaw remediation requirements that require the developer to develop procedures for identifying and correcting flaws and disseminating corrections to consumers. The acceptance of this responsibility by the vendor should improve the consumer's confidence, that even if the product had been shipped with some undetected flaws (almost a tautological statement for most products at lower levels of trust, even those on the EPL), problems will be quickly addressed and corrected**. While correction of problems after the fact is never ideal, it may be satisfactory for many environments in both private and public sectors.

Lastly, the Federal Criteria borrows a concept, first articulated in the European Community's Information Technology Security Evaluation Criteria (ITSEC), of a "security target." As used in the FC, a security target is a product-specific description that elaborates on the general requirements specified in a protection profile, and provides detailed evidence, generated by the producer, of how a specific IT product meets the security requirements of a given protection profile. The security target can offer a way of reaching some middle ground between an in-depth evaluation on the one hand and emphatic assertion on the other. The security target is a vehicle whereby a manufacturer, perhaps working with evaluators or other security experts, can document how a particular product satisfies the requirements of a given protection profile. From a consumer's point of view, it can serve as a set of evidence presented in a standard form that justifies why an IT product deserves a particular rating. It puts the vendor's credibility behind a fairly detailed statement of compliance that could be reviewed by consumers. While not intended to fully replace an independent assessment, the security target, implemented properly, could offer a vast improvement over current practice. Coupled with improved development assurance and sound flaw remediation procedures, it can help us move in the direction of relying more on manufacturing quality and less on the need for a separate in-depth product evaluation.

** . The work of Carnegie-Mellon University's CERT has been quite instrumental in raising vendor consciousness to treat security problems more seriously than routine software bugs. Several manufacturers have even established special centers to address security problems that arise in their fielded products.

6. Increase the emphasis on implementation of effective accountability measures in IT products and systems

Discussion of access control in its various forms has tended to dominate the literature of computer security. Susceptible as it is to application of formal methods, it has always had a certain cachet compared with more mundane aspects of the field. On the other hand, the subject of accountability (defined in the Federal Criteria to include identification and authentication (I&A), system entry, trusted path, and audit) has been often treated like a poor relation. Audit, in particular, has never met with great favor in many parts of the security community, primarily since detection has always taken a back seat to prevention. While we acknowledge that the latter is always preferable to the former, the complexity of modern systems coupled with the practical difficulties associated with implementing "perfect" prevention mechanisms will require us, for the short term at least, to rely heavily on detection tools in managing our information infrastructure

Ironically, while our community was preoccupied with more esoteric matters like formal modeling and verification of MAC, many of the classic IT security incidents of the 1980s had little to do with access control violation *per se*, but a lot to do with exploitation of inadequate accountability safeguards [18]. The Morris worm of 1988 was successful largely because it either bypassed (SENDMAIL debug feature) or exploited (password dictionary attack) inadequate I&A capabilities [21]. The Wily Hacker incident also was launched by violations of I&A that led in turn to bypass of discretionary access controls. Moreover, this incident was eventually resolved by application of creative, home-grown auditing and intrusion detection mechanisms coupled with innovative trap-baiting techniques[22]. The message to us should be clear: while continuing to work on improved methods of access control, we really need to heighten the attention we pay to accountability in our IT infrastructure.

The Federal Criteria, in an attempt to confront this reality, does devote more attention to accountability than previous efforts. Specific accountability requirements occupy a significant portion of Volume I of the FC, and there is considerable guidance on I&A and audit in the individual protection profiles. Cost-effective security measures such as the use of token-based authentication and automated audit analysis and intrusion detection tools are described much more fully than in any previous criteria document.

Still, there is more that the community can do. The fact that most systems still rely on traditional passwords as the sole authentication mechanism when more secure token-based alternatives are available is an indictment of our level of commitment to protecting our systems. In part, fears of additional cost of such systems, both in terms of purchase price and administrative overhead, have inhibited widespread use. A more significant contributor, however, may simply be inertia. Near real-time intrusion detection systems have been developed as research projects, but as yet have achieved only limited operational use.

There are enough practical examples to offer hope. The adoption of a token-based authentication scheme on DOCKMASTER has significantly reduced the threat of break-ins. The Multics Intrusion Detection and Analysis System (MIDAS) [20], initially developed in the late 1980s and enhanced several times since, is an integral part of DOCKMASTER's security management operations. Work done by researchers at U.C. Davis [11] appears to constitute a promising start at dealing with

intrusion detection in a distributed system. NSA's Pre-Message Security Protocol (PMSP) initiative is doing promising work in the area of token-based authentication. What we need are more practical worked examples of I&A and intrusion detection in operational distributed computing environments. Since such systems will only achieve maximum utility if they become part of the IT products built by commercial vendors, the worked examples should be planned up-front in such a way that the resulting technology can be directly transferred to the U.S. IT industry. Thus, in addition to the technology and resource planning typically done on research projects, it will be necessary to address the legal and economic issues in such a way that we get maximum benefit from the results. Interoperability concerns (see next section) also need to be addressed as part of the planning. It's difficult to estimate the investment cost of these worked examples; the conjecture here is that they will be relatively small and perhaps could be shared by government and the commercial concerns likely to benefit from the results. The potential benefits to the community, on the other hand, could be enormous.

7. Increase investment in efforts to promote interoperability of security mechanisms across heterogeneous products

As already discussed, the 1980s saw a marked improvement in the types and quality of the protection features built into commercial IT products. For several reasons, the community was much less successful in incorporating adequate security into distributed systems made up of heterogeneous products. One reason is the lack of a body of accepted knowledge and associated technology required to allow us to deal with issues of composition; putting together a secure distributed system is still more an art than a science. On a more practical level, however, the lack of attention to interoperability of protection mechanisms across different products has been equally contributory. If a system integrator wants to build a system from heterogeneous pieces, he or she usually finds that the I&A, audit, and labeling mechanisms in the different components, while each satisfying TCSEC requirements, are implemented in such fundamentally different ways as to make interoperability difficult or impossible. Focused as we were on getting better products built it's not surprising that we were less concerned about how those products worked together. In the 1990s, however, we need to ensure that we can plug components together and still be confident that protection features work smoothly and effectively.

There are a number of existing interoperability initiatives on which we can build. The Trusted Systems Interoperability Group (TSIG), that initially grew out of efforts to make Compartmented Mode Workstations (CMW) work together, now has representatives of more than 25 producers of IT products and systems as members [7]. The TSIG is developing interoperability standards in such areas as labeling (an Internet Protocol Security Option), a trusted network file system, system administration, and trusted X. Functioning as it has at the technical level, the TSIG has been admirably free of the usual bureaucratic overhead associated with standards efforts, and has made considerable progress in the areas it has addressed. The challenge for the community is how to institutionalize the results of the TSIG effort, i.e., get producers to build to and consumers to buy from its standards, without destroying the relatively free and open environment in which the work has been done to date.

The TSIG work is an excellent foundation for some of the interoperability needs we face in the coming years. To fully achieve secure and usable distributed systems it is necessary to do additional work in the areas of I&A, audit, and labeling. Achieving

a secure unitary I&A design that works smoothly and transparently across different manufacturers' products is an absolutely essential prerequisite for widespread acceptance of IT security. Nothing in the security realm turns off IT users more than the need to login at multiple points in a distributed system. Kerberos, developed under MIT's Project Athena, is close to becoming a *de facto* standard in this area, despite security concerns expressed from a number of quarters [12]. PMSP is another useful step in moving toward an eventual unitary Login. NIST, NSA, and DISA have begun a collective effort to define an acceptable federal government I&A architecture for a distributed computing environment. Given the ubiquity of Kerberos, the challenge will likely be to use it as a foundation, and to build in necessary security improvements developed as part of PMSP and other programs. The government can play an important catalytic role in this activity, by bringing together the experts from the IT vendors, system integrators, and academia. Ultimately, however, it will be the vendors who will be most instrumental in building the products with interoperable IT security mechanisms.

The need for an audit reduction/intrusion detection system that will operate in a distributed computing environment was discussed in the previous section. Such an effort can build on existing single-system implementations, e.g., MIDAS, UBART, IDES, DIDS, but, in addition, needs to define and implement the architecture and protocols necessary to exchange and process audit in a distributed environment. Some promising work has already been done that used the OSI system management protocols for this purpose [2]. While far from complete, this is at least a promising beginning in achieving audit (and system management) interoperability.

SUMMARY

In a rather sweeping treatment, this paper has tried to summarize some of achievements of IT protection in the 1980s, examine some areas where we still have shortcomings, and suggest some potentially fruitful directions for work in the 1990s. None of the strategies proposed should require large amounts of investment capital. Nevertheless, they are not free, and will require some public and private commitment and investment. More importantly, they will require active and continuing cooperation between public and private institutions if we are to make the strides necessary to significantly improve the protection posture of IT systems in the next decade. In executing some or all of these strategies, this partnership will be one of the overriding requirements. The exact form of this partnership and specific proposals for implementing these (or other) proposed strategies are the tasks ahead of the community in the years ahead.

REFERENCES

1. Anderson, J.P., *Computer Security Technology Study, Volume 2*, NTIS-AD-772 806, NTIS, October 1972.
2. Banning D., et al., "Auditing of Distributed Systems", *Proceedings of the 14th National Computer Security Conference*, Washington, DC, October 1991.
3. Bell D.E. and L.J. LaPadula, *Secure Computer Systems*, MITRE Technical Report MTR-2547, 1973.

4. Clark, D.D. and D.R. Wilson, "A Comparison of Commercial and Military Computer Security Policies", *Proceedings of the 1987 IEEE Symposium on Security and Privacy*, Oakland, CA, 1987.
5. Clinton-Gore National Campaign Headquarters, "Technology: The Engine of Economic Growth", Final Version, September 1992.
6. Congress of the United States, *Computer Security Act of 1987*, Public Law 100-235, January 1988.
7. Cummings, Paul T, "Initiatives of the Trusted Systems Interoperability Group", *AFCEA Computing Conference and Exposition Proceedings*, Arlington, VA, February 1993.
8. Denning, Dorothy E., "A New Paradigm for Trusted Systems", *Proceedings of the 15th National Computer Security Conference*, Baltimore, MD, October 1992.
9. Federal Register, Executive Order 12356, "National Security Information", April 1992.
10. Garfinckel, Simson and Spafford, Eugene, *Practical UNIX Security*, O'Reilly and Associates, Inc., 1991.
11. Heberlein, L.T., et al., "A Method to Detect Intrusive Activity in a Networked Environment", *Proceedings of the 14th National Computer Security Conference*, Washington, DC, October 1991.
12. Krajewski, Marjan, "Concept for a Smart Card Kerberos", *Proceedings of the 15th National Computer Security Conference*, Baltimore, MD, October 1992.
13. National Computer Security Center, *Department of Defense Trusted Computer Evaluation Criteria*, December 1985.
14. National Institute for Standards and Technology, *Guide to Federal Agencies on the Use of Trusted Systems Technology*, NCSL Bulletin, July 1990.
15. National Institute for Standards and Technology and National Security Agency, *Federal Criteria for Information Technology Security: Version 1.0*, December 1992.
16. National Research Council, Clark, D. D., et al., *Computers at Risk: Safe Computing in the Information Age*, National Academy Press, 1991.
17. National Telecommunications and Information Systems Security Committee, *National Policy on Controlled Access Protection*, July 1987.
18. Neumann, Peter G., "Rainbows and Arrows: How the Security Criteria Address Computer Misuse", *Proceedings of the 13th National Computer Security Conference*, Washington, DC, October 1990.
19. President's Council on Integrity and Efficiency, *Review of General Controls in Federal Computer Systems*, U.S. GPO, Washington, DC, October 1988.

20. Sebring, Michael, et al., "Expert Systems in Intrusion Detection: a Case Study", *Proceedings of the 11th National Computer Security Conference, Baltimore, MD, October 1988*.
21. Spafford, Eugene, "Crisis and Aftermath", *Communications of the ACM*, Vol. 32, No. 6, June 1989.
22. Stoll, Clifford, *The Cuckoo's Egg*, Doubleday, 1989.
23. Treasury Board of Canada Secretariat, *Security Policy and Standards*, December 1989.

Panel: Strategies for Integrating Evaluated Products

Dr. James G. Williams, Chair
The MITRE Corporation
Bedford, MA 01730
jgw@mitre.org

Mr. Paul Boudra
National Security Agency, MS I9
9800 Savage Road
Fort Meade, MD 20755

Ms. Hilary Hosmer
President, Data Security, Inc.
Bedford, MA 01730
Hosmer@dockmaster.ncsc.mil

Mr. Milan S. Kuchta
Senior Scientific Advisor, INFOSEC
Communications Security Establishment
Ottawa, Canada

Dr. John McLean
Center for High Assurance Computer Systems
Naval Research Laboratory
Washington D.C. 20375
McLean@itd.nrl.navy.mil

Ms. Ruth Nelson
Information System Security Consultant
48 Hardy Ave.
Watertown, MA 02172
rnelson@cs.umb.edu

Mr. Bill Shockley
Cyberscape Computer Services
1885 Franklin St.,
Lebanon, Oregon 97355
72604.3077@compuserve.com

Mr. W. Olin Sibert
President, Oxford Systems, Inc.
30 Ingleside Road
Lexington, MA 02173
sibert@oxford.com

Dr. Bhavani Thuraisingham
The MITRE Corporation
Bedford, MA 01730
thura@mitre.org

Summary of Panelist Positions

By way of introduction, Kuchta introduces a simple taxonomy or hierarchy with which to discuss degrees of system integration. Kuchta and McLean both point out inherent difficulties in achieving success. Their papers identify essential prerequisites for achieving successful, assured integration of products to form trusted systems. Sibert counters with examples of new products that feature trusted networking interfaces.

Four panelists focus on pragmatic approaches for near-to-midterm improvement. Nelson argues that evaluation standards must depend more on product function. Hosmer recommends that relevant product protection profiles be named in message labels, in order to help hosts and trusted network interfaces make access-control decisions. Shockley advocates the use of partially ordered integrity labels for similar reasons. Thuraisingham suggests a general, object-oriented approach for connecting potentially dissimilar database systems.

Finally, Boudra proposes a long-term plan for success. Both Boudra and Kuchta point to ISO reference models as instructive examples that accomplish part of what is needed for success.

What is An Integrated System?

View Point by Milan S. Kuchta

This position paper discusses various levels of system integration, argues that current systems are not well integrated, and lists some necessary prerequisites for progress.

Integrating Systems: What's The Problem?

Current systems are NOT [real systems]! Problems are too numerous to individually catalog. The issue is that current "systems" do not function as systems but rather as loosely associated (or, more often disassociated) groups. This may seem desirable, but it provides too much opportunity for:

- (1) inconsistency –things don't "fit" together.
- (2) incoherence – things don't work together toward a common goal.

Many systems have (in many cases) been constructed with marginal models which are incompatible (i.e., no useful morphology exists between the models used in components of the system) and incomplete with respect to structure and behaviour. As a result, most "integrated" systems are unpredictable at some level of their operation.

Systems can usually be determined to be in one of the following somewhat hierarchical categories:

- (1) co-existent, (2) co-aware, (3) consistent, (4) coherent

Co-existent systems are not integrated in any systematic way. *Co-aware* systems are able to communicate and to selectively address one another. *Consistent* systems are able to perform differing functions on related data in such a way that things fit together properly. Finally, *coherent* systems are able to provide coordinated effort in the service of shared goals. Current integration of systems generates primarily the first two kinds of system and sometimes the third but rarely the fourth kind.

A Better-Understood Common Foundation/Framework is Required

It appears to be unreasonable (which doesn't mean that it actually is unreasonable) to require all systems to conform to some very specific criteria. Compatibility comes from conformance to uniform standards and coherence comes from conformance to uniform goals and objectives.

A Generic Systems Model

Some generic form of systems model is required which can be used as a grounding point (i.e., common reference) for all the specific models which exist or will be developed. Some might say we have this in models like the ISO OSI reference model. Such a model is a

beginning but applications are still free to interact wildly within such a framework model. The generic model must be more comprehensive in both scope and nature. It must be able to represent all relevant structure and behaviour. Please note that this does not advocate its use to represent all relevant structure and behaviour. Specific contextual models (which may be much more efficient for design and analysis) should all have a well specified morphology with respect to the generic model.

A Generic Security Model

In conjunction with a generic systems model used by designers and developers, a generic security model is required. Again, this does not advocate the sole use of this model but simply that every model that is used have a mapping to and from the generic model.

Common Semantics Are Required Regardless Of Syntax

The essence of effective systems integration is this: it doesn't matter how you describe or represent a subsystem, the subsystem must have a set of specifications which are semantically relatable to all the other subsystems in an integrated system. If there are no meaningful (semantic) relations between components of a system, there is no meaningful system!

Integrating Specifications, Integrating Assurances

View Point by John McLean

Evaluated products come with a specification and some assurance level that reflects the accuracy of the specification. Composition of evaluated products, therefore, entails composition of specifications and of assurance levels. This paper examines both types of composition and draws some conclusions.

Introduction

Assume (in a "naively optimistic" sort of way) that we have a collection of evaluated products, each with a specification of the product's user-visible properties and some assurance level that indicates how accurate the specification is. Further, assume that we have an architecture for a proposed system that consists entirely of these products integrated in some specified way. Is there any way to produce a specification for the composite system and an assurance level that accurately reflects the chances that the composite system meets its proposed specification?

Put this way, the question of "How do we integrate evaluated products?" can be broken down into two parts: "How do we integrate specifications?" and "How do we integrate assurance levels?". I propose to address each of these questions in turn.

Integrating Specifications

Specifications are descriptions of user-visible system properties. They state the system's domain of acceptable inputs and the system's range of acceptable outputs given these inputs. For functional specifications (e.g., "Given any two integers x and y between $-n$ and n for inputs, the system will produce $x + y$ as output."), the integration problem is pretty well understood.¹ If we use the output of one system as input to a second, the functional behavior of the composite can be determined from the functional behaviors of the two components as long as the first system's output respects the second system's input constraints.

When we turn to security, the matter is not so simple. Standard work on system composition does not apply since security is not a standard property. Most standard software engineering methodologies view properties as sets of traces and view a program as satisfying a property if its acceptable traces are a subset of the property. The rub is that properties such as noninterference are not sets of traces, but rather properties of sets of traces — i.e., meta-properties. This follows immediately from the fact that security is not preserved by trace subsetting. For example, consider a system that allows high-level input and low-level outputs, and assume that security is some set of traces S . A system X that consists of all traces would certainly be secure and would thus be a subset of S . However, a system Y that consists only of those traces of X in which high-level input was immediately echoed as low-level output would be nonsecure and thus not a subset of S . We are faced with the contradiction that X is a subset of S and Y is a subset of X , yet Y is not a subset of S .

McCullough has shown that the composition of specifications can violate security policies satisfied by each component specification.² Cases unrelated to McCullough's example are not hard to come up with. For example, imagine the increase in covert channel capacity that would be obtained by integrating an extremely accurate clock into a system that does not currently possess one. Similarly, the addition of an extra CPU to a secure architecture can be devastating, as can be the effect of composing a crypto box with itself.

It is also the case that the security of a composition can be greater than security of its components. For example, consider a system where high can communicate to low via a binary symmetric channel with crossover probability of .2. That is, if high sends a 1, then low will receive a 1 with probability of .8 and a 0 with a probability of .2, and if high sends a 0, then low will receive a 0 with probability of .8 and a 1 with a probability of .2. The capacity of such channels is well known (see, for example, the text by Jones³). In this case, the capacity is .28 bits. However, if we compose this system with itself, we increase the crossover probability to .32. This reduces the capacity to .09 bits, a decrease of over 64 %.

When we move from information flow to access control, we are on a bit firmer footing in some ways, but slipperier footing in another way. For example, McLean has given an algebra for combining MAC specifications that support different downgrading policies.⁴ This algebra can be used to compute the security policy supported by a composite system that is made up of Boolean combinations of its components. However, although we may be able to determine the policy that a given system supports, this does not help us in determining the policy that we wish to be supported. As noted by Hosmer, the real problem with combining access control policies is determining the policy to be followed when the component policies contradict each other.⁵

Integrating Assurance Levels

The higher our assurance that component systems meet their specifications, the higher is our assurance that composite systems made from those components meet their specifications. Unfortunately, low assurance component systems provide very little assurance about their composites. Part of the reason for this is that if we are not certain about the behavior of system parts, this uncertainty is bound to be increased as we add more parts. Further, part interaction may amplify undesirable system properties. For example, returning to the information flow realm, consider a system that contains a covert channel and some bandwidth estimations for various attack scenarios. These estimates are usually taken as providing us with some assurance that covert channel capacity is below a certain threshold. However, they really tell us very little about the channel's capacity (i.e., the maximum bandwidth that can be obtained through optimal coding). If we append to this system another system that performs optimal coding, we may be able to raise the high-to-low throughput for the same scenarios above the given threshold, destroying any assurance we had that the capacity of the original (or composite) system was below the threshold.

Conclusions

I think we can draw the following morals from these considerations:

- Composition of security properties is more difficult than composition of functional properties and requires research beyond what standard software engineering practice has provided us.
- We must know what policy we wish to enforce before we can begin composing systems.
- We can have high assurance in our composed specification only if we have high assurance in our component specifications.

References

1. Abadi, M. and Lamport, L. *Composing Specifications*, Technical Report 66, Digital Equipment Corporation Research Center, Palo Alto, 1990.
2. McCullough, D. "Specification for Multi-Level Security and a Hook-Up Property," *Proceedings of the 1987 IEEE Symposium on Research in Security and Privacy*, IEEE Computer Society Press, 1987.
3. Jones, D. S. *Elementary Information Theory*, Clarendon Press, Oxford, 1979.
4. McLean, J. "The Algebra of Security," *Proceedings of the 1988 IEEE Symposium on Research in Security and Privacy*, IEEE Computer Society Press, 1988.
5. Hosmer, H. "Metapolicies II," *Proceedings of the 15th National Computer Security Conference*, Baltimore, October 13-16, 1992.

Integrating Products into MLS Networks

View Point by W. Olin Sibert

Increasingly, MLS computer products are available today that can interoperate productively in a multi-vendor environment. This includes both traditional minicomputer/mainframe systems and workstations such as those developed for the Compartmented Mode Workstation (CMW) program. It is possible today to plug different products together to create a multi-level TCP/IP network that provides most of the standard services of the Internet protocol suite. However, different vendors have chosen different technical approaches and different degrees of support for the Internet protocols, and standardization is still a long way away, despite efforts such as the TSIG interoperability group and the MAXSIX consortium. This talk will discuss some of the approaches for MLS networking in commercial products and describe how they do (or don't) work together.

The Role of Function in System Integration

View Point by Ruth Nelson

The integration of products or components into a system requires an understanding of both the functional requirements of the system and the functional capabilities of each of its products or components. The system is designed to do some particular collection of functions, and each of its components plays some role. This is true whether or not the system must be secure. If evaluated products are to be useful in secure systems, we must be able to understand their role in the system environment. This role includes special security functions, but viewed in the context of the mission of the system as a whole.

Product security evaluation criteria attempt to be independent of the application or use of the products being evaluated. This is true of the TCSEC and it is also true of the newer criteria, including the Federal Criteria. The result is, unfortunately, that the evaluation level of a product may not give much information about its applicability in a secure system or about the security of a system which relies on that product for some of its functions. In addition, product evaluation is currently done on a particular configuration; when new components are attached, the evaluation may or may not still be valid.

Functional characterization of evaluated products and the development of some standard product types can provide some understanding of secure integration and improve system security. This paper will give a few examples of product types and their places in information systems.

Processing Platforms

One interesting and commonly needed product is a physical computing platform with its associated operating system software. In the 1970s, these platforms consisted of general-purpose time-sharing operating systems running on isolated computer systems with dumb terminals for user access. The popular computer security models, including the Bell & LaPadula and noninterference models, were developed for securing these operating systems against threats of unauthorized disclosure. These models and the currently available criteria include some assumptions which reflect general-purpose operating system requirements and use. They assume that most application software in the system runs on behalf of human users, is subject to change, and does not perform security-critical processing. The TCB, which is mostly application-independent, is supposed to maintain the security of the system no matter what the users and application software attempt to do.

This understanding and the evaluation criteria which embody it have been extended and interpreted to apply to dedicated processing systems, networks, data base management systems, etc., as well as to the operating systems for which they were designed. The assumption is made that security requirements, properties and functions are the same for these different kinds of products. This abstract outlook does not give much insight into integration.

Today, some processing platforms are used for general-purpose computing. This is the category of product where the existing criteria are the most useful. Other processing platforms are dedicated to running mission-critical, security-sensitive application software. In these systems, the data flow model of security and the subject-object paradigm may not

apply. A large fraction of the application software may perform functions which affect security of data or of the system itself. Current evaluation criteria do not deal well with this kind of processing requirement.¹ For this kind of use, the security engineer needs to understand how well the product protects the application software from unauthorized change or misuse.

Software packages which run under the control of operating systems constitute other product types. Some of these, like data base management systems, have been addressed by security criteria; others have not. It is important for the system integrator to understand the interactions between these software products and the platforms on which they run.

Communications Products

Most information systems today include both processing and communications. Looking at the different security problems of processing (changing) data and of moving data has proven productive in some of our security research.² Suitable definitions, mechanisms and assurances for confidentiality, integrity and access control differ between the processing and the communications functions. Applying existing evaluation criteria to communications systems ignores differences in requirements and function and has resulted in difficult problems of requirements interpretation in design and evaluation of communications products.

If the communications part of the information system includes switches, routers, gateways, etc., these can be considered as dedicated processing platforms and evaluated as such. It is also necessary to understand the security protocols, cryptographic products and management products used in the network. Programs such as the Secure Data Network System³ have developed architectures and protocols. These standards help define the functions of the communications software and hardware in the system. The correct performance of these functions, even in a hostile environment, is the essence of security for communications products and should form the basis for their evaluation.

Integration Process

System security can never be separated from the correct functioning of the system. It is never quite generic. Standards and evaluated products can provide some building blocks, but systems engineering skill will always be needed to analyze specific system security requirements, examine available products (evaluated or not), select the appropriate functional components and integrate them into a secure and useful system.

References

1. C. Limoges, R. Nelson, J. Brunell, J. Heimann, "Security for Mission-oriented Systems," *MILCOM '92*, October 1992, San Diego, CA.
2. R. Nelson, D. Becker, J. Brunell and J. Heimann, "Mutual Suspicion for Network Security," *13th National Computer Security Conference*, October 1990, Washington, DC.
3. R. Nelson, "SDNS Services and Architecture," *National Computer Security Conference*, September 1987, Baltimore, MD.

Multipolicy System Composition

View Point by Hilary H. Hosmer

Introduction

One of the hardest problems in multilevel secure (MLS) system accreditation is knowing what level of trust one has achieved when combining different kinds of systems evaluated at different Trusted Computer System Evaluation Criteria (TCSEC or Orange Book) levels. For example, if an A1 network links an A1 system, four B3 systems, and fifty C2 workstations, what is the rating of the combined system? Must it be rated at C2, the lowest level? The Trusted Network Interpretation (TNI or Red Book) addressed this problem for TCSEC-based systems, but it has not yet been addressed for the proposed Federal Criteria¹ and Multipolicy Paradigm² systems.

Both the Federal Criteria (FC) and the Multipolicy Paradigm compound the MLS system composition problem by permitting many more combinations of policy, functionality and assurance levels. For example, when there are multiple policies enforced, each policy may involve multiple sensitivity levels, producing multiple variations of the "cascade" problem. Similarly, protection profiles each combine many different components rated at different levels. When systems based upon many different protection profiles and many different policies are combined together, what evaluation rating could these systems have?

Theses

This paper takes three positions, introduces a high-level model to support them, and explains the philosophy of protection that motivated this model. These positions are:

1. Composite ratings for a heterogeneous system are generally meaningless;
2. A composite multipolicy system should guarantee that the appropriate policies are correctly enforced by systems which meet the appropriate protection profile requirements for functionality and assurance;
3. End-to-end encryption and restrictions based on one or more protection profiles may support dynamic but secure networks of indefinite size and flexibility.

¹ *Federal Criteria for Information Technology Security*, Volume 1, National Institute of Standards and Technology and National Security Agency, December 1992.

² Hosmer, Hilary H., "The Multipolicy Paradigm", *Proceedings of the 15th National Computer Security Conference*, Baltimore, MD, October 1992.

Components of a High-Level Model

The above-mentioned high-level model is a high-level model consisting of the components mentioned in figure 1 below.

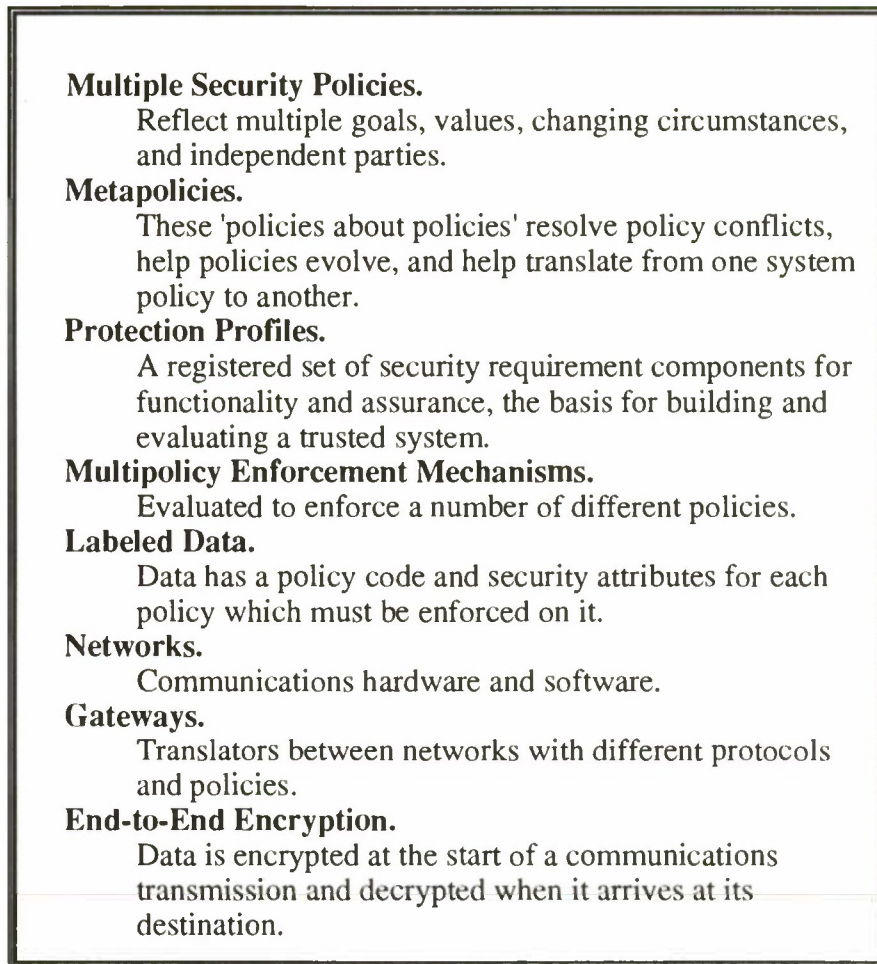


Figure 1. Components of a Multipolicy Composite Model

Philosophy of Protection

End-to-end encryption protects data in transit across multiple networks and enables data to be transmitted through untrusted or less trusted machines. In the multipolicy environment, encryption via different keys separates data on which different policies are enforced and data with different security levels. Thus, only machines with the proper policies and security levels can produce cleartext.

When data is exported from a machine onto the network, its tamperproof packaging may identify not only the protection profile of the exporting product or system, but also restrict

which protection profiles are permitted to process this data. This ensures proper security functionality and assurance in a varied world of many incompatibly-evaluated systems.

Many supporting assumptions are necessary, such as standard multipolicy enforcement mechanisms, and a network-wide key management system. These may belie the heterogeneous network goal.

PROTECTION PROFILES: PP1 PP2 PP3 POLICIES: P1 P3 DATA: Fred Jones BB Hospital....

Figure 2: Decrypted Data Packet with Receiving Profiles and Policies

Summary

Data put out on the heterogeneous MLS multipolicy network should carry with it information about which FC-style protection profiles are permitted to process it as well as what policies and sensitivity levels must be enforced on it. This guarantees that the proper kinds of evaluation will be enforced in a varied network of many incompatibly-evaluated systems. Because of unequal levels of evaluation, end-to-end encryption will protect data as it travels through untrusted or less trusted systems. By making data accessible only by those systems which will handle it appropriately, end-to-end encryption and protection profile restrictions may permit dynamic heterogeneous networks of indefinite size.

Acknowledgments: The Air Force Materiel Command, ESC/ENS, at Hanscom Air Force Base, Bedford, MA and the Air Force Cryptological Support Center, AFCSC/SRER, at Kelly Air Force Base in San Antonio, TX, supported this research through the Small Business Innovative Research (SBIR) program. Several colleagues provided constructive criticism: Dr. Grace Hammonds of AGCS, Dr. Jim Williams of MITRE, Allan Grachan of DSD Laboratories, Inc., Olin Sibert of Oxford Systems, and Eric Leighninger of Arca Systems.

References

1. *Federal Criteria for Information Technology Security*, Volume I, Version 1.0, The National Institute of Standards and Technology and The National Security Agency, December 1992.
2. Hosmer, Hilary H., "The Multipolicy Paradigm," *Proceedings of the 15th National Computer Security Conference*, Baltimore, MD, October 1992.

Protection of Distributed Applications

View Point by Bill Shockley

Introduction

First things first! If we are going to ask our distributed service to enforce anything at all, we must first know how its trusted components are to be protected.

This becomes a knotty issue in a large-scale client-server environment. The key problem becomes: how are we to implement a "trusted server" (and what does that mean) where clients may be hosted on heterogeneous nodes. We may trust the clients themselves to differing degrees, or, indeed, clients may be running on behalf of mutually suspicious organizations with quite different ideas about what it means to be "protected" or "trusted" in the first place. If I happen to trust a particular service, I would like to be able to use it to support my own "trusted application". However, if I don't happen to trust it, I don't want the fact that somebody else does to raise the possibility that I might inadvertently use it.

In order to meet these goals, a more complex model of protection is needed than is assumed by such documents as the TCSC, TNI, TDI, etc. In this paper I sketch such a model. Brevity precludes a rigorous presentation — a more complete paper for submission to a refereed conference is in preparation. Here, "proof by emphatic assertion" must suffice.

The model contains the following primary components:

- 1) Structural submodels that (a) describe the aggregated local execution *domain* structure available for the various nodes of interest in the network, and (b) show how components of potentially distributed applications are allocated to domains.
- 2) An abstract model of *communications objects* that allow components in different nodes to communicate. The primary intended mapping to a communications object is an end-to-end encryption channel.
- 3) A node-local model for the labeling of selected execution domains (and by implication, messages from them) with a consistent set of *trust labels*.
- 4) A *Basic Protection Theorem* and corollaries, along with their proofs.

Structural Submodels

The structural components of the model include a set D of (local) execution domains, arranged into a hierarchy by a "privilege" relation P . P organizes the domains into a set N of one or more directed, acyclic graphs called nodes. There is also a set O of objects, each labeled with a domain label. "Domains" play the part ordinarily played by "subjects" in security models. Rules constraining how domains may access and invoke objects labeled for other domains are given. Essentially, if d_1 is privileged with respect to d_2 , p_1 may be given

access in all modes to d2's objects. Finally, potentially distributed applications, are divided into pieces, called software components (SCs), associated with particular domains. One of the model requirements is that if one domain is privileged with respect to a second, then software components in the latter domain must depend on those in the first (but not conversely).

The Communications Model

Two components of the same application in different nodes communicate — but not via privilege or (local) "invocation". So we must model some "invocable" operations on objects that facilitate inter-node communication. To this end we introduce a subclass of objects *C* called *communications objects*. At the level of detail of the model, the smallest addressable entity is a domain, and we represent a "message address" as a domain label. In fact, for greatest generality, each communications object is assumed to have a set of domains allowed to send messages via the object, and a set of domains allowed to receive messages via the object. As part of the defined operator semantics, each message is labeled with the domain of the sender. As a semantic postulate, the domain implementing a communications object is required to obtain and append this argument.

There is a key point here that I would like to emphasize. It is essential that the communications service (not the sender) provide a proper domain identifier (or equivalent). Allowing senders to provide source domain labeling turns out to be useless for building general purpose "trustable servers". The critical case is where clients I wish to trust and clients I do not trust coexist on the same node. If the communications service (which I trust) does not do source domain labeling (or its equivalent) on the sending end, I have no way to tell whether the source is actually the client domain I trust, telling the truth; or the client domain I do not, telling a lie. Similarly, source labeling by protected communications software to the granularity of a whole node, or to a process or a thread using names that cannot be associated with a domain, are also practically useless for purposes of distributed protection.

Unfortunately, since the lack of trusted labeling doesn't compromise the communications service itself, it is often left out — making it impossible to erect useful "secure servers" that are CLIENTS of the communications service.

The Trust Model

So far, we have taken a global view of the system and have modeled enough structure to understand its organization — without introducing the notion of "trust". Because the notion of trust may vary from network user to network user, I model trust as a configurable "policy" that can have different contents for different nodes. Different nodes might have quite different policies. It is plausible (from the point of view of enforcing a protection policy) to identify an administrative domain as a set of nodes having the same defined trust policy. The node-local trust policy TP for a given node contains the following essential parts:

- 1) A set *L* of *trust labels*. Note that trust labels are quite distinct from domain labels!

- 2) A *less-trusted-than* relation T that partially orders L . (Technically, it must be at least a complete lower semi-lattice with a universal lower bound. Whew!)
- 3) An complete function $TA : D \rightarrow T$ which assigns trust labels to domains in D . (In practice, TA is given by explicitly specifying trust labels for some domains and implicitly mapping all others to a distinguished totally untrusted label U .)

The trust model axiomatizes a number of obvious consistency constraints on the contents of TA , e.g., that the trust levels must be compatible with the privilege and dependency orderings. The primary labeling rule, however, is this: when I receive a message, it must be received into an object with a trust label that is less than or equal to $TA(d)$ where d is the source domain label provided in the message. As a semantic axiom, it is assumed that the receiving component is "smart enough" to know what to do with messages from sources less trusted than itself. What this means is application dependent, but it implies that all "protection-critical" decisions be based only on messages that are at least as trusted as itself.

The Protection Theorem

With a few more flourishes, a nice set of basic theorems can be shown.

- A. Suppose no domain at trust level t and up has been corrupted anywhere in the system. Then there is no way at node N to incorrectly label any message with trust level t or up, and any incoming message so labeled is, in fact, from some domain trusted at level t or up.
- B. By induction, the same property holds for any subsystem of nodes sharing the same trust policy TP . (I.e., I can build a distributed trusted application in a set of "add-on" nodes having "trustworthy enough" domains.) Note that a plausible definition of "administrative domain" is such a collection of nodes.
- C. By a second induction, the same property holds for ALL administrative domains jointly! A surprising, and pretty result.

Practical Application

The final bit needed to make a practical system work is to use "vector trust labels" — i.e. *descriptors* — to express complex trust policies. This requires defining a plausible partial ordering for the vector labels: each component of a descriptor is itself drawn from a partially ordered set, and we take the Cartesian product. This approach lets us encode highly useful rules right into the trust label (ahem, descriptor) — e.g., regarding the authentication mechanism to use for communications, whether or not messages must be encrypted, etc. It appears that any reasonable description about what it means to be trusted can be encoded into such a vector label. Since trust labeling is a strictly local phenomenon, different organizations don't have to agree about what is important and what is not.

Object-Oriented Approach to the Interoperability of Trusted Database Management Systems

View Point by Bhavani Thuraisingham

This position paper describes an object-oriented approach to connecting trusted database management systems. In particular, an object model of the heterogeneous environment and the operation of the entire heterogeneous system are discussed.

Introduction

Many organizations have a number of computerized databases scattered across several sites. Efficient access to the information contained in these databases as well as sharing it has become an urgent need. As a result, an increasing number of heterogeneous database systems need to be interconnected. For many applications of heterogeneous database systems, secure interoperability is essential. Furthermore, for military applications, it is also important that such systems support a multilevel user/data handling capability.

Recently, there have been some developments in homogeneous trusted distributed database management systems and research is now beginning on interconnecting trusted database management systems (TDBMSs) in a heterogeneous and autonomous environment.

Interconnecting different TDBMSs raises several issues. These include handling (i) schema heterogeneity and schema integration, (ii) different security policies, (iii) different accreditation ranges, and (iv) different query and transaction processing algorithms. In order to resolve these issues, one needs to first develop a model of the entire heterogeneous system. This paper proposes the use of an object-oriented model for this purpose.

An Object Model of the Heterogeneous Environment

In this section we describe the essential points of an object model for the environment under consideration. In our model, every entity is an *object*. That is, an object could be a federation, a node, a database, or a TDBMS. We group collections of objects with similar properties into *classes*. The classes form *class hierarchies*; we support inheritance and encapsulation. The properties of a class are specified by *instance variables*.

Figure 1 represents the environment partially. The classes include FEDERATION and NODE. The instances of the FEDERATION class are the various federations. Each federation has the following instance variables: the federation-ID, the collection of nodes which form the federation, the federated schema, the federated security policy, and an administrator or group of administrators (if there is one). The NODE class has nodes as its instances. Each node has the following instance variables: the node ID, node-name, federations (the federations to which the node belongs), the accreditation range (i.e., the range of security levels processed by the node), the database system (this includes the local policy, the schema, the DBMS, and the database), the administrator, local users, and global

users. Administrator and users are instances of the PERSON class with instance variables which include Person-ID, Name, Type of user, and Nodes.

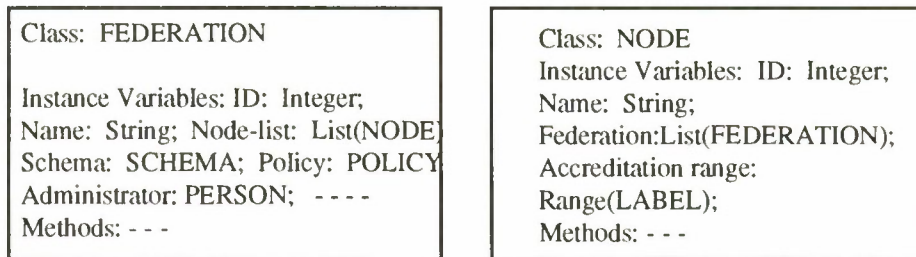


Figure 1. Sample Classes

In addition to the instance variables shown in the figure, each node will have database system as an instance variable. This instance variable is an object and represents the local database system. This system consists of the multilevel database, the local TDBMS, the local schema, and the local security policy. That is, the database system object is a composite object. The specification of the component objects are yet to be defined. Note also that the local TDBMSs may be relational systems or object-oriented systems. At a higher level of abstraction, we do not distinguish between these systems. That is, the interface to the database system object is uniform. The actual methods which implement the functions may be different for the various types of data models utilized.

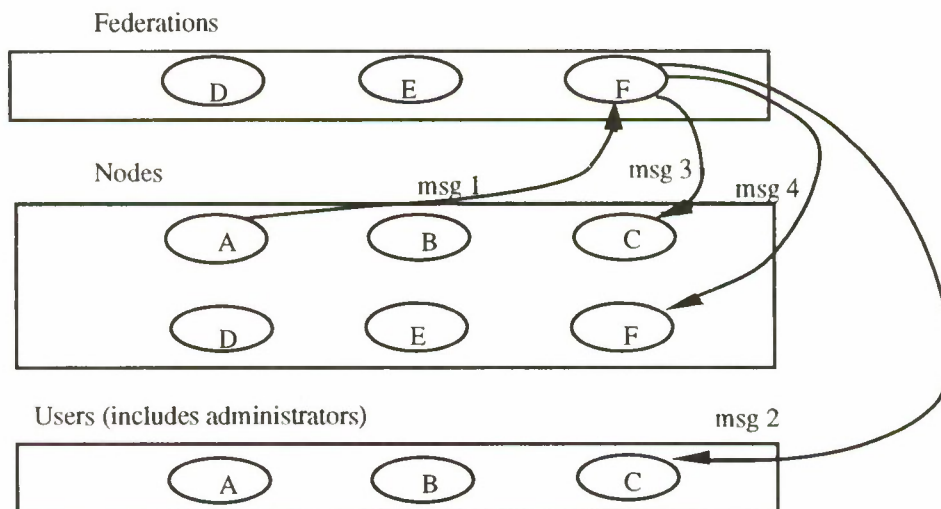


Figure 2. A Scenario

Figure 2 provides an example of how the system operates. In this example, node A wants to join the federation F. Node A sends a message to the class FEDERATION with federation F as a parameter (msg 1 in figure 2). Node A may also give some other information as to what information it needs from others and the information it is willing to share. When federation F gets the message, the corresponding method gets executed. The federated policy may be examined to see if A can join the federation (msg 2 in figure 2 where it is assumed

that the administrator C maintains the federated policy). It may send messages to nodes that are already part of the federation to see if these nodes are willing for A to join the federation (msg 3 and msg 4 in figure 2). If all checks are satisfied, F sends a message to node A to join the federation and it includes A as part of the list of nodes which belong to it. That is, the value of the instance variable of F which specifies the nodes gets updated. Node A in turn updates the value of its instance variable for the federation.

Expected Benefits and Future Considerations

To our knowledge this is the first model proposed for interconnecting different TDBMSs. An advantage of this model is the uniform representation of all of the component of the heterogeneous system. That is, the TDBMSs, the multilevel databases, the federations, the nodes, and the various types of users are all represented as objects. Interaction between the various components is achieved via message passing. The encapsulation feature of the object-oriented approach enables some degree of autonomy for the individual objects. For example, each TDBMS could implement its own algorithm for functions such as query processing and transaction management and yet be part of the federation.

Evaluating and accrediting the interconnected TDBMSs are still open issues. It needs to be determined whether the approaches specified in the Trusted Database Interpretation could be used to evaluate the heterogeneous system based on the object model described in this paper. For example, could the method of "evaluation by parts" be used to evaluate such a system?

With respect to accrediting the heterogeneous system, if the components have autonomy, then the individual Designated Approving Authorities (DAAs) will have the freedom to accredit their own systems. However, in order to accredit the heterogeneous system as a whole, there must be some negotiation between the DAAs of the individual TDBMSs and the DAA of the integrated system. With the object model proposed here, the local DAA's could be represented as components of the local nodes and the federated DAA could be represented as a component of the federation. The negotiation could be carried out via message passing between these component objects.

Much remains to be done on the interoperability of TDBMSs. We have specified only the essential constructs of the model. Issues on handling autonomy and heterogeneity need to be investigated further and appropriate constructs need to be incorporated into the model. Research should be directed not only towards handling different types of heterogeneity and autonomy, but also towards developing strategies for evaluating and accrediting the secure heterogeneous system.

Acknowledgment: Much of the information in this position paper has been obtained from the paper by Bhavani Thuraisingham and Harvey Rubinovitz on the object-oriented approach to interconnecting Trusted Database Management Systems. That paper was presented at the December 1992 ACM SIGSAC Workshop on Secure Distributed Database Management System in St. Antonio, TX.

Long Term Prospects

View Point by Paul Boudra

Building systems that provide strong defenses against attack is a very difficult task. It is difficult partly because of demanding expectations. Experienced integrators understand that the task is difficult but many people expect "state-of-the-art" security requirements to be met within the constraints of the existing equipment and low budget. It is difficult partly because systems are complex and security problems are often subtle. The designer must cover all bases, an adversary must only find one weakness. The weaknesses in systems are often not obvious. Finally, it is difficult because we do not have the right components. The components we have today are few and they were not designed to work together. Trusted operating systems were designed with the timesharing model of the mainframe in the protected computer room connected to physically protected dumb terminals by physically protected cables. The TCSEC is not an interoperability standard. It does not require the same abstractions, the same security policies for various machines. It does not address any machine-to-machine protocols. We have a few communications products, but no means for multi-level communications to hosts. The security policies supported by the available products do not map well to the policies of our customers.

The bad news is that building systems which meet their security requirements is not going to get easier in the near term. In the long term, however, there is an answer to building the right systems and to building the right components. That answer comes from experience in systems engineering. It is a top-down approach guided by experience. The concept is that system requirements drive system architecture. System architecture defines the necessary components and their requirements. This is no different than the position taken in the beginning of Appendix A of the TNI. We must take a broader long-term view of requirements and document those requirements in the appropriate Information System Security Policies. We must then develop the appropriate system architectures that meet those requirements. These architectures will determine the components to build. We must populate the architectures with standards so that we can achieve a system-homogeneous, vendor-heterogeneous situation. Finally, we must encourage the building of components that populate these architectures.

An analogous situation occurred in network communications. In the past, communications solutions were vendor proprietary and non-interoperable between different vendor's products. The ISO Basic Reference Model is a communication architecture that addressed the interoperability concerns of customers. That architecture was populated with standards that have allowed progress toward a system-homogeneous, vendor-heterogeneous solution.

16 th NATIONAL COMPUTER SECURITY CONFERENCE

Title: Multilevel Information System Security Initiative
(MISSI)

Chair: Gary Secrest, Office of Trusted Products, NSA

Overview:

The need for automated sharing of information at multiple security levels over open, uncontrolled networks is well known. The MISSI within the National Security Agency's Information Systems Security Organization is developing a suite of products to allow this sharing in a secure manner using both the existing DOD TCP/IP networks as well as the GOSIP-compatible networks of the future. This panel will explore the systems issues associated with security in open network systems and the phasing of a suite of products being developed to provide a range of both security features and assurances. That suite of products includes: workstation-based protection for unclassified but sensitive information (MOAIC) and classified information (Applique); Network Security Management (NSM); and the Secure Network Server (SNS).

Panelists:

Clark Wagner, NSA, MISSI
Bill Bialick, NSA, Workstation Security (MOAIC)
Mark Roberts, NSA, Secure Network Server (SNS)
Phil Quade, NSA, Workstation Security (Applique)
Robin Gerretson, NSA, Network Security Management (NSM)
Lee Johnson, NSA, MISSI Release 2 (MR2)

----- first session -----

MISSI Overview:

Mr. Wagner will discuss the requirements for security in open systems networks, and the MISSI approach to satisfying those requirements (both security features and assurances) with a "phased approach" and "graded assurance." Included in this presentation will be an overview of the process used to assure interoperability of the various MISSI components, and the relationship of the MISSI with evolving network standards. He will include a discussion of the MISSI evolutionary strategy and the phased releases proposed for development. (60 min)

MOSAIC:

Mr. Bialick will discuss the MOSAIC project, which provides writer-to-reader security services for unclassified but sensitive information. MOSAIC includes a combination of infrastructure, hardware, and software at the user workstation to provide confidentiality, integrity, non-repudiation, and user authentication. The hardware consists of the TESSERA Crypto Card and a PCMCIA reader to allow this peripheral to be used with the end user workstation. It will be available with both SMTP and X.400 and will use the X.500 protocol standard for certificate management heirarchy. It is designed to address the needs of Phase 1 of the Defense Message System program. (20 min)

----- second session -----

Secure Network Server:

Mr. Roberts will discuss the Secure Network Server (SNS), identifying the major guard, downgrader, and file server functions performed by the SNS and the principal physical components of the SNS, which will form the high assurance security backbone of the MISSI architecture. Although the SNS supports the basic Bell-LaPadula security model, it will be "customizable" to allow for site-specific security policies. This discussion will be followed by a description of how the SNS supports and operates as a component of MISSI. Finally, ongoing SNS acquisition efforts and plans for future efforts will be discussed. (20 min)

Applique:

Mr. Quade will focus on the APPLIQUE as a peripheral to a standard IBM-compatible workstation (minimum 386). That peripheral will provide cryptography for protection of information to achieve the overall goal of "writer-to-reader" information protection. He will highlight what's required of the workstation if the cryptography is not in-line (i.e., a security monitor), the requirements for the APPLIQUE security monitor, and why TMACH was selected for the APPLIQUE security monitor. (20 min)

Network Security Manager:

Mrs. Gerretson will open with a high level identification of the various aspects of network management. She will quickly focus on security management and the Network Security Management (NSM) components implemented in the Multi-level Information System Security Initiative (MISSI) architecture. These components include: an Audit Manager, a Directory Server, a Domain Security Manager, a Local Authority Workstation, a Mail List Agent and a Rekey Agent. Together, they will provide the infrastructure to support the security functionality of the various components of the MISSI approach to network security. An overview of the functional capabilities of each of the NSM components will be given.

MISSI Release 2:

Mr. Johnson will report on the status of release 2 of the MISSI (MR2). He will provide a presentation of the security capabilities of (MR2), which will include confidentiality, integrity, and proof-of-origin/delivery for e-mail at the Secret through Unclassified levels. This will be based on new and/or enhanced components such as the Secure Network Server (SNS), the Crypto Peripheral (CP), and Network Security Management (NSM) devices. He will highlight important program milestones, planned beta test activities, and initial system availability dates. (20 min)

TRUSTED APPLICATION PANEL

Panel Chair: *Janet Cugini, NIST*
Panel Participants: *John Campbell, NSA*
Mark Carson, IBM
Chii-ren Tsai, Citicorp
Stuart Stubblebine, ISI

Computer environments have drastically changed since the first days of the Orange Book. In the mid-80's, computers were mostly stand-alone, non-networked entities and all our security needs could be addressed by operating system security. Today, most computers can be found in heterogeneous networked environments, and the mounting of remote filesystems, distributed client/server applications such as databases, and world-wide Internet access are the reality. Of course, complex computing environments also increases the need for security and the complexity of the mechanisms that are needed to address security. And, unfortunately, all the security ramifications of this new distributed environment have yet to be worked out. For example, how do we deal with access control over a distributed network environment? How do we manage the security requirements of a distributed network environment? We are seeing the emergence of new trusted applications, such as secure mail and biometrics for authentication, but since so much of the focus has been on operating system security, we lose sight of basic research and product development in other types of applications and what type of secure applications are needed for the future.

Our panel cannot answer all questions on trusted applications, but we can explore some of these issues. Two of our panelists will discuss how to handle distributed issues of today, and two will talk about applications that are more in the area of basic research. John Campbell will discuss secure database management systems and future trends, and Chii-ren Tsai will discuss a prototype for a distributed security management system. Mark Carson will discuss how to supply security to hierarchically organized spatial or graphical data, and Stu Stubblebine will discuss the security infrastructure for multimedia conferencing. Today, operating system security is not enough, and as we move forward with different kinds of technology we must make sure that the security aspect is an integral part since our security concerns actually increase with a more complex distributed computing environment.

Janet Cugini,
Program Chair

Trusted Database Systems: An Application for Trusted Operating Systems

John R. Campbell
Campbell @ Dockmaster.ncsc.mil
National Security Agency

I. Importance of Database Management Systems

Database Management Systems are universal tools utilized by users in defense, manufacturing, finance, health, government and industry. Your payroll or social security check, your tax bills, your bank deposits and your health records all use database management systems. Over 90% of mini and mainframe computers have installed database systems.

II. Importance of Trusted Database Management Systems

The users mentioned in the first paragraph rely on the security and correctness of the database systems to accomplish their sensitive tasks. Trusted database systems undergoing evaluation, either single or multilevel, are being examined for security flaws. Security holes have been closed or controlled. In some cases, integrity has been added. Audit and Identification & Authentication capabilities have been expanded. Because of these characteristics, a reasonable person would use this type of database system when dealing with data that is sensitive, data that he needs to protect, or data that he relies upon.

III. Status

Approximately eight vendors currently sell trusted database systems. Three are in evaluation at the National Computer Security Center (NCSC). Two, which have been in evaluation for some time, are having products evaluated at the C2 and B1-levels. A third has just started the evaluation process. Most use a trusted subject architecture. One provides a choice of either TCB-Subset or trusted subject architecture.

IV. Usage Requirements

The Trusted Database System usually works with the Trusted Operating System and the hardware to provide security. It is therefore important to remember that a Trusted Database System is effective only when coupled with a specific Trusted Operating System and hardware. Therefore when the NCSC evaluates a Trusted

Database System, it is evaluated on specific Trusted Operating System and hardware.

Frequently the Trusted Database System provides critical discretionary access control and audit and sometime mandatory access control. This is because the controls are on small objects that users work with, such as records or tables, objects that database system manipulates, but are too small for the operating system to control. The operating system typically works with files, which may be composed of many thousands of records. The operating system can only grant access to the whole file or deny access to the whole file.

V. Future User Requirements

Users were polled for future database requirements. They desire more secrecy and integrity. They want systems that are always up. They desire client-server and distributed systems. They want systems where the labels on records, in multilevel systems, can be passed, in a trusted fashion, to multilevel applications sitting on the database system. They want to be able to work with very large databases. They want aggregation controls. They want multimedia systems. And they want to be able to do this with user-friendly, compatible, and high performance systems. This is a difficult list to achieve, but it provides us with a needed target, and one that we need to actively pursue, because it will not happen by itself.

VI. Summary

Trusted Database Management Systems have come a long way in a short period of time. Five years ago there were none. Today, you can purchase them off-the-shelf. They provide purchasers with increased integrity, secrecy and accountability benefits, while keeping many of the ease-of-use and compatibility benefits. Some users now realize the need for future systems that provide more assurance in the areas of secrecy, integrity and availability. These systems must be compatible with client-server and distributed architectures. These systems must be actively developed.

Labeled Quadtrees: Security and Geographical Information Systems

Mark E. Carson, Mudumbai Ranganathan

Secure Workstations Department
IBM FSC 182/3F42
Gaithersburg, MD 20879

Department of Computer Science
University of Maryland
College Park, MD 20742

1. Introduction

With the growth in processing power and storage capacity of commonly available computer systems, computerized geographical information systems (GIS) are becoming a practical reality for a wide variety of users. With the growth in high-speed networking and the large-scale collection of information through many means, large central repositories of geographically-related information shared among a diverse community of users are also becoming a reality. With these capabilities, however, come increasing concerns about the security and (lack of) privacy of the often sensitive information contained in these databases; a recent example was the controversy over Lotus's aborted Marketplace: Households product, a CD-ROM database containing Census and credit bureau information, which was withdrawn in response to widespread protests.¹

In this paper, we discuss how "Orange Book"—style security mechanisms may be applied to spatial databases in general, and to GIS in particular. We then describe a prototype secure spatial database we have implemented which employs these security mechanisms completely yet efficiently. The work described can be useful for many situations involving the security of hierarchically-organized spatial or graphical data, not just GIS.

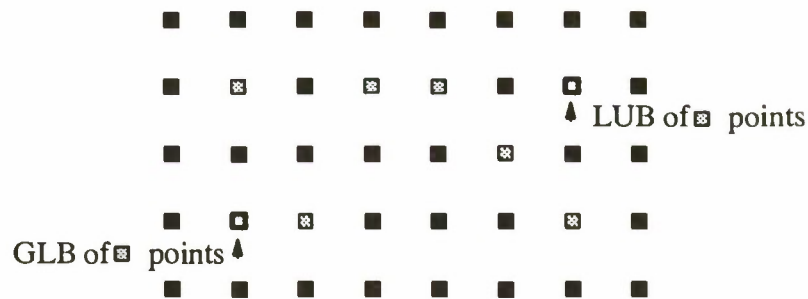
Disclaimer: The work described here is part of a research project. No IBM product commitment is made or implied.

2. Security background

We assume the reader is familiar with the basic security concepts of access control, both discretionary and mandatory, and with security labeling. See 23 for more details. In this project, our goal is to speed access control and labeling operations by making use of precomputed range information.

"Range" here refers to the bounds (GLB and LUB) on a subset of elements in a *lattice*. A lattice is a collection of elements along with a partial ordering such that the *greatest lower bound* (GLB) and *least upper bound* (LUB) of any subset of elements always exist. The GLB of a set of elements is the largest element which is smaller than any element of the set. Similarly, the LUB is the smallest element which is larger than any element of the set. Here is a schematic picture of a lattice, where the (partial) ordering is (smallest to largest) left to right and bottom to top.

The lattice ordering for the usual sensitivity labels used for mandatory access control is the obvious one, with labels corresponding to higher classifications/more compartments being



“greater.” The GLB and LUB also have a natural meaning in this context. However, LUB and GLB operations may also be applied to other security attributes in addition to labels. Here the interpretation is that if a subject has read access to the LUB of the security attributes of two objects, it must also have read access to the two objects. On the other hand, if a subject does not have read access to the GLB of the security attributes of two objects, it cannot have read access to either of the two objects. Thus for UNIX* permission bits, for example, the LUB amounts to an “and” operation, while the GLB is an “or.”

The operating environment for our secure GIS project is a multilevel X Window System†–based system, which provides individual security attributes for X objects.⁴ Our GIS takes advantage of this by controlling the security attributes of its display window based on the attributes it determines for the displayed information.

3. GIS and Security

Information in a GIS may be viewed (irrespective of the actual means of storage) as consisting of map *layers*, sets of a single data type accessible by map location.⁵ One type of security control one might expect in a GIS would be access controls on an entire map layer. For example, in a census database, average population density in a given area may be publicly viewable, while the layer listing average per-household income for a given location would hopefully be restricted.

Of more interest here are those areas where security controls may be expected to vary across space, within a map layer. One type is where controls vary with the degree of resolution. For example, average household income may be generally available for large areas, but require restriction as the area of inquiry shrinks down to an individual house. Another example is provided by military mapping information (including GPS location information), where lower resolution versions are generally available, while higher resolution ones are restricted.

Another type, of principal interest in our prototype system, is where the security controls vary from location to location. A simple example is provided by telephone numbers: some locations will have public numbers, while others will have unlisted numbers. Many companies have somewhat complicated rules about which phone numbers will be given to whom, which can be modelled as an access control policy. Another example which can actually be seen on local maps is information about restricted areas. “Detailed” local maps show few or no details about road and building positions in Camp David or the NSA area in Fort Meade, for example. Outside the

* UNIX is a trademark of UNIX Systems Laboratory, Inc.

† The X Window System is a trademark of MIT.

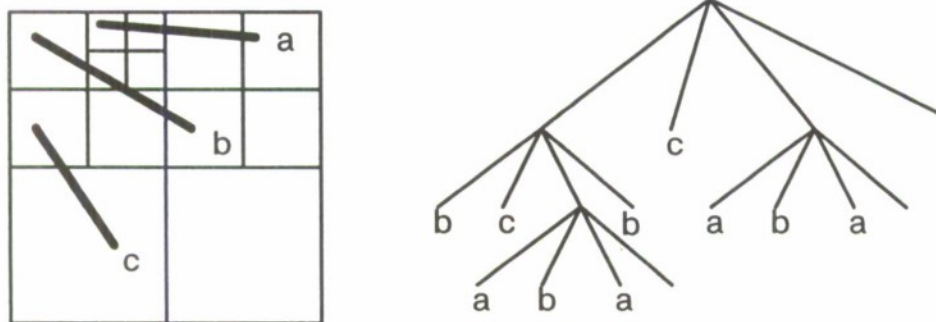
GIS area, this sort of variation of security controls by location may be fairly common in other graphical databases. For example, in a missile design database, access to guidance circuitry information might be restricted to one compartment, while access to armament circuitry might be restricted to another. At lower levels, these areas of the diagram would simply appear blank.

Given such security information, a variety of spatially-based security-related operations are possible. For instance, a user may want to delete all the classified or controlled-access information in a region when declassifying a map. Or, a user may want to list of all features that are classified at a given level. Another type of query may be to list of all features that are potential threats within a given distance from a sensitive location. Essentially, the security attributes of an object can be used in the same way as other object attributes in a GIS.

Common to most of the examples above is the expectation that locations with similar security characteristics will tend to cluster in space.* This makes feasible more efficient handling of security information by including it directly in the spatial indexing scheme. In this project, we have taken advantage of this idea, attaching security information to the nodes of RECT and PMR quadtrees.

3.1 Quadtrees

The *quadtree* is a data structure used for *spatial indexing* — organizing objects distributed in 2-dimensional space for quick access.⁶ As the name suggests, a quadtree works by recursive subdivision of space into quadrants. The subdivision proceeds until all objects are sufficiently well separated. A tree is built corresponding to the subdivision, and objects are attached to leaves in the tree according to which quadrant(s) they lie in. Quadtrees differ in which objects they handle, and the amount of separation between objects they require. The following is a schematic illustration of a simple PM₁ quadtree, which handles line segments and requires complete separation (if possible) of all objects:



For this project, we used two quadtree types, the RECT and PMR quadtrees, as the basic GIS organization. The RECT quadtree handles rectangular objects, allowing at most one object per leaf node. The PMR quadtree handles line segments, allowing multiple objects per leaf, but using a “divide once” heuristic to keep the per-leaf object list relatively small. In a GIS setting, the rectangles might represent bounding boxes for arbitrary objects such as buildings, and the line segments roads.

* Unlisted phone numbers tend to be concentrated in higher-income areas, though not in New York City, where apparently being “ex-directory” is unfashionable.

3.2 Secure GIS Organization

Our secure GIS prototype consists of an X-based front end, a spatial database indexed by the two quadrees, and a back end set of security routines.

Both the database information and the indexed map items themselves are assigned security attributes (ownership, access permissions, security labels) on creation. Users of the system have their own set of access rights (user and group IDs, privileges, security labels), which are used by the system both to control their access to information and to label information they create.

We picture typical usage to be scrolling around the displayed map area, adding new "buildings" or "roads," viewing and updating the information about particular objects, and selecting groups of items based on various criteria. The security controls to be enforced are that users can only view objects and object information that they have read access to; they may only alter or remove objects and object information they have write access to; and as the display is scrolled, the overall (least upper bound) security label of the currently viewable region must always be displayed.

4. Data structures and algorithms

In this section, we describe the security-related data structures and algorithms used by our secure GIS. We have divided the handling of security attributes into a generic package, applicable to any application enforcing its own security controls (such as an ordinary database), and a package specific for hierarchical data structures (in our case RECT and PMR quadrees). We will describe each in turn.

4.1 Generic security attribute handling

The generic package organizes subject and object security attributes to optimize four basic operations: creation of new objects, access of subjects to objects, and determining the greatest lower bound and least upper bound of the security attributes of pairs of objects. The generic package has been organized to be portable, and has been ported to several varieties of "secure" UNIX with little problem.

Security attributes for GIS objects are contained in blocks which are only referenced indirectly by code outside the security module. All security blocks with the same attributes (ownership, access bits, label) share the same space. (Since the system can track individual labels for every object, a single security label suffices; there is no need for separate sensitivity and information labels as in the Compartmented Mode Workstation.²) A reference count is used to determine whether to reallocate the blocks on change.

GIS subjects also have object security blocks associated with them, which indicate what attributes should be applied to new GIS objects they create. Hence at object creation time, only a reference count increment and pointer copy are needed to obtain the correct security attributes. In addition to this default security attribute template, subjects also have two other items associated with them: their own security characteristics, and a cache of previous access decisions. The cache is primed with the subject's access rights to its default security object template (which for a "sensible" subject should grant all access).

The subject and object structures are organized into several tables. First of all, both subject and object structures are kept in hash lists for lookup on new subject and object creation. (This is needed because all blocks with the same characteristics share the same space, both to save space and to increase the effectiveness of the access caching scheme described below.) Secondly, each subject has an object access cache list. (This speeds access decisions; after initial setup, most access checks need merely check the access cache.) Finally, the object tables are cross-indexed by combinations (LUB and GLB), with the combination information having its own hashed lookup scheme. (This speeds LUB and GLB determination; again after some initial setup, most LUB and GLB determination can be done by checking the appropriate cached value.)

The access cache is maintained as a self-organizing list, with by-one promotions whenever a selected entry exceeds the usage count of its predecessor by two. (This test helps prevent thrashing in the list with alternate accesses.) The usage counts are decayed exponentially over time to de-emphasize older usage patterns. Most subjects, then, will have the entry for their most commonly-accessed objects (presumably the ones they create) at the beginning of the cache, so that only a single pointer comparison will be required to grant them access to such objects (or other objects with the same characteristics).

One approach to handling the LUB/GLB tables would be to use a sparse matrix representation to hold computed values. We decided instead to use a hashing scheme, similar to that used for security attribute lookup. The hash values used for the LUB/GLB tables are calculated by interleaving the low order bits of the (precalculated) hash values of the two objects. This is simple to calculate, but ensures the hash values are evenly spread over the available range.

4.2 Hierarchical security attribute handling

Implementing the desired security controls on the GIS database requires two region-based operations. The first of these is filtering by access rights: given a rectangular area, we want to determine quickly which portions are accessible by the current user. The second of these is labeling: given a rectangular area, we want to determine quickly the least upper bound of the information labels of all the (accessible) items in the area.

Both of these operations are essentially range queries over a window. For example, a read access query asks, "what items in this area are dominated by my security attributes?" For the labeling query, since in practice we always have an existing label to compare against (for example, the label of the old region, before the new portion is scrolled in), the query is of the form "are there items in this area not bounded by the existing label?"

We handle both queries by a method inspired by the range priority tree.⁶ Attached to each node in the RECT and PMR quadtrees are two labels, the least upper bound and greatest lower bound of the labels assigned to the objects (rectangles or line segments) in the subtree branching from that node.

For a RECT quadtree leaf, the LUB and GLB are both equal to the security attributes of the single rectangle object attached to the leaf. For a PMR quadtree leaf, since several objects may be attached to the same leaf, the LUB and GLB may differ; they are bounds on the entire list of line segments attached to the leaf. Currently we only keep LUB and GLB information for the

security attributes of the geometric objects themselves, not that of their associated information, since we have no region-based queries which involve the associated information.

It is a quite straightforward process to update the tree labels when adding and deleting objects. On adding, we need merely float and "defloat" the LUB and GLB labels for each node we encounter in descending the tree by the label of the object we are adding. (New non-null leaf nodes of course receive the label of the attached object; null leaves ("buds") implicitly have null labels.)

Deletion is slightly more complicated, just as for a normal quadtree. We descend the tree recursively to each leaf which has the object attached, then collapse the tree and update labels as required in coming back up. In the worst case, label changes will propagate all the way to the root of the tree. However, since at each node the new LUB and GLB are immediately determinable from those of its four children, the incremental work required is minimal; there is no need to descend into other parts of the tree. Also, to avoid unnecessary recalculation, we return at each stage an indication (a bitmask) saying whether the subtree's LUB or GLB has changed, or whether the subtree has been converted to a leaf or bud. If any of these have not changed for all four children, they need not be examined again at this or any higher level.

Window queries use the node labels as hints on whether it is necessary to descend the tree further. The basic (read) access control algorithm for window queries on RECT quadtrees is given below. (For the purpose of the discussion here, we have presented the access algorithm in isolation. In actuality, this algorithm is embedded in the window query code, so that both access and window checks ($\text{child} \cap \text{window} \neq \emptyset$) must pass in order to descend into the subtree.) Here we have the algorithm return an "accessible subtree."

Algorithm RECTreadaccess (subject, root)

Begin

```

    foreach child[i] in children (root) do
        if (empty (child[i])) /* Quadrant is empty */
            returntree[i] := NULL;
        else if (terminal(child[i])) /*Rectangle*/
            if (readaccess(subject, secattributes(rectangle)) )
                returntree[i] := rectangle;
            else
                returntree[i] := NULL;
        fi
        else if (readaccess(subject, LUB(child[i])) )
            returntree[i] := child[i];
        else if (  $\neg$  readaccess(subject, GLB(child[i])) )
            returntree[i] := NULL;
        else
            returntree[i] := RECTreadaccess(subject, child[i]);
        fi;
    od;
    return returntree;

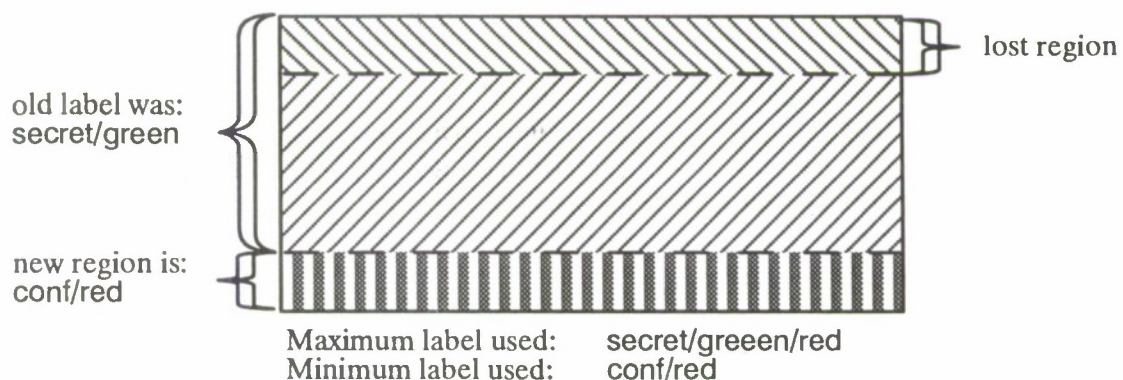
```

End;

The algorithm for PMR quadtrees is basically the same, save when leaf nodes are reached, the list may need to be searched through to determine accessible elements. Determining write access (for "write" in the sense of modify; there are no controls on adding new objects) is similar, using the GLB and LUB information to screen out non-writeable regions.

Labeling queries are slightly different. Here we are scanning a region, attempting to determine the overall label of the items in that region. At any point in the scan, we have two labels, a maximum (bounding) label and a minimum (actual) label. The maximum label is determined before the query is processed; for an isolated query, it could be chosen as the LUB of the smallest quad which contains the entire region. More typically, we are updating an existing label, for example because we have deleted an object. In this case, the previous actual label then represents the new maximum label. The minimum label at any point is the LUB of all we have scanned so far. In an updating situation, where we are adding a new object, the minimum may be chosen to be the LUB of the previous actual label and the new object label.

To handle scrolling, which effectively means the deletion of one (small) region and the addition of another (equally small) region to the window area, we first determine the actual label of the newly added region, then use the LUB of this label with the old existing actual label as the maximum label for the new region. The added region label may be used as the initial minimum:



Here is the basic labeling algorithm for RECT quadtrees, where minlabel is a known minimum, and maxlabel is a known maximum:

Algorithm RECTlabel (root, minlabel, maxlabel)

Begin

```

    if (minlabel > LUB(root) ) return minlabel;
    else if (maxlabel = LUB(root) ) return maxlabel;
    else if (terminal(root) ) return secattributes(root);
    else foreach child in children(root) do
        minlabel = max(minlabel, RECTlabel(child,minlabel,maxlabel) );
        if (minlabel = maxlabel) return minlabel;
    fi;
    return minlabel;

```

End;

The minimum label is only updated when we descend all the way to an object which is actually in the region, so some amount of descent is required. However, the two labels do allow some

short-circuiting of the query. Any subtree whose LUB label is bounded by the actual label need not be examined, since it cannot change the actual label. For a window query, if a quad is entirely within the window region, its LUB can be taken as is, since all of the objects in the quad will be within the window. Also, if at any point the actual label becomes equal to the maximum label, no further examination is required and the query can return immediately.

5. Observations

Not much can be said about the worst-case performance of these algorithms. In the worst case, when objects with all possible security characteristics are evenly interspersed, the GLBs will mostly be “all read/system low” and the LUBs “no one read/system high,” so they will provide no help whatsoever in shortening queries. (Though at least the quadtree structure will cut down the region that needs to be checked.) However, in the expected case where objects with similar labels cluster together, the tree labels should be quite effective. With a high degree of clustering, access and label checks can run in $O(\log n)$ time or better, depending on the degree of descent required, where n is the number of leaf nodes in the quadtree.

In practice, we have not noticed any performance degradation in, for example, scrolling behavior for the secure quadtree with access control and labeling vs. an ordinary quadtree. An earlier implementation used for handling labeled text kept an array of labels used mainly as a queue corresponding to the row ordering of the space.⁷ This worked well for additions and deletions at either end, and for vertical scrolling, but not as well in general. By contrast, the labeled quadtree works relatively well in all cases.

6. Future work

The work so far has demonstrated the feasibility of region-based security attributes in a spatial database. In future work we intend to expand the scope to deal with a wider range of queries, and to implement similar fast access/label algorithms for other data types, including text.

7. References

- 1 Discussion on RISKS Digest, January 1991.
- 2 J.P.L. Woodward, *Security Requirements for System High and Compartmented Mode Workstations*, DIA report DRS-2600-5502-87 (REV 1), 1987.
- 3 *Trusted Computer System Evaluation Criteria*, DoD 5200.28-STD, 1985.
- 4 Mark E. Carson, Janet A. Cugini. “An X11-based Multilevel Window System Architecture,” *Autumn 1990 EUUG Technical Conference Proceedings*, Nice, France.
- 5 C.D. Tomlin, *Geographic Information Systems and Cartographic Modeling*, Prentice-Hall, Englewood Cliffs, NJ, 1990.
- 6 H. Samet, *The Design and Analysis of Spatial Data Structures*, Addison-Wesley, Reading, MA, 1990.
- 7 Mark E. Carson, Wen-Der Jiang, Jeremy G. Liang, Gary L. Luckenbaugh, Debra H. Yakov. “Secure Window Systems for UNIX,” *Winter 1989 USENIX Technical Conference Proceedings*, San Diego, California.

Security Issues on Distributed System Applications

Chii-Ren Tsai

Citicorp International Communications, Inc.
Citicorp Global Information Network

Abstract

Several security issues considered during the design of a prototype of distributed system and security management (DSSM) [Tsai 92] are addressed. The prototype of DSSM, which consists of distributed SMIT¹ (System Management Interface Tool), distributed audit [Tsai 90] and access control list (ACL) management for AIX¹ systems, has been implemented on the RISC System/6000¹ running AIX version 3 with an experimental secure remote procedure call (RPC) mechanism [Tsai 91] based on Network Computing System² (NCS) [Dineen 87] and Kerberos³ [Steiner 88]. An X³/Motif⁴-based user interface is provided for DSSM.

1. Introduction

A distributed application is an application of the client-server model. Because of the simplicity of its programming, the remote procedure call (RPC) mechanism is commonly used in distributed applications for the communication among clients and servers. Vanilla RPC mechanisms might not support security features such as data confidentiality (secrecy), data integrity, authentication and access control. To provide a secure RPC mechanism, we have combined the services of the MIT Kerberos authentication protocol with vanilla RPCs of Network Computing System (NCS). Based on the secure RPCs, we have implemented DSSM.

DSSM consists of a distributed-system management subsystem, a distributed audit subsystem and an access control list (ACL) management subsystem.

The prototype of DSSM was performed when the author was with VDG, Inc. under contract to IBM. The work described herein only contains the author's perspectives. It does not describe, imply or represent any IBM products. Also, it does not represent any opinions of Citicorp International Communications, Inc.

- 1 AIX, SMIT, and RISC System/6000 are trademarks of International Business Machines Corporation.
- 2 NCS is a trademark of Hewlett-Packard Corporation.
- 3 Kerberos and X window system are trademarks of the Massachusetts Institute of Technology.
- 4 Motif is a trademark of Open Software Foundation.
- 5 NFS is a trademark of Sun Microsystems, Inc.

tem. The distributed-system management subsystem is an extension of AIX SMIT, which can be used to handle various system management tasks from system configuration to network management. The distributed audit subsystem can invoke and revoke auditing for remote hosts, locate the *audit trail server* to which audit trails are transferred, perform audit system management, such as dynamically adding or deleting audit events on a per-user, per-group, or per-system basis and querying the audit status of remote hosts, and trace audit trails. The ACL management subsystem can browse and change ACLs on a per-file, per-directory or per-application basis.

In this paper, we highlight distributed SMIT, ACL management and the distributed audit mechanism. We also present security issues considered during the design of DSSM.

2. Overview of DSSM

DSSM provides a Motif-based interface for the central administrator to manage distributed audit, distributed SMIT and ACLs in a distributed environment. An instance of the interface with seven RISC System/6000s distributed on two Ethernet and two token ring networks is shown in Figure 1. There are two sets of main selection menus. The set of menus that contains "ON," "OFF," "CONFIG," and "TRACE" buttons is designed for distributed audit, the other for distributed SMIT and ACL management. Each main selection button is associated with a pull-down list of hosts which are controlled by the *xadministrator*. Each host selection may lead to subsequent submenus or dialog boxes.

The *xadministrator* can turn on/off auditing of any or all hosts through "ON/OFF" buttons. The "CONFIG" button can be used to configure (1) audit trail servers, (2) the high-water-marks of audit trail size, the local audit trail filesystem and the central audit trail filesystem, and (3) audit classes on a per-user, per-group or per-system basis, where an *audit class* is defined as a subset of audit events. Consequently, the *xadministrator* can instruct the audit subsystem to collect the audit records of certain

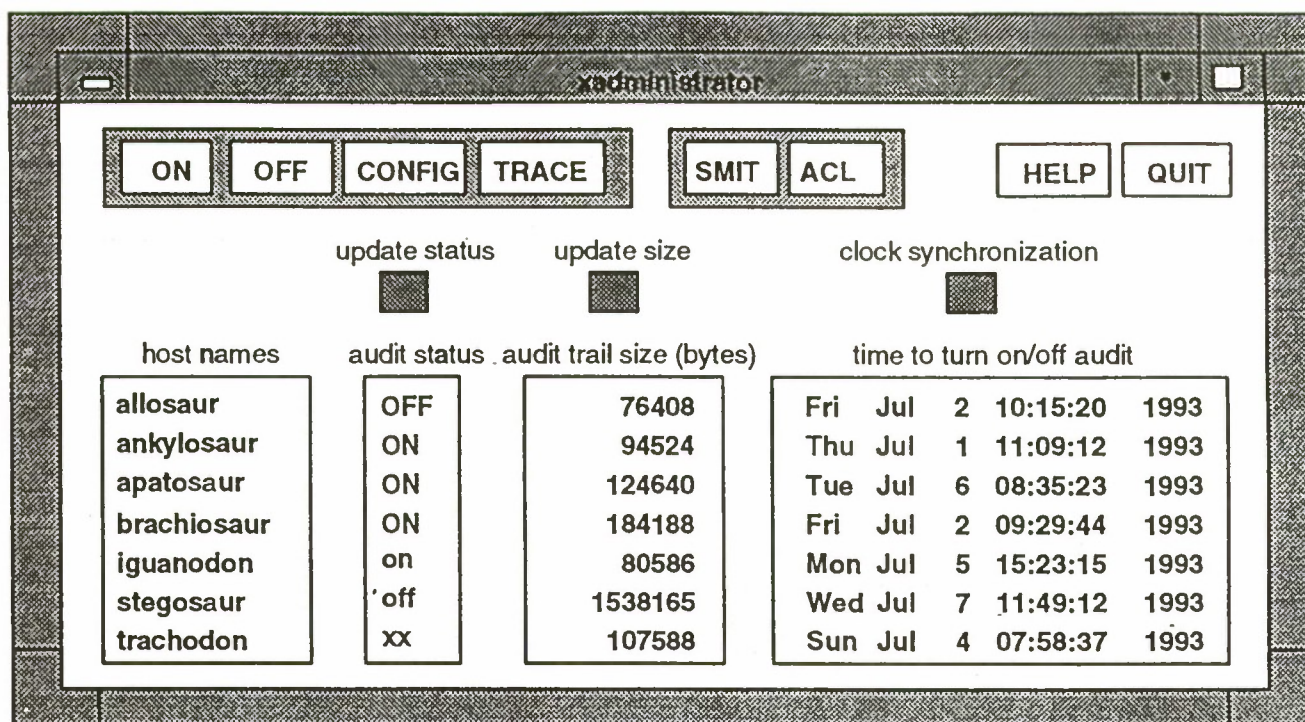


Figure 1. The Central Administrator Interface

audit events for a specific user, or the users of a specific group, or all the users in the system. The function of the "TRACE" button is to trace an audit trail or selected audit trails through a query dialog. The four panels under the main menus show host names, audit status, audit trail sizes, and the time the auditing was turned on/off, respectively. Audit status and audit trail sizes are updated periodically. The buttons above those two panels are used to instantly update the data in them. The button above the time panel can be used to show system-clock discrepancies and to synchronize each host's clock with that of the local host. The purpose of clock synchronization is to satisfy the Kerberos requirement for loosely synchronized clocks and to maintain the consistency of the timestamps of audit records to allow accurate tracing of users' activities.

2.1 Distributed SMIT

The AIX SMIT integrates all system management functions in a single tool, which can be used to interactively manage software installation, devices, physical and logical storage, user accounts, communications applications and services, the spooler, resource scheduling, system environment and processes, and applications. SMIT provides a hierarchical screen structure. Users are guided through the use of menus and dialogs to run system management commands. The data objects managed by

SMIT are handled by the Object Data Manager (ODM) [IBM90], which is a data manager intended for the storage of system data. System data managed by ODM include devices configuration information, display information for SMIT, vital product data for installation and update procedures, communication configuration information, and system resource information. All the data are stored either in the `/etc/objrepos` directory or the directory specified by the `ODMDIR` environment variable. SMIT generates and updates two log files, `smit.log` and `smit.script`, for each user. The `smit.log` file keeps additional detailed information that can be used by programmers to extend the SMIT system, while the `smit.script` file records shell-script commands used to perform system management functions. These two files can be used as audit trails.

Distributed SMIT is an administrative function that can provide the SMIT interface and communicate with each system's ODM through a daemon, `dsmitd`, to retrieve and modify system data or to execute configuration functions. The distributed-system administrator interacts with `dsmitd` via secure remote procedure calls. The mechanism of distributed SMIT is shown in Figure 2.

2.2 ACL Management

An object's Access Control List defines the access authorization of the subjects in the system. AIX Ver-

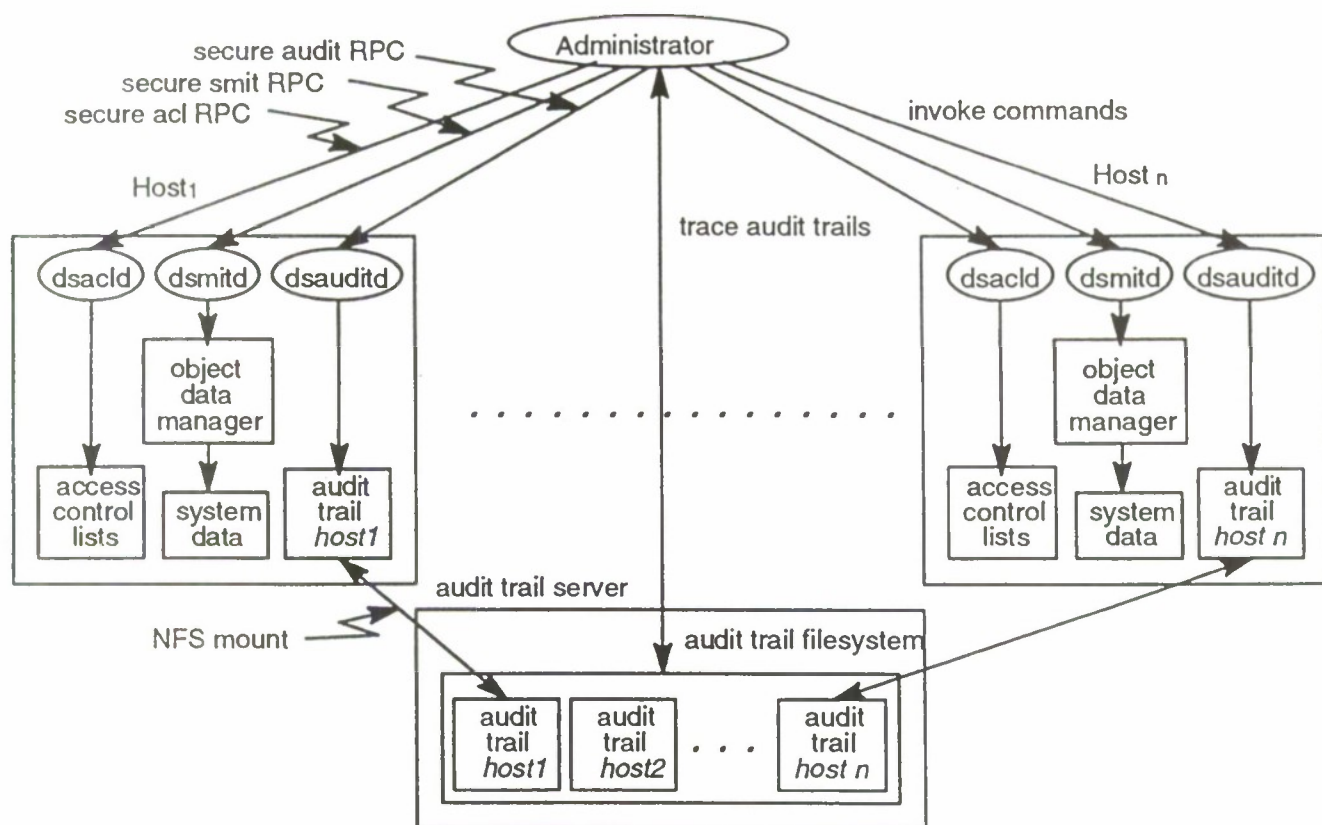


Figure 2. The Mechanisms of Distributed SMIT, ACL and Distributed Audit

sion 3 implements ACLs on objects such as files, directories, named pipes, message queues, shared memory segments and semaphores, so that users can browse and change the ACL of an object through system calls. The structure of AIX ACLs is shown in Figure 3. Each ACL may consists of *base permissions* and *extended permissions*. Base permissions, which can be modified by **chmod**, contain the **setuid**, **setgid** and **save text** bits, and three sets of access modes for owner, group and others, respectively. Each set of access modes consists of read, write, and execute/search permission bits. Extended permissions consist of an unordered list of Access Control Entries (ACE) [IBM90]. Each ACE contains a list of identifiers, the type of the ACE and a set of access modes. The ACE types include *permit*, *deny* and *specify*. The *permit* and *deny* types indicate that the specified set of access modes is granted or denied, respectively; the *specify* type means that only the specified set of access modes is granted and others are restricted. The type of an identifier is either **USER** or **GROUP**.

Distributed ACL management is the centralized control of ACLs in a distributed system, so that the central ACL administrator can manipulate the ACLs

of objects through a daemon, **dsacld**. To simplify ACL management, ACLs can be changed on a per-file, per-directory or per-application basis, which means that the ACL administrator can change the ACL of a file, the ACLs of all files in a directory, or the ACLs of files categorized to the same application. The mechanism of ACL management is shown in Figure 2. A snapshot of the ACL interface is shown in Figure 4.

2.3. Distributed Audit Mechanism

The distributed audit mechanism discussed herein is another central administrative function that can perform audit system management, invoke/revoke auditing for each host, instruct each host to transfer its audit trail to a specific site called an *audit trail server*, and trace audit trails [Tsai 90, Tsai 91]. As shown in Figure 2, the central audit administrator invokes/revokes auditing by using the secure RPC mechanism. Each host mounts the audit trail filesystem from the audit trail server over a local directory by utilizing NFS⁵, so that the host's audit records are compressed and stored in a file, called the audit trail of the host. Consequently, audit trails are collected in the audit trail filesystem of the audit trail server, and the audit administrator can manage

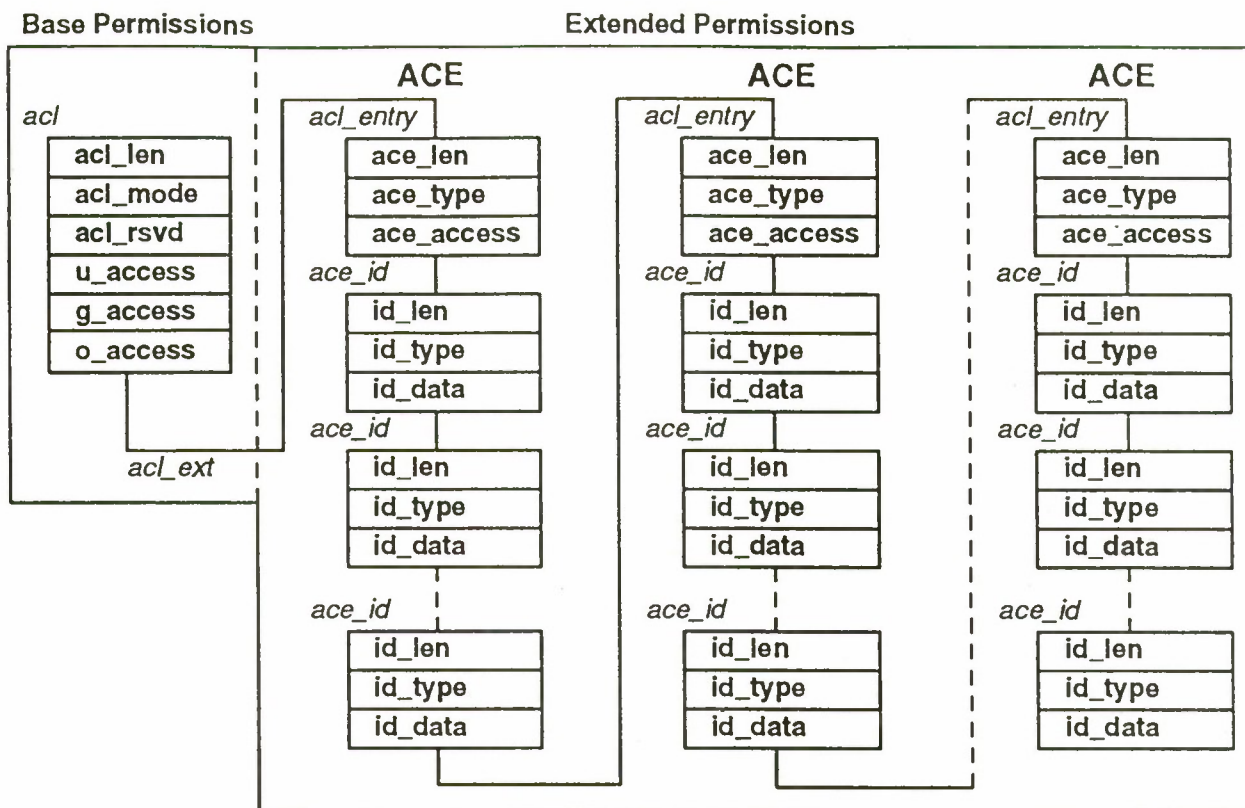


Figure 3. The Structure of AIX Access Control Lists [Tsai 92]

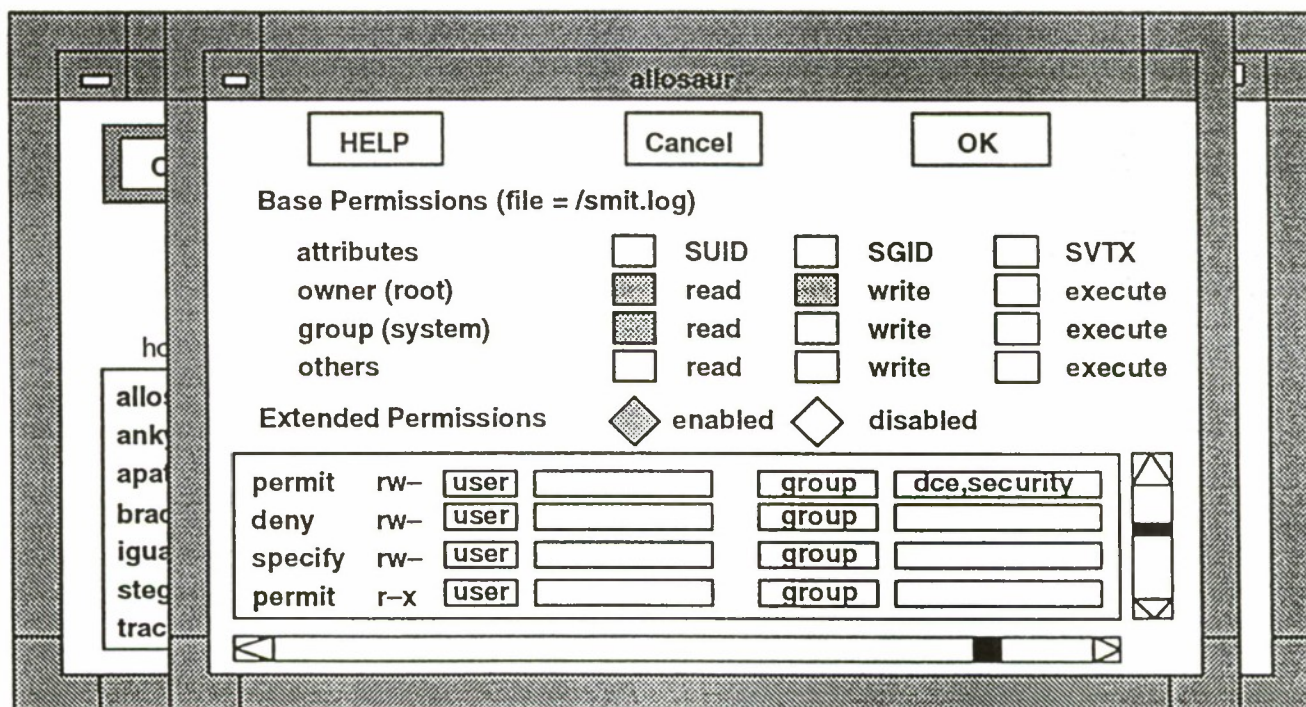


Figure 4. A Snapshot of ACL Interface

3. Security Issues

- (1). authentication between the DSSA and each server
- (2). data confidentiality (secrecy)
- (3). data integrity
- (4). access control for RPCs
- (5). access control for audit trails
- (6). separation of roles

3.1. RPC-Related Security Issues

Figure 5, the secure RPC mechanism uses the Kerberos authentication protocol, encrypts input and returned data, provides data encryption standard (DES) cipher-block-chaining (CBC) checksums for data, and performs access checking against the caller's identity. Therefore, it provides the capability of authentication, data secrecy and integrity, and access control. All security enhancements, including authentication, data encryption and decryption, and access control, are implemented at the client and server stubs, so that the interfaces of secure RPCs are the same as those of vanilla RPCs, and RPC runtime is left unchanged.

3.2. Access Control for Audit Trails

The audit trail of each system is transferred to the central audit trail filesystem through NFS. In other words, the central audit trail filesystem of the audit

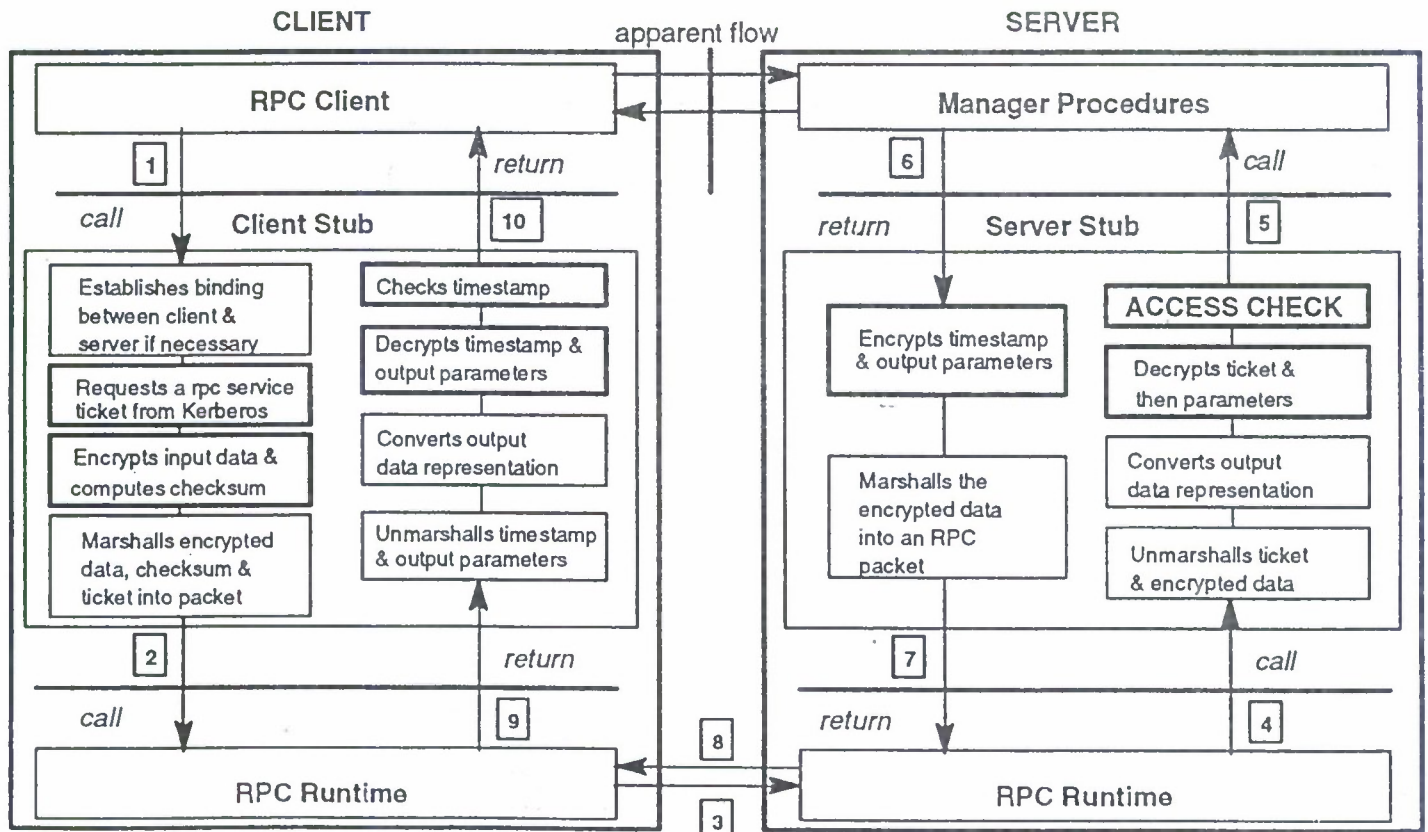


Figure 5. Secure RPC Paradigm [Tsai 91]

trail server is mounted to the audit directory of each client. Consequently, audit records of each client are appended to its audit trail file in the audit directory. Although secure NFS provides authentication between the server and clients, it does not imply that there is no access control problem. Unlike access control on secure RPCs that use Kerberos principals as access identities, access control on audit trails relies on the user ID mapping from one system to another. If a distributed system is configured as a single-system image, user IDs are the same for each user on any system. Access control can be enforced easily. On the other hand, if a user has different IDs on different systems, user ID translation must be done to avoid access violations. Each audit trail is written by the distributed daemon which must be running as the root since only the root can mount a remote filesystem over a local directory. The audit trail filesystem must be protected from being modified or deleted by any users except the root and the auditor. If an auditor that is different from the root is specified, its user ID must be the same on all systems. The ACL of each audit trail can be modified to grant the auditor access permissions.

3.3. Separation of Roles

The central administrator interface can be used to manage distributed SMIT, ACL and distributed audit. Nevertheless, we may want to split the central administrative role into several administrative roles such as the system administrator, the security officer and the auditor. To resolve this issue, we grant the access permissions of these subsystems to appropriate administrative users. Even if these administrative users may have the same user interface, they can perform different sets of functions. For example, the auditor can only perform audit functions, namely ON, OFF, CONFIG and TRACE shown in Figure 1, while the system administrator and the security officer can invoke SMIT and ACL, respectively.

4. Conclusions

Security problems become important issues in distributed system applications, especially in an open environment. If security problems are not resolved, distributed system applications cannot function properly. The design of DSSM, which is aimed at reducing management efforts while still maintaining security, can serve as an example for designing a distributed system application.

Acknowledgments

The author is grateful to Professor Virgil D. Gligor of University of Maryland and Janet A. Cugini of NIST for their insightful suggestions and discussions on this paper.

References

- [Dineen 87] Dineen, T. H., et al., "The Network Computing Architecture and System: An Environment for Developing Distributed Applications," in Proceedings of the 1987 Summer USENIX Conference, Phoenix, Arizona, pp. 385-398, June 1987.
- [IBM 90] IBM Corporation, *AIX Version 3 for RISC System/6000 : General Concepts and Procedures*, 1990.
- [Steiner 88] Steiner, J. G., C. Newman, and J. I. Schiller, "Kerberos: An Authentication Server for Open Network Systems," in Proceedings of the 1988 Winter USENIX Conference, Dallas, Texas, pp. 191-202, Feb. 1988.
- [Tsai 90] Tsai, C. R., V. D. Gligor and M. S. Hecht, "Potential Pitfalls of a Distributed Audit Mechanism," in Proceedings of the 1990 EurOpen Autumn Conference, Nice, France, pp. 91-103, October 1990, also available as *IBM Gaithersburg Technical Report 85.0098*, October 1990.
- [Tsai 91] Tsai, C. R. and V. D. Gligor, "Distributed Audit with Secure Remote Procedure Calls," in Proceedings of the 1991 IEEE International Carnahan Conference on Security Technology, Taipei, Taiwan, pp. 154-160, October 1991.
- [Tsai 92] Tsai, C. R. and V. D. Gligor, "Distributed System and Security Management with Centralized Control," in Proceedings of the 1992 EurOpen/USENIX Conference, Jersey, Channel Islands, pp. 137-146, April 1992, also available as *IBM Gaithersburg Technical Report 85.0155*, June 1992.

Security Services for Multimedia Conferencing

Stuart G. Stubblebine

USC Information Sciences Institute
4676 Admiralty Way
Marina del Rey, CA 90292
stubblebine@isi.edu

Abstract

Multimedia conferencing is the use of mixed media such as real-time audio, video, and groupware for group tele-collaboration. Multimedia conferencing may well become as widespread an application on network computers as electronic mail. The demand for security in multimedia conferencing is high because the users' expectation of being monitored is high. In this paper we discuss security services for protecting multimedia conferencing, how well these services scale to large conferences, and what security infrastructure is needed.

1. Introduction

Electronic mail is perhaps the most popular application on networked computers. Several standards for secure electronic mail have evolved, including the Internet's Privacy-enhanced Electronic Mail [L93] and the U.S. government's Pre-Message Security Protocol [B93]. With the exception of electronic mail systems running on private networks protected by link encryption, secure electronic mail has been slow to mature. The slow growth is partially attributed to the difficulty of developing an acceptable infrastructure for public key certificate based authentication [X.509]. It is also attributed to users' lack of awareness to the susceptibility of electronic mail to wire-tapping attacks.

Multimedia conferencing, the combination of real-time packet audio, video, and groupware multiway tele-collaborations, is likely to become as widespread an application on network computers as electronic mail. As such, significant research and development is needed to integrate security into the design of teleconferencing systems. The growing demand for teleconferencing is demonstrated by the regular multicasts of audio and video from the Internet Engineering Task Force (IETF) meetings over the MBONE, a multicast capable portion of the Internet [CD92]. The most recent teleconference included 339 sites in 14 countries. There is a growing demand for security as participants are becoming aware of how easy it is for others to monitor their conferences. This is because architectures that scale well to support large conferences do not restrict the distribution of conference packets. That is, conference sessions are announced via network multicast to a session directory service, and users participate in the conference by simply joining the multicast address associated with the conference. Private announcements via electronic mail or other means offer only limited protection since an eavesdropper can still scan the multicast address space to detect conferences in progress.

In the next section, we discuss security services for protecting multimedia conferencing. We assume the use of trusted computing bases and emphasize the requirements of communication security. In section 3, we discuss the scalability of these security services to large conferences. In section 4, we discuss what security infrastructure is needed to provide these services.

2. Security Services for Multimedia Conferencing

A multimedia conference may consist of several media flows, for example audio, video and groupware application flows. (A groupware application typically makes use of a shared

workspace. An example of a groupware application is a shared whiteboard.) The protection required for each data flow in the conference may vary. For example, it may be necessary to protect the confidentiality and authenticity of an audio stream. However, we may only need to protect the authenticity of the corresponding video stream. Furthermore, the rights of a conference attendee may vary according to the particular data flow. For example, designated attendees may be able to present on an audio flow while others can only receive the audio flow.

The security requirements for protecting multimedia conferences depend on the role of the authorized conference attendee. We will say a conference attendee is a *presenter* when the attendee generates information that is sent to other attendees, whereas a *recipient* receives information. The security services desired by presenters and recipients vary depending on the nature of the conference. A list of some of the most common services from which to select follows. (Note that data confidentiality usually takes precedence over other security requirements.)

Presenter

Data Confidentiality. The presenter's data is protected against unauthorized disclosure.

Identity Confidentiality. The identity of the presenter is protected against unauthorized disclosure. Identity confidentiality can be maintained with respect to all others (as required in voting protocols) or only with respect to non-participants of the conference (as might be required between a reporter and a source or between a buyer and a seller). Traffic analysis must be considered when protecting the identity of the presenter.

Non-repudiation of Receipt. Unforgeable proof of receipt of data by the recipient. Proof of receipt, including time of receipt, can be proven to a third party such as a court of law.

Repudiation of Transmission. Information presented in a conference can not be substantiated.¹ Laws of many countries preclude the contents of a private conversation to be used against them in a court of law. Therefore, parties to a conference have an expectation that the information they exchange will not be used against them [R93]. (Note that identity confidentiality may provide repudiation of transmission.)

Availability to Present. A presenter is not denied from contributing to a conference in accordance with the conference policy.

Recipient

Identity Confidentiality. The identity of a conference recipient is protected against unauthorized disclosure.

Origin Authentication. A recipient can authenticate the origin of data.

Data Integrity. The recipient can detect an unauthorized modification of data. Strict protection against message-stream modification may be required for some flows (e.g. groupware application flows). Weaker requirements such as a hybrid connection/connectionless integrity service may be adequate for real-time video streams. In the hybrid integrity service, the integrity of individual packets is protected; however, some packet loss in transit is acceptable provided the packets are played back in order.

Non-repudiation of Transmission. Unforgeable proof of shipment. Proof of shipment, including time of shipment, can be proven to a third party. Non-repudiation of a sequence of information may need to be shown.

¹ The need for repudiation of data is attributed to Michael Roe and Russ Housely.

Availability to Receive. A recipient is not denied from receiving information in accordance with the conference policy.

Some of the preceding security services can also be required for protecting flows that control site-specific media devices. Controlling access to conferences is also an important security requirement. We can specify conference access policies using the access matrix model [HRU76].

Access Control

The access control matrix specifies the *rights* a *subject* may have with respect to an *object*. A subject in a conference consists of a conference site or attendee at a site. Objects can be a data flow or a media device. (An example of a media device is a camera or a microphone.) The rights a subject may have with respect to an object include writing (e.g., presenting information to the flow or controlling a media device), reading (e.g., receiving a flow) and owning the object. A subject that owns an object may participate in assigning rights to other subjects (e.g., add read and write rights to an attendee for a particular flow). Rules for assigning rights may be non-trivial. For example, attendees working for different companies may not be authorized to confer between themselves although they may be allowed to join in the audience. Other complexities may exist in the different models for admitting attendees to a flow. For example, assigning read and write rights might require a consensus from multiple conference owners.

3. Scalability of Security Services

We now discuss how well today's security technology scales in providing the various security services. The design of a protection architecture depends on the size of a conference. Conferences are sometimes described by the number of attendees [S93]. A small conference is highly interactive and consists of only a few participants. A medium sized conference might consist of hundreds or thousands of participants and is typical of an interactive seminar with many recipients but relatively few presenters. A large conference consist of hundreds of thousands or millions of participants; an example is a pay-for-view television broadcast.

Security services that require unique processing of information for each recipient do not scale well to medium and large conferences. This is particularly true for multimedia conferences imposing real-time communication constraints (e.g., bounded communication delays). Distribution of session keys is an example of a per-recipient task. This distribution is required for providing data confidentiality using symmetric cryptography. Schemes for distributing keys *before* a conference can help alleviate this problem. However, these schemes have limitations in responding to changes in access permissions during a conference. For example, when an conference moves to executive mode (i.e., a more restrictive session), revoking read privileges requires a new session key to be distributed to the new set of authorized participants.

Session keys may also be used to perform data origin authentication. However, the use of session keys for data origin authentication has an additional scaling problem since representations of authentication digests need to be processed on a per-recipient basis for each packet sent. (For example, authentication digests may be encrypted using symmetric cryptography under a secret key shared between the sender and each recipient.) This problem can be alleviated by using asymmetric cryptography when preparing an authentication digest.² The task of key distribution is effectively distributed to the recipients. Each recipient obtains and verifies the presenter's public key.

² The designer must choose a redundancy function that has the mathematical properties sufficient to support the integrity policy [SG93].

4. Security Infrastructure for Multimedia Conferencing

Significant research and development is needed to create the infrastructure for wide scale deployment of secure multimedia conferencing over unprotected packet-switched networks like the Internet and the National Research and Education Network (NREN). Much of the foundation for the security infrastructure either exists or is being developed to support other applications. Some of the components needed to protect multimedia conferencing are as follows:

Trusted Computing Bases. The integrity of conferencing applications is protected by building on trusted operating systems and trusted hardware platforms.

Key Management. A key management infrastructure is needed that will enable the creation of trusted paths between heterogeneous key management systems and heterogeneous trust hierarchies/models.

Conference Control Servers. Conference control servers are needed that will enforce access control policies to information flows and media devices.

Secure Communication Protocols. Communication protocols are needed that will support the protection requirements for real-time communications. For example, work is in progress to augment an experimental Real-time Transport Protocol (RTP) [SC93] to support security features that can be used to satisfy the requirements of a hybrid connection/connectionless integrity service. Secure communication protocols at other layers (e.g. network layer) might also be important.

Network Service Guarantees. Network service providers need to provide guarantees for use of bandwidth. This is because real-time traffic is highly susceptible to communication delays leading to a denial of service. Common carriers operating in a multi-vendor multi-user packet switched environment will use network layer security to enforce service guarantees.

Identity Confidentiality Servers. Identity confidentiality servers are needed to protect the identity of conference participants from traffic analysis. As with all security services, the value of using the service should justify its cost.

5. Summary

The demand for security services in multimedia conferencing will rapidly grow as multimedia conferencing becomes widespread. We discussed security services for protecting multimedia conferencing, scalability issues for implementing these services and some of the security infrastructure that is needed to meet the demands for protecting conferences.

Acknowledgements

We thank Celeste Anderson, Stephen Casner, Kathleen Madden, Clifford Neuman, Michael Roe, Eve Schooler, and Joe Touch for their helpful discussions and reviews.

REFERENCES

- [B93] B. Bialick, "Pre-Message Security Protocol (PMSP) Overview", Federal Digital Signature Symposium, National Institute of Standards and Technology, Gaithersburg, MD, February 1993.
- [CD92] S. Casner and S. Deering, "First IETF Internet Audiocast," *ACM SIGCOMM Computer Communications Review*, Vol. 22, No. 3, July 1992.
- [HRU76] M. Harrison, W. Ruzzo, and J. Ullman. "Protection in operating systems," *Comm. ACM* 19, No. 8 (Aug. 1976), 461-471.
- [L93] J. Linn, "Privacy Enhancement for Internet Electronic Mail: Part I -- Message Encipherment and Authentication Procedures," Internet Working Group, RFC-1421, February 1993.

- [R93] Michael Roe, "Real-time Communications", PSRG Memo, April, 1993.
- [S93] E. Schooler, "The Impact of Scaling on a Multimedia Connection Architecture", to appear in the *ACM Journal of Multimedia Systems*, 1993.
- [SC93] H. Schulzrinne and S. Casner, "A Transport Protocol for Real-Time Applications", Working Draft, Internet Engineering Task Force, Audio-Video Transport WG, May 1993.
- [SCP91] E. Schooler, S. Casner, J. Postel, "Multimedia Conferencing: Has it come of age?," Proceedings 24th Hawaii International Conference on System Sciences, Vol.3, pp.707-716 (January 1991).
- [SG93] S. Stubblebine and V. Gligor, "Protocol Design for Integrity Protection," Proc. IEEE 1993 Research in Security and Privacy, IEEE Computer Society Press, Oakland, California, May 1993.
- [X.509] CCITT Recommendation X.509 (1988), "The Directory - Authentication Framework".

BEST OF NEW SECURITY PARADIGMS II WORKSHOP

Held August 2-5, 1993, in Little Compton, R.I.

Paradigms, fundamental ways of looking at the world, are often unconscious until challenged. This New Security Paradigms Workshop (NSPW) challenges long-held security assumptions. Roshan Thomas and Ravi Sandhu, for example, replace subjects and objects with tasks and activities in distributed applications. Yvo Desmedt completely redefines what a computer is so that security can be built in from the beginning.

Some paradigms are evolutionary. Via "responsibilities", Ros Strens and John Dobson create a requirements language that is meaningful to both users and trusted system designers and can act as a bridge between the two worlds. Kioumars Yazdanian and Frederic Cuppens use a "data neighborhood" concept to prevent inference in databases. Bill Shockley addresses the problem of identifying and authenticating users who have multiple accounts.

Some paradigms are dying. Dixie Baker and Marv Schaefer question the viability of high assurance systems and the feasibility of producing anything truly secure.

Multiple policies and networks are major themes of the NSPW. Jose Gomez develops strategies to handle interactions between multiple policy domains. Hilary Hosmer describes methods for representing non-traditional security policies. James Slack builds a secure object-oriented model with significant network implications. Leonard LaPadula and James Williams extend their model for external consistency to distributed systems.

The best NSPW papers have been brought to the NCSC for discussion before a larger group. A panel of participants summarizes the lessons learned. Finally, a group exercise provides a chance to experience a bit of the New Security Paradigms Workshop itself.

NCSC Program for Best of the New Security Paradigms Workshop

September 22 , 1993

- 2:00-2:05 p.m. "Why New Security Paradigms"
Hilary Hosmer, Data Security, Inc., NSPW Chair
- 2:05-2:10 p.m. "Overview of the Program"
David Bailey, Galaxy Systems, NSPW Program Chair
- 2:10-2:30 p.m. "How Responsibility Modelling Leads to Security Requirements", Ros Strens and Dr. John Dobson,
University of Newcastle upon Tyne

We propose that an organisation be viewed as a network of responsibilities that embody aspects of structure as well as function. Users' real requirements are manifest in the responsibilities they hold.

- 2:30- 2:50 pm Leonard LaPadula of MITRE leads discussion.
- 2:50-3:10 pm "Task-based Authorization: A Paradigm for Flexible and Tailorable Access Control in Distributed Applications",
Roshan Thomas and Ravi Sandhu, George Mason U.

Historically, access control has been couched within the framework of subjects, objects, and rights... Authorization in distributed applications should be seen in terms of tasks/activities rather than individual subjects and objects.

- 3:10-3:30 pm Eric Leighninger of Arca Systems leads discussion.
- 3:30-4:00 pm Break
- 4:00-4:50 p.m. "Summary of the New Security Paradigms Workshop"
James Williams of MITRE, Yvo Desmedt of U. of Wisconsin,
Jose Gomez of Supelec, James Slack of Mankato State U.
and other participants.
- 4:50-5:10 p.m. "Identification and Authentication When Users Have Multiple Accounts", William Shockley, Cyberscape Computer Services

The key idea is to distinguish the authentication function from the identification function.

- 5:10-5:20 p.m. Dr. Dixie Baker of Aerospace leads discussion
- 5:20-5:50 p.m. "New Paradigms Exercise"

How Responsibility Modelling Leads To Security Requirements

Ros Strens and John Dobson

Department of Computer Science,
University of Newcastle upon Tyne
NE1 7RU, UK

ABSTRACT

When a technical system is placed in a social context (a socio-technical system) organisational requirements arise in addition to the functional requirements on the system. Security is a good example of such an organisational requirement. A means of identifying these organisational requirements is needed and also a way of specifying them that is meaningful both to users and systems designers.

This paper proposes that the concept of responsibility fills both these needs. Responsibilities embody requirements in that the responsibility holder needs to do things, needs to know things and needs to record things for subsequent audit. These needs form the basis of a 'need-to-know' security policy. These needs can be interpreted as functional and data requirements on the IT system implementing such a security policy. Responsibilities can thus be used as a boundary object between the worlds of users and designers in a way that is meaningful to both. Furthermore a model of responsibilities describes the context within the organisational structure in which the requirements, including those related to security, arise.

This paper shows how organisational structure may be represented as a network of responsibility relationships, how requirements arise from the discharge of obligations associated with responsibilities, and how these concepts have been applied to the particular example of specifying user requirements for clinical workstations in acute hospitals.

KEYWORDS

Responsibility, obligation, requirements, enterprise modelling, security

1. INTRODUCTION

The first requirement that an organisation will have of a technical system is that it has the functionality necessary to serve the organisation's purposes. This defines the functional requirements on the system. Of equal importance is the need for the system to support those functions in a way which matches the structure, objectives and characteristics of the organisation. These requirements that come out of a technical system being placed in a social context are termed **organisational requirements**. We

shall be particularly concerned in this paper with security as an organisational requirement.

The need to capture and deal with organisational requirements in the system design process has long been recognised, and a number of methods are now in existence to support the handling of such issues in IT systems design, but there is very little evidence that they are widely used. The ORDIT project, an Esprit II project investigating information technology and organisational change, has addressed these problems on the basis of socio-technical systems theory, with its premise that the system contains within it two sets of resources: technical and social (human) resources, and that these are so inter-related that any attempt to optimise only one of these sets of resources may adversely affect the other set so that the resultant utilisation is suboptimal. Design methods appropriate for technical systems cannot simply be applied to socio-technical ones, since equal consideration must be given to both human and technical issues if the design is to meet the real requirements of the organisation and be supportive of people in their work roles. Again, security systems will be considered as socio-technical systems instead of just as technical ones, recognising that security policies and models that have been developed in a purely technical context may not be applicable to the wider context we are here considering.

We therefore need a means by which system developers can recognise organisational requirements such as security properties and specify them in such a way that enables them to envisage and propose solutions to meet the achievement requirements. The problem here is twofold. Firstly there is the problem of how to capture the requirements, some of which may be apparent and easily ascertained, others may be more difficult to elicit, if, for example, they are implicit in the working practices, and others may only arise when design solutions are proposed.

The second problem is that the language of systems designers is suited to technical systems whereas the users' language is appropriate to the organisational context. What is needed is some set of boundary objects where these two worlds can meet. We are proposing that the concept of responsibility is one such boundary object, and that responsibilities may be regarded as the key to understanding requirements in implementable terms in that a responsibility has attributes that can be appreciated in both worlds although the language and implications differ.

We also see the concept of responsibility as being a means of solving our first problem, that of identifying requirements in the first place. We propose that an organisation can be viewed as a network of responsibilities that embody aspects of structure as well as function. The users' real requirements are manifest in the responsibilities they hold in that they have a **need to know** things and a **need to do** things for the proper fulfilment of their responsibilities, and a **need for audit** in order to show how they have fulfilled their responsibilities. A responsibility thereby implies requirements for information, requirements for action and requirements for the recording of history, and, by approaching these requirements through the responsibilities held by users by virtue of their work roles within the organisation, we not only capture the requirements but gain an understanding of the organisational context in which they arise. Note that we are assuming primarily a 'need-to-know' basis for security policies, though our ideas can accommodate other alternative bases for a security policy.

On the designer's side of the boundary, it is clear that the requirements for information and action can be translated into the data and functions that the IT system must provide. Thus the concept of responsibility as a boundary object between users and designers should lead to a better understanding by designers of what the technical system should achieve (rather than how it will do it which is purely within the domain of the designer), and its context of use within the socio-technical system.

The concept of responsibility is also a valuable boundary object between different types of model and between reality and models. By looking at all of the responsibilities held within a work role, we can unify different models of the organisation, such as a process-oriented horizontal view and a management or vertical view, into one responsibility based view.

In the next section the rationale underlying our assertion that responsibilities embody requirements and the context in which those requirements arise in terms of organisational structure is presented, and the final section briefly indicates how these ideas have been applied in the real world to produce a specification of user requirements for an integrated clinical workstation.

2. THE CONCEPT OF RESPONSIBILITY

In the paper delivered at the New Security Paradigms workshop last year, we argued for responsibility being a key issue for security. This section elaborates the notion of responsibility; a subsequent section will relate this elaboration to issues of security.

2.1. The Responsibility Relationship

So far we have spoken of responsibilities held by users as though they are a 'thing' that the user possesses. In fact the holding of a responsibility implies that there is also a giver of that responsibility and therefore the existence of a relationship between the holder and the giver of the responsibility. (From now on we shall refer to the 'people' involved as agents, since an agent can be any size of group from an individual to a department or even a whole organisation.) We therefore define responsibility as a relationship between two agents regarding a specific state of affairs, such that the holder of the responsibility is responsible to the giver of the responsibility, the responsibility principal (Figure 1).

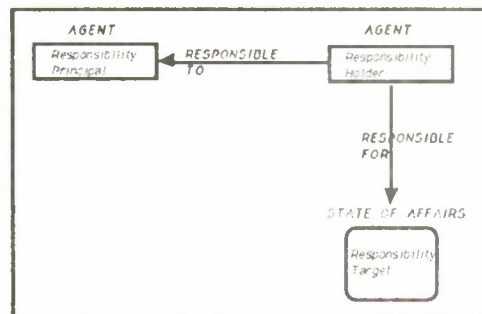


Figure 1. A Responsibility Relationship between Two Agents

The definition of a responsibility consists of:

- a) who is responsible to whom;
- b) the state of affairs for which the responsibility is held;
- c) a list of obligations held by the responsibility holder (how the responsibility can be fulfilled);
- d) the type of responsibility (these include accountability, blameworthiness, legal liability).

2.2. The Relationship between Responsibilities, Obligations and Activities

This brings us to the distinction between responsibilities, obligations and activities. We use these concepts in the sense that agents execute activities in order to discharge obligations imposed on them by virtue of the responsibilities they hold. These obligations are what the agents have to do and effectively describe their 'jobs' or roles. They are the link between their responsibilities and the activities they execute. Another way of describing this relationship is to say that responsibilities tell us *why* agents do something, obligations tell us *what* they do and activities are *how* they do it.

The distinction between responsibilities and obligations is apparent from the words we use: a responsibility is *for* a state of affairs, whereas an obligation is *to do* something that will change or maintain that state of affairs. Thus a set of obligations must be discharged in order to fulfil a responsibility. As such, obligations define how that particular responsibility can be fulfilled. For example a hospital doctor may have responsibility for the medical condition of certain patients. To fulfil this responsibility the doctor must discharge certain obligations such as to diagnose, treat, monitor and/or prescribe.

The distinction between obligations and activities is that obligations define *what* has to be done rather than *how* it is done. Activities are defined as operations that change or maintain the state of the system or affect the outside world. Role holders may (or may not) have a wide choice of activities that discharge the obligations they hold. Consider again the hospital doctor who has an obligation to make a diagnosis. According to circumstances he may choose one or more of several activities such as to examine the patient, order x-rays or do tests.

2.3. Creation of Responsibility Relationships: the Delegation Process

The responsibility relationship implies a structure as, for example, whether a particular responsibility held by a doctor is to the patients, to the employer or to other staff. These responsibility relationships are created when delegation takes place and obligations are transferred from one agent to another. This delegation process will frequently be implicit rather than explicit, and may be used to explain how the hierarchical organisational structure and distribution of responsibilities has come about over time. Our account of the delegation process is based on the view that, because a responsibility is a relationship between two agents, responsibility holders cannot independently transfer their *responsibilities* to other agents, but they can transfer their *obligations*. The result of this process is the establishment of new responsibility relationships between the pairs of agents involved. The original holder becomes the principal of the new responsibility relationship and the receiver of the obligation is the

new responsibility holder. We will now examine this process in a little more detail, since a security policy must have the concepts to permit statements about what happens to capabilities for access to resources associated with obligations in the presence of delegation. It is possible, for example, that as a result of delegation of obligations, an undesirable set of capabilities ends up in the hands of the same roleholder; this would force the re-examination of the desirability of the delegation, and perhaps of the original division of responsibilities (which would have to be solved in the social system, of course, not the technical one).

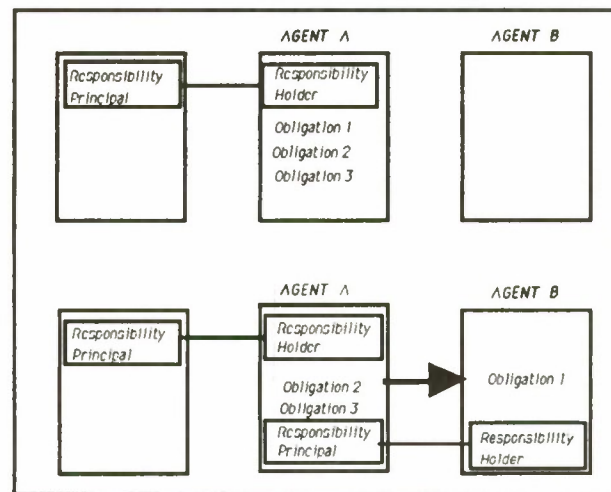


Figure 2. A Responsibility Relationship Created by the Transfer of an Obligation

The top diagram in Figure 2 shows a situation where agent A is the holder of several obligations associated with a responsibility. If an obligation to do something is passed to agent B (lower diagram), agent A still retains the original responsibility since this is not transferable, and we will see in the next section how that responsibility can be fulfilled. Meanwhile agent B has acquired an obligation relating to the state of affairs for which agent A holds responsibility. Agent B also holds responsibility now for that same state of affairs, as well as agent A, because it will be affected when the obligation is discharged. However agent B's responsibility is to agent A who delegated the obligation; in other words a new responsibility relationship has been created between them.

An example of this process is where the first author of a book is responsible to a publisher for the production of a text. The first author retains this responsibility to the publisher even if the obligations to write individual chapters are transferred to other authors. The other authors then acquire responsibility for the writing of their respective chapters, but their responsibility is to the first author and not directly to the publisher.

A chain of responsibility relationships can thus be created as obligations are passed from one agent to another. Within each individual responsibility relationship both agents have a responsibility for the same state of affairs, although their obligations differ.

2.4. Functional and Structural Obligations

The obligations referred to above are functional in nature. They are what agents must do with respect to a state of affairs (e.g. execute activity), in order to fulfil their responsibilities regarding that state of affairs. These we term functional obligations.

We have seen however that when an agent delegates an obligation to another agent, the first agent still retains responsibility for the resulting state of affairs. To fulfil this responsibility the first agent must ensure that the transferred obligation is discharged satisfactorily by the other agent. The first agent thus acquires a new obligation to do whatever is appropriate *with respect to the other agent* in order to fulfil his responsibility, such as directing, supervising, monitoring and suchlike of the other agent. This other agent also acquires an obligation of a complementary nature to be directed, to be supervised or whatever. These we term structural obligations (Figure 3). For example if a director passes an obligation to a manager, the director acquires a structural obligation to direct the manager in the discharging of the transferred obligation, and the manager acquires an obligation to accept direction. Other examples of these structural obligations (e.g. to verify) occur in the context of auditability obligations. Again the structural obligations may be implicit in the hierarchical structure of the organisation rather than a result of explicit delegation.

To summarise, we have shown that everything that a responsibility holder must do, whether with respect to a state of affairs or to another agent, is represented by the functional and structural obligations held.

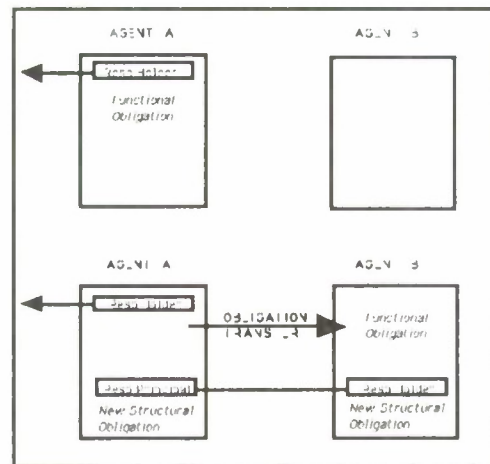


Figure 3. New Structural Obligations created by the Transfer of an Obligation

2.5. Responsibilities Embody Requirements

We have distinguished between functional (process) and structural (organisational) obligations solely to show how the distribution of function and the organisational structure are embodied in the network of responsibility relationships. From the point of view of defining requirements we only need to know what the agents need to do and the distinction between functional and structural obligations is unimportant.

Thus the obligations that a responsibility holder must discharge tell us what the responsibility holder needs to do, and this leads us directly to requirements on the IT system. These fall into two categories. Firstly some of the actual obligations (what the agent *needs to do*) can be transferred to the IT system and realised as functions on the system. These are therefore functional requirements on the IT system. Secondly the IT system may be used to support agents in discharging their obligations. One form of this support is meeting their information requirements, i.e. what the agents *need to know*. Another form of support is keeping a record of what has been done (the *need for audit*). In practice the 'need to do', 'need to know' and 'need for audit' lists are generated for each responsibility and are interpreted as functional and information requirements on the IT system.

2.6. Responsibility Modelling within the ORDIT Modelling Framework

The concepts presented above form part of a modelling framework developed by the ORDIT project. This framework will now be described briefly to show how responsibility modelling fits into the broader field of enterprise modelling.

The Generic ORDIT Model

The core concept in the ORDIT way of looking at organisations is the agent entity. These are the primary manipulators of the state or structure of the system, but essentially they are the people in the socio-technical system, although it is possible for a machine to behave as an agent entity. An agent entity is not just a person but any size of group from an individual to a whole organisation. Other essential elements in modelling an organisation are actions and resources, where an action entity is an operation that changes or maintains the state of the system, and a resource entity is what enables the agent to do the action. An icon showing these entities and the relationships between them is shown in Figure 4.

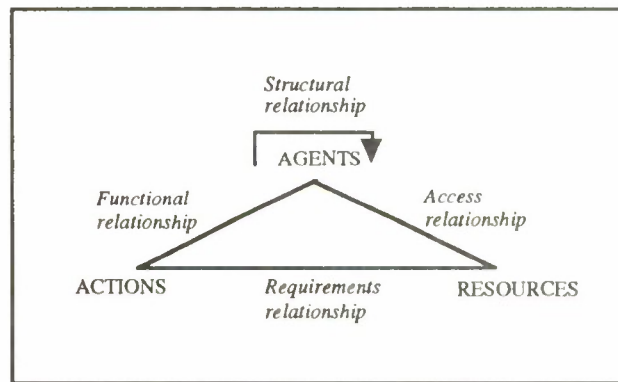


Figure 4. Icon depicting the Generic Concepts in ORDIT Models

We say that agent entities have functional relationships to action entities, since agents do the actions, and that they have access relationships to resource entities, while the action entities have requirements relationships to resource entities; i.e. agents must access resources that are used by actions performed by agents. We are particularly interested in organisational structure so structural relationships between agents are

shown. These are basically responsibility relationships. Relationships between resources (resource schema) and between actions (interactions) are of less interest as these can be represented by data and process models respectively.

The ORDIT Modelling Framework

We have taken this generic model of an enterprise and from it developed three separate but inter-related models of the organisation. These models are of different aspects of the organisation based either on responsibilities, on obligations, or on activities. Each model includes the same three basic types of entity: agents, activities and resources, and also the relationships between them.

Figure 5 shows how the three models are related in that the vertical links join concepts of the same type. This scheme is based on the recognition of obligations, as we define them, as the link between responsibilities and actual activities. The model based on obligations is called an obligation model because the obligations held describe the holder's job or role. Similarly for the resource entity, capability tokens signifying capability to access resources are seen to be the link between rights or authorisations to access and the actual accessing of resources. For example a doctor must first be authorised to access the necessary parts of the IT system by an authorising agent before being able to obtain tokens such as an identifier and password that provide the capability to access. This allows access to the information resource.

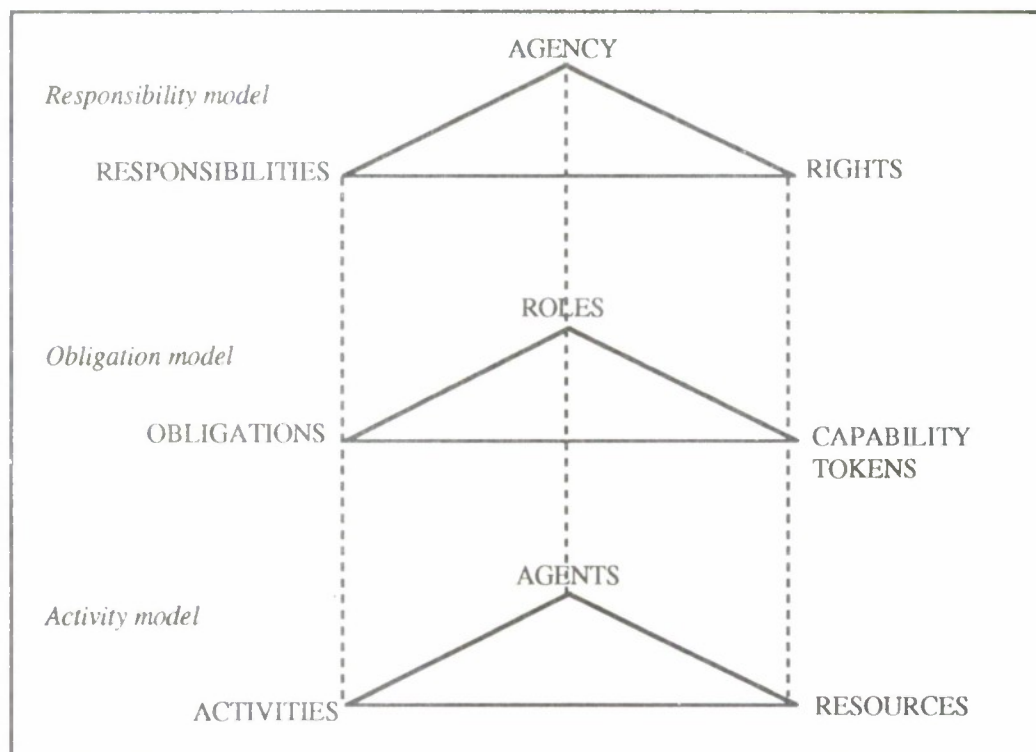


Figure 5. The Three ORDIT Models and how they are related.

3. SECURITY REQUIREMENTS

Figure 5 can be considered as a framework for positioning security requirements. Security is often thought of as a way of binding together a particular set of capability tokens and resources according to the dictates of some security policy. What is often not stated, however, is how the security policy is derived, or any justification that a particular set of bindings compose together to achieve a particular security objective. The richer set of concepts, and the relations between them, exhibited in Figure 5 go some way to providing a language in which these arguments can be made.

We have used the approach outlined in this paper to develop a set of requirements, including security requirements, for an integrated clinical workstation for use in acute hospitals. These are open windows into extensive computer and communication services that provide a broad range of support to clinical staff in meeting their responsibilities at the point of care. The immediate objective of the part of the project cited here has been to capture the nature of the requirements of medical doctors, and ultimately of nursing and other staff who provide direct clinical care. The scope covers the problems of organising the process of medical care, of supporting the medical records, and of implementing computer support through carefully tailored user interfaces. Issues of security of access and confidentiality of information have throughout been of paramount concern.

The methodology used accepts that the fundamental requirement is for users to have a solution to their problems. In other words requirements are the obverse of problems. The process therefore starts by making lists of problems and frustrations with current procedures and records, based on statements from potential medical users. These are couched in the language and concepts of the users.

Structuring the problems and transforming them into user requirements has been done by applying the concept of responsibility and related concepts shown in Figure 5, so that the user requirements can be expressed in terms familiar to the users while at the same time the expression reaches the edge of the kind of language used by systems developers.

The first step was to generate a list of eleven key responsibilities held by medical doctors; a need to know, a need to do and a need for audit list was then generated for each responsibility. Each item in the need to do list was then divided into two components: functions that could be transferred to the system and tasks for the management of the organisation.

Without going into details (which are in any case sensitive), one example of the process should indicate how the concepts were used. Consider the design of the security privileges to be afforded a consultant who has two separate sets of responsibilities: a National Health Service consultant and a private practice consultant. Whereas most of the time these two separate roles do not conflict, it is obviously desirable¹ to enforce separation of duties at the level of the computing system so that the private practice consultant cannot access NHS information, and *vice versa*. The approach we take to defining the boundaries of this separation is first to establish the separate responsibilities and associated rights (need to know, need to do, need for audit) for each

¹ at least in the case of elective treatment; the emergency situation is very different and it might not be at all desirable.

agency. These responsibilities will be different since the two responsibility principals are different (NHS, the patient in person). The obligations associated with each responsibility are then examined and the need for capabilities assessed, bearing in mind that deriving capabilities from rights is part and parcel of, and structurally isomorphic to, the process of deriving obligations from responsibilities. Part of this process involves examining delegation in the way we have suggested (responsibilities cannot be delegated, but derived obligations can) and examining the consequences of the corresponding delegation of capabilities. It turns out, for example, that the delegation rules in NHS and private practice are different.

The mapping from obligations to activities is not always straightforward if the possibility and consequences of failure of activities is to be considered. In terms of the classical approach to fault tolerance, the obligation can be considered as the "acceptance test" and a variety of alternative activities might have to be considered. Each alternate will of course require its own set of resources and access modes and the way in which these derive from generalised capabilities will have to be considered.

4. CONCLUSIONS

A security policy must be capable of showing where security responsibilities lie — for example, who can authorise access to resources, who can validate claims of 'need to know', who can specify and operate prevention mechanisms. The concepts we have designed and shown in Figure 5 are an attempt to provide a modelling language for these kinds of policy concerns so that the issues of drawing security system boundaries can be discussed.

This issue of drawing system boundaries is not trivial.² It might be difficult to prove, but our suspicion is that when a system seriously fails to fulfil its security objectives, the failure can more likely be traced to inappropriate or faulty boundary drawing³ than inappropriate or faulty (use of) security mechanisms. In our experience, the main problem in system boundary drawing is a clear delineation or model of *the space in which the boundaries are to be drawn*. At the level of security mechanisms, this space is one containing activities, resources and agents (our activity model), but there are often real difficulties in relating this space to the space of an organisational security policy if the latter space is not well defined.

The aim of this paper has been to show how responsibility modelling can be used as a means of specifying security policy requirements on an IT system that is meaningful to both users and systems designers and in a way that solves these difficulties. We hope to have shown how obligations define what a responsibility holder must do, and how these can be divided into those that must be done by people and those that are transferable on to the IT system, thus creating functional requirements on that system. By listing what a responsibility holder needs to know and needs to record we can create lists of information requirements.

² In a previous study we conducted for a patient records system for a small ward in a small hospital, one half of the time, and one third of the effort, spent in the requirements phase was simply finding out where the system boundaries lay. These proportions are not untypical.

³ Back door entries for system programmers are a good example of this.

We also hope to have shown how organisational structure may be interpreted in terms of responsibility relationships, and therefore how a model of responsibilities and their associated obligations not only represents function but also the context within the organisational structure in which the responsibility is held. This has direct bearing on whether the responsibility holder holds the necessary authorisation and capability tokens to access the information resources required for performance in a role while respecting the constraints derived from a need-to-know security policy.

ACKNOWLEDGMENTS

We gratefully acknowledge the contributions made by our colleagues in the ORDIT project and also to Stewart Orr, Project Manager of the Integrated Clinical Workstation Project to the ideas presented here. We also gratefully acknowledge the support of the University of Newcastle upon Tyne and the Esprit programme (Project 2301).

TASK-BASED AUTHORIZATION: A PARADIGM FOR FLEXIBLE AND ADAPTABLE ACCESS CONTROL IN DISTRIBUTED APPLICATIONS

(Extended Abstract)

Roshan K. Thomas and Ravi S. Sandhu¹

Center for Secure Information Systems

&

Department of Information and Software Systems Engineering

George Mason University, Fairfax, VA 22030-4444

Keywords: access control, authorization, authorization tasks, separation of duties

1 INTRODUCTION

Historically, the access control problem has been couched within the framework of subjects, objects, and rights (access types). An access control request thus essentially seeks an answer to a question posed typically as: Is subject s allowed access a (or possess the right a) to object o ? A tuple (s, o, a) , which we define as an *authorization*, can be input to a function f , which returns true (or false), to indicate if the subject s has the right a (or not) to object o . We can visualize the implementation of such a function with an access control matrix. This subject-object view can be traced to the subject-object paradigm of access control that was formulated in the early era of the development of general multi-user computers and operating systems [7, 5].

Over the last two decades we have seen considerable advancements in the discipline of computer security. In particular, we have seen the evolution and development of many access control models. The initial proposals of Lampson [7] and Graham and Denning [5] led to formulation of the *HRU model* by Harrison, Ruzzo, and Ullman [6]. This was followed by the development of the *Take-Grant Model*. A good summary of these early efforts (in the first decade) can be found in [13]. More recent efforts have resulted in the *Schematic Protection Model* (SPM) by Sandhu [8], the *Extended Schematic Protection Model* (ESPM) by Amman and Sandhu [1], and the *Typed Access Matrix Model* (TAM) also by Sandhu [12].

In reviewing the above development in access control models, we note that the overriding concern was the fine-grained protection of individual objects and subjects in the system. However, with the advent of databases, networking, and distributed computing, we have witnessed a phenomenal increase in the automation of organizational tasks,

¹The work of both authors is partially supported by a grant from the National Security Agency, contract No: MDA904-92-C-5140. We are grateful to Pete Sell, Howard Stainer, and Mike Ware for their support and encouragement.

as well as the computerization of information related services. Is it not fitting that we shift our focus on security issues from the protection of individual objects and subjects in isolated computer systems, to the automation and provision of distributed tasks and services? Such tasks may involve groups of related activities that span multiple networks and databases. Authorization (access control) may be required for groups of related activities at several departments. Thus, we believe it is timely and necessary to transcend the above classical subject-object view of access control, and work towards newer paradigms.

2 TASK-BASED AUTHORIZATION

In this section we elaborate on the central point of this paper. In a nutshell, authorizations in distributed applications should be seen in terms of tasks/activities rather than individual subjects and objects. We argue this, based on two emerging trends:

1. The integration of computing within organizations, and the subsequent increase in the automation of organizational functions and work-flows.
2. The shift from main-frame computer systems to workstations and client-server technologies.

With the first trend, we are witnessing an increased demand for the support of multi-system applications. Such applications may even cross departmental and organizational boundaries. A very good example of this in the telecommunications industry is that of service order provisioning [2]. This is the automated process of providing telephone services to customers. Upon receiving a service request, a service order is generated. The processing of the service order demands coordination and data exchange between several business units in the company, and eventually leads to the assignment of lines and equipments, as well as the update of billing information (among others). As another illustration, consider the automation of a paper-based sales order processing application (system). Sales order processing begins with the receipt of a customer purchase order. The subsequent processing steps may involve several documents such as sales orders, invoices, customer statements, and journal vouchers. These documents may propagate through several departments in the organization(s) such as SALES-ORDER, CREDIT, FINISHED-GOODS, SHIPPING, BILLING, and ACCOUNTS-RECEIVABLE, completing the many subtasks involved in processing the sales order request.

The above documents would have to undergo a sequence of authorization/approval phases. For instance, a sales order may be routed through the CREDIT department, and shipment authorized only after a credit check on the customer succeeds. An organization may also incorporate various controls and checks to minimize risks due to fraud. One way to achieve this is through separation of duties. Custodial functions performed by FINISHED-GOODS and SHIPPING departments are separated from the recording functions of BILLING and ACCOUNT-RECEIVABLE, and the authorization functions performed by the SALES-ORDER and CREDIT departments. Separation of duties among individuals can ensure that only goods intended for shipment to customers

are removed from the FINISHED-GOODS storeroom, and that all such goods are shipped only to authorized customers and are billed correctly.

The need for task-based authorizations arises even within the world of a single user in an office, accomplishing a simple and routine task such as printing from a workstation. Resources such as printers, files, and applications, may be shared over a local area network. The printing of a multimedia document, for example, may require authorization and access to multiple servers, and data stored at several objects.

A task, as identified in the scenarios above, may be characterized as one that:

- is long-lived;
- may involve multiple subtasks, where a subtask may need to be individually authorized;
- may involve multiple principals to carry out individual subtasks;
- is distributed in space and time.

We believe, that the authorization of tasks that span multiple systems over departmental and organizational boundaries, as well as those that involve individual workstations and servers, are conceptually similar and can thus be addressed in a unified manner.

As an initial attempt, we introduce the abstraction of an *authorization-task* as a unit for managing the authorizations in distributed applications. A *task* is a logical unit of work in such applications and may consist of several individual subtasks. In the earlier mentioned sales order processing system, when an order is taken, a corresponding authorization-task is begun. Individual authorization actions, such as the credit approval for a customer, can be done by finer units of authorization-tasks called *authorization-subtasks*.

3 FLEXIBLE AND ADAPTABLE ACCESS CONTROL

In the subject-object view of access control, every authorization tuple represents a primitive unit of access control information. Collectively, these tuples are unrelated to each other. Contrast this with the requirements of our application above, where the sales order processing task involves several related individual subtasks that need approvals (authorizations). Such a requirement calls for a higher level control structuring facility. An analogy to the above predicament can be seen in the realm of transactions and databases. Classical transactions with the ACID (Atomicity, Consistency, Isolation, and Durability) properties represent concurrent but unrelated units of work. Consider a requirement (re-stated from an example in [14]) for the sequencing of three transactions such as:

Execute T_1 , followed by T_2 and T_3 in parallel; If T_2 fails, then abort T_3 as well.

With the transaction as the main control abstraction, it is impossible to implement the above without ad-hoc application programming. This has led researchers to propose other abstractions, such as the so-called ConTracts [14]. Authorizations in distributed applications similarly call for abstractions beyond individual subject-object authorizations.

We list some of the obvious questions that need to be answered on the road that could lead to a task-based authorization approach.

1. Abstraction and Modeling:

What are the proper abstractions to express and manage the required authorizations for tasks?

2. Grouping Authorizations:

How can groups of related activities be collectively authorized?

3. Flow Control and Dependencies:

How can we describe and manage the control flow and dependencies between the authorizations of the various steps in a task?

4. Incorporation of Integrity Mechanisms:

How can we incorporate controls such as those based on separation of duties and multiple approvals?

5. Failures, Exceptions, and Recovery:

How can we handle failures in the authorization of individual steps of tasks? If a certain authorization/approval for a certain step in a task is not forthcoming, we may wish to specify alternate paths to be taken. For example, if the credit worthiness of a customer cannot be immediately established, the organization may have a policy that allows the sales order to go through, so long as the value of the items ordered does not exceed a certain amount. Or perhaps, in another scenario, we may wish to express that a certain authorization/approval step may be selectively ignored, under some conditions.

4 AN EXAMPLE

To illustrate the intuition, flexibility, as well as generality, of task-based authorizations, let us take a concrete example that requires separation of duties. Suppose there already exists some predefined mechanisms and formalisms for expressing separation of duties. For example, Sandhu in [10, 11] has proposed *transaction control expressions* as an approach to implement separation of duties in computerized systems. It is based on a database activity model that utilizes the notions of *transient* and *persistent* objects. Transient objects include documents such as vouchers, purchase orders, sales slips, to name a few. These objects are transient in nature in the sense that they issue a finite set of operations and then leave the system (in a paper world this happens when a form is archived). These operations eventually affect persistent objects such as inventory databases, and bank accounts. The fundamental idea is to enforce controls primarily on the transient

objects, and for transactions to be executed on persistent objects only as a side effect of executing transactions on transient objects.

Consider a check processing application where a clerk has to prepare a check and assign an account, followed by three (separate) supervisors who have to approve the check and account, and finally the check to be issued by a different clerk (in the paper world, this would be accomplished through a voucher). This can be represented by the following transaction control expressions:

```
prepare • clerk;  
3: approve • supervisor;  
issue • clerk;
```

The colon is a voting constraint specifying 3 votes from 3 different supervisors. Each expression consists of a *transaction* and a *role*. Separation of duties is achieved by requiring the users who execute different transactions in the transaction control expression be all distinct.

Now consider a certain application that requires the use of two vouchers (transient objects). Now suppose the first voucher needs to be completely processed before the second one can be started. Further, we require separation of duties across these vouchers. We would proceed by defining an authorization-task that consists of two authorization-subtasks. Each subtask is assigned to a single transient object (a voucher in this example) and executes the transaction control expressions of the transient object. A subtask may specify the failure semantics within a transient object. If for example, the same clerk attempts to issue the check after preparing the voucher, the separation of duties requirement is violated. The authorization-subtask may then pursue some alternate action. A violation in the separation of duties across the two vouchers will be detected by the parent authorization-task. The authorization task may have to maintain global history and context information for this purpose.

If a transient object generates other transient objects, complex (and often nested) structures of authorization-tasks and subtasks could result. In the paper world, this would happen if approval on a form generates other forms. The key here is to express the authorization-task in some convenient formalism/language that has the flexibility to capture the failure semantics and dependencies between subtasks. We leave such specification as well as enforcement issues open for investigation.

5 CONCLUSIONS

In this paper, we have argued for a new paradigm for flexible and adaptable access control in distributed applications. We have motivated a task-based approach that represents a departure (and a paradigm shift) from the subject-object view of access control. In motivating the need for this paradigm shift, we have drawn an analogy from the experience of the database research community. Database researchers realized the limitations of the classical transaction model as a unit to support long-lived and distributed activities,

and consequently had to reexamine the transaction concept itself. This has led to the formulation of many advanced transaction models.

We believe that many of the ideas in the recent advances of transaction models are useful [4]. For example, one may specify dependencies and failure semantics between transactions, in a flexible way and often user-defined fashion. The insights here could be used to give more internal structure and semantics to authorization tasks. A long-term vision should be the development of a comprehensive framework (such as ACTA, for database transactions [3]) for specifying and reasoning about authorizations in distributed applications. Also, we need to investigate how inter-task dependencies can be specified and enforced.

References

- [1] Ammann, P.E. and Sandhu, R.S. "The Extended Schematic Protection Model." *Journal of Computer Security*, Volume 1, Numbers 3 and 4, 1992, pages 335-383.
- [2] M. Ansari, L. Ness, M. Rusinkiewicz, and A. Sheth. Using flexible transactions to support Multi-system telecommunications applications. *Proc. of the 18th VLDB Conference*, British Columbia, Canada, 1992.
- [3] P. K. Chrysanthis and K. Ramamritham. ACTA: A Framework for specifying and reasoning about transaction structure and behavior, *Proc. of the ACM SIGMOD conference*, pages 194-203, 1990.
- [4] *Database Transaction Models for Advanced Applications*, A.K. Elmagarmid (Editor), Morgan Kaufmann Publishers, San Mateo, California, 1992.
- [5] Graham, G.S. and Denning, P.J. "Protection - Principles and Practice." *AFIPS Spring Joint Computer Conference* 40:417-429 (1972).
- [6] M.H. Harrison, W.L. Ruzzo, and J.D. Ullman. Protection in operating systems. *Communications of the ACM*, 19(8), pages 461-471, 1976.
- [7] Lampson, B.W. "Protection." *5th Princeton Symposium on Information Science and Systems*, 437-443 (1971). Reprinted in *ACM Operating Systems Review* 8(1):18-24 (1974).
- [8] Sandhu, R.S. "The Schematic Protection Model: Its Definition and Analysis for Acyclic Attenuating Schemes." *Journal of ACM* 35(2):404-432 (1988).
- [9] Sandhu, R.S. "Separation of Duties in Computerized Information Systems." In *Database Security IV: Status and Prospects*, (Jajodia, S. and Landwehr, C.E., editors), North-Holland 1991, 179-189.
- [10] R.S. Sandhu. Transaction control expressions for separation of duties. *Proc. of the Fourth Computer Security Applications Conference*, pp. 282-286, 1988.

- [11] R.S. Sandhu. Separation of duties in computerized information systems. *Database Security IV, Status and Prospects*, S. Jajodia and C.E Landwehr (Editors), Elsevier Science Publishers B.V. (North-Holland)
- [12] Sandhu, R.S. "The Typed Access Matrix Model." *Proc. IEEE Symposium on Research in Security and Privacy*, Oakland, California, May 1992, pages 122-136.
- [13] Snyder, L. "Formal Models of Capability-Based Protection Systems." *IEEE Transactions on Computers* C-30(3):172-181 (1981).
- [14] H. Wachter and A. Reuter. The ConTract Model. In *Database Transaction Models for Advanced Applications*, A.K. Elmagarmid (Editor), Morgan Kaufman Publishers, San Mateo, California, 1992.

Identification and Authentication when Users have Multiple Accounts

W.R.Shockley
Cyberscape Computer Services
8051 Vierra Meadows Place
Salinas, CA 93907

Abstract

Most security models assume that each user has only a single account. This simplifies the enforcement of a security policy by allowing rules about *individuals* to be replaced by rules about *accounts*. However, the assumption fails badly for large-scale networks, because no realistic approach exists for ensuring that it is true. It is therefore preferable to acknowledge that users may have multiple accounts and make adjustments to the identification and authentication mechanisms. Examples of security policies where the potential for multiple accounts makes a difference are given. A simple mechanism for *account alias detection* is described that supports the correct enforcement of these policies even when account aliases may exist. The key idea is to separate the *authentication* function -- determining that the owner of the account is present -- from the *identification* function -- determining whether the owners of two different accounts are the same individual. Identification is a natural application for biometric technology. The use of biometrics for identification alone has significant operational and cost benefits over its use for authentication. A system that used conventional authentication techniques coupled with biometric identification would seem to be optimal.

1. Terminology

Before an **individual** can successfully use a shared computer system to perform user-level work, typically an **account** must be established for that individual. This operation, which will be called **user registration**, is a sequence of steps performed by an administrator who will be called a **registrar**. User registration is a security-critical operation, and the registrar is trusted to conform to procedural and administrative controls ensuring that individuals not authorized to own accounts are not registered, and that any account information pertaining to a newly-registered individual is properly verified and entered into the account database.

The account itself is represented by a protected, security-critical data record of some sort that contains these data. The set of account records is keyed by some kind of **account identifier** that uniquely identifies individual account records. Often "user friendly" substitute keys, such as a **user name**, are supported that can also be used to identify the account record.

Among the data generated during the registration operation are data that can be used later to **authenticate** the new user when that user establishes a session on the system. In general, we can think of the authentication data as consisting of two parts: an **authenticator** which is held by the actual user, and an **authenticand** which is stored as part of the account data record. An **authentication algorithm** is built into the trusted computing base that, given an *<authenticator, authenticand>* pair determines, with an appropriately high probability of success, whether or not the given authenticator and authenticand match. We think of the registrar as generating (or causing to be generated) an initial *<authenticator, authenticand>* pair for each new account, with the authenticator being provided to the owner of the new account and the authenticand being stored in

the account record for later use. It is often the case (e.g., when the authenticator is a password) that the procedure for generating and distributing a new *<authenticator, authenticand>* pair can be automated so that already registered users may be permitted to replace their own authentication data without further intervention by a registrar.

This simple model of an **authentication technology**, consisting of

- * a method for generating a non-repeating sequence of *<authenticator, authenticand>* pairs
- * a method for determining whether a given authenticator was generated in association with a given authenticand
- * a procedure to be followed by a registrar when assigning new accounts to users, and
- * an optional method for replacing a registered user's old *<authenticator, authenticand>* with a new one

is sufficiently abstract to cover a wide range of existing authentication approaches, such as password or passphrase-based authentication, the use of smart cards or other physical tokens for authentication, the use of biometric information, techniques based on symmetric key encryption as found in Kerberos [MIL87], or on asymmetric key encryption as proposed for DSSA [GAS89,LIN90].

A typical scenario for *using* an authentication technology to limit the access to a computer system to authorized users is the following:

1. The prospective user first claims to own a particular account by supplying its account identifier or its surrogate, a user name.
2. A trusted component of the system fetches the nominated account record and prompts the user for an authenticator.
3. The user supplies an authenticator.
4. The trusted log-on procedure uses the comparison method to determine whether this authenticator together with the authenticand found in the nominated account record are valid *<authenticator, authenticand>* pair, granting or denying access to the computer system depending on the result.

Of course this procedure is vulnerable to situations where the authenticator has been compromised, either deliberately by the owner of the account or registrar, or from some other cause. Some procedural means of recovering from an authenticator compromise, e.g., generation of a new *<authenticator, authenticand>* pair, is therefore often considered a requirement.

We will define the **identification function** as the determination, for two given accounts, whether or not these accounts are owned by the same individual. A third result -- "no determination" -- is also admitted. While this may not be an immediately intuitive definition of what the word "identification" means, it turns out to be the most useful one for dealing with the problems described later in this paper.

In order to support a distinct identification function, we assume that an account record may optionally include **identification data** collected by a registrar from the owner of the account when the account was created. In principle, the identification data should have the following properties:

- * It should uniquely identify the individual. I.e., no two individuals should have the same identification data.
- * It should be difficult for a given individual to forge or produce false identification data whether at different times or in different places.

While weaker kinds of identification data may be imagined, the use of a **biometric identification technology** comes immediately to mind. These technologies generally have the following functional components:

- * a **biometric reader** can be used to capture physical data from an individual -- e.g. fingerprint, retina scan, voiceprint.
- * a **reduction algorithm** is available for reducing the raw biometric data to a canonical representation we will call a **biometric profile**.
- * a **profile comparison algorithm** that, given a pair of biometric profiles tells, with some appropriately small probability of error, whether or not the profiles represent measurements taken from the same individual.

2. Authentication and Identification Technologies Contrasted

The thesis of this paper is that while the requirements driving the choice of an authentication technology and those driving the choice of an identification technology are *similar*, they are also in certain environments (e.g., the Internet) and for certain policies *incompatible* -- which motivates us to carefully separate the two functions so that we can use the best technologies for each. In particular, the requirement on an authentication technology to be able to recover expeditiously from an authenticator compromise clashes with the requirement on an identification technology that it must be difficult to change an individual's identification data.

As a concrete example, suppose that a biometric technology is chosen for the authentication function (the usual security application heretofore suggested for these technologies). The *<authenticator, authenticand>* pair is taken as:

- * for the authenticand, a biometric profile captured from the user during the registration operation
- * for the authenticator, a biometric profile captured from the user during log-on

The biometric comparison algorithm is used as the authentication matching algorithm. This approach has a fatal flaw if recovery from authenticator compromise is an issue (and it usually is). By definition, it is difficult to change a user's biometric profile (otherwise, the biometric technology simply wouldn't work as advertised). Therefore, if a user's biometric profile is stolen (i.e., its bits are known to a penetrator) one has the worry forever after that the penetrator can successfully bypass a log-on biometric reader somewhere and use those bits to spoof the log-on software.

An example of the converse class of problems might involve the use of public-key authentication technology in environments where identification (i.e., detection of alias accounts) is an issue. Suppose, for example, that (as is asserted by the DSSA designers) a user's public key is taken as the user's "real" identity. In environments where it is impossible to prevent a given user from acquiring multiple registrations (e.g., Internet) it is quite possible for a user to accumulate several distinct public key "identities" -- i.e., by being registered by different registrars. As we will see, this becomes a problem for certain security policies.

3. The Single Account Assumption

In my experience, most security modeling efforts have assumed, either tacitly or explicitly, that *no user has more than a single account*.. This is an attractive simplifying assumption because it allows rules about individuals (e.g., security policies) to be immediately re-expressed as a set of homologous rules about accounts that can then be enforced by a properly designed reference monitor.

For example, if we examine a conventional access control policy expressed in terms of **access control lists (ACLs)**, what we will find stored in the ACLs are typically account identifiers or surrogates for account identifiers (i.e., user names). The access of *individuals* to a protected object is really controlled indirectly: the ability of a user to obtain a terminal process is controlled by authentication, making sure that the individual owns the account used to label the process. When the process makes a request to access a protected object, typically it is the user name associated with the process that is matched against the user name found in the ACL.

More recent systems addressing the problems of authentication and account maintenance systems in networks, e.g., DSSA, use references to a different substitute key -- viz. the account's current authenticand (public key). This allows the account record to be renamed (i.e., moved around as part of the name service object hierarchy). However, since authenticands are generated on an account-by-account basis there is still no guarantee that multiple accounts for the same user (which may have been installed by completely different registrars) are associated with the same authenticand.

Neither style of system precludes situations where an individual owns multiple accounts -- nor does enough information exit to definitively prevent or detect such a condition. A few security policies where this makes a difference will be described later. It has been all-to-common practice to simply *assert* as an implied or explicit axiom for a security policy model that a given individual possesses at most a single account.

What is wrong with this assumption?

- * In practice, even for small shared systems or networks, the rule is often honored more in the breach than the observance. One commonly finds that, for very practical reasons, operators may have both "system" and "user-level" accounts; "group accounts" are common, and if different machines are networked, various users may have distinct accounts on various machines.

- * When it is left up to a system operator to enforce the "one user, one account" rule administratively, the enforcement becomes administratively difficult as the system grows to incorporate several operators and more users than re personally known to all of them -- yet no help is provided in determining when multiple accounts exist.

- * For still larger networks, such as Internet, it becomes unrealistic to even *suppose* that such a rule could be enforced as users are registered. Either the owner of a new account would have to be trusted to provide a list of all other accounts owned by that individual, or some service

would have to be provided that *searched* the network for all accounts owned by the user being registered. The first is manifestly insecure, and the second highly impractical.

* Finally, for some administrative domains a policy of “one user, one account” may be considered inconsistent with privacy regulations. It can be argued that the notion of “privacy” includes the right to perform at least some activities in such a way that they cannot be correlated with other activities performed by the same individual -- e.g., that it must be possible to perform some functions *anonymously*.

4. Multiple Account Model and Mechanism

It appears necessary, then, to embrace the requirement to permit users to own multiple accounts. A basic model incorporating such a requirement is neither complex nor surprising in content. We permit users to own multiple accounts. Two accounts owned by the same user are called **alias accounts**. It is not generally the case that alias accounts were created by the same registrar, nor do we require a registrar to determine, when a user opens a new account, whether an alias exists. As for the conventional model, part of the registration procedure includes the generation of a valid *<authenticator, authenticand>* pair with the authenticator being given to the account owner, and the authenticand stored within the account record.

Some accounts, when they are opened, will include within their account record additional identification data, collected from the registrar from the user at the time the account is opened. If a biometric technology is used to support identification, this means the user must be present at the time the account is opened so that a biometric profile can be computed. Weaker forms of identification technology (e.g., Social Security number, mother’s maiden name. etc.) might not require the physical presence of the user but would result in a correspondingly weaker identification system. Note that identification data is *not* intended to be used during user authentication but for identification only in support of specific policies. We will see later why it is desirable to make this exclusion.

Accounts that contain the optional identification data are called **identified accounts**. Accounts without identification data are called **anonymous accounts**. It is assumed that the record structure for account records allow identified accounts to be distinguished from anonymous accounts. In practice, type information identifying the particular authentication and identification technologies employed would be included as well.

When a user logs in to initiate a session, the trusted computing base authenticates the user just as for the Single Account Model. The identification data is not used by the authentication system in any way. As usual, the result of a successful authentication is that a terminal process is created for the user. We have thus established after a successful authentication that the user, on whose behalf the process is executing, is the owner of the account associated with the process.

The identification data, copied from the account record, is also associated with the process as part of the process security data. Since we know after authentication that the current user owns the account, we know that the identification data is also the data for the current user even though it was *not* collected at the time of log-on but at the time of registration. Of course, we assume throughout that the account record has been properly protected by the TCB.

Now, whenever the process requests access to an object, the identification data associated with the process is available for potential use by the reference validation mechanism. In a distributed environment, it would be included and protected with other security context data as part of the request context for remote accesses.

It remains to describe how the identification data might be *used* by the reference validation mechanism to enforce specific security policies: the discussion above is focused on how the proper identification data is *supplied* to the RVM.

5. Selected Security Policies that Need Identification Data

The existence of alias accounts is of little consequence to the enforcement of many traditional security policies. For example, policies such as the DoD mandatory policy for the classification of label do not depend on the identity of individuals, but only their clearances. It makes no difference from which of many properly registered accounts an individual might choose to access information with respect to this policy. Similarly, a policy that *grants* access to an object based on account identifier or user name is not compromised should the user happen to own a different account: the user would be able to access the object in question from one account, but (inconveniently but securely) not from the other. If the user should try to access the object from the wrong account, the access would simply fail.

However, there are fairly interesting policies whose enforcement *fails* in the presence of alias accounts unless identification data supporting detection of alias accounts is provided as part of the request context.

5.1 Explicit Denial

One such policy is part of the DoD Criteria for Trusted Systems [DOD85] for the higher ratings -- viz., the requirement to support the **explicit denial of access**. The intent of this requirement is that it should be possible to positively deny access to a given protected object by a specific *individual*.

If a conventional implementation is used (i.e., using ACLs containing account names or equivalents) and alias accounts exist, the policy cannot be accurately enforced. Denying access to a particular *account* does not ensure that the accounts' owner cannot gain access using an alias account. As has been noted earlier, a *search* for all possible alias accounts at the time the explicit denial is entered or encountered is unfeasible in large networks.

The mechanism described earlier solves the problem easily. An explicit denial as recorded in, e.g., an ACL, is constrained to contain a reference to an identified account and is treated as a reference to the identification data contained in the named account record. When a request to access the protected object is mediated, the RVM follows this reference to obtain the identification data (i.e., a biometric profile) from the referenced account, and uses the biometric comparison algorithm to match it against the profile contained as identification data in the request context. The first data refers to the individual intended by whoever set up the ACL: the second to that individual on whose behalf the request is being made. Of course, all requests made from an anonymous account must be denied.

An unfeasible search is avoided, because all of the information needed to make the decision can be directly located. Note that it is quite possible that the profile located via the ACL, and the profile contained in the request context may well come from *different* identified accounts. The policy is enforced because if a denied individual makes a request from *any* identified account, the profile will match and the request therefore rejected, while if from an anonymous account (or one using a different identification technology) the request will be denied because no profile of the right type is provided with the request.

5.2 Separation of Duties

Another class of policies that are very important for many production-level applications are policies related to the **separation of duties** [CLA87]. Generally, these policies require that specific steps of given business process must be performed by *distinct individuals*. The intent of such policies is generally combined one of reducing error rates and inhibiting fraud by requiring collusion among would-be perpetrators. Approaches for enforcing a separation of duties policy involving the use of special-purpose accounts, groups, or role-based mechanisms fail badly in the presence of alias anonymous accounts because the possibility exists that a lone malicious individual could perform the critical steps from different accounts (e.g., on perhaps the day he or she changed jobs).

Such policies are very naturally specified and enforced where identification data, over and above authentication data, is available. When the first step is performed, the security context (including identification data) is captured and stored in a protected location (i.e., within the security perimeter). When a request for initiation of the second step is received, in addition to whatever other security checks may be indicated, the identification data from the second request is compared to that stored for the first. If there is a match, the requests presumably come from the same individual and initiation of the second step is blocked.

Again, since identification data must be used, requests to perform either step from anonymous accounts must be rejected.

5.3 Support for Privacy

In an earlier section, reference was made to policies for the privacy of information and activities. Such policies are considered of more or less importance than security policies depending upon the administration involved.

The author considers one aspect of "privacy" (as something distinct from "security") as the ability of an individual to perform some activities (e.g., private activities such as managing one's checkbook) that cannot be definitively correlated with other activities performed by the same individual (e.g., public or official activities for which an individual is held accountable). Put another way, individual accountability for public or official actions must be possible, but without implying disclosure of *all* of an individual's activities, however personal.

The provision in the model for anonymous accounts is intended to capture this possibility. Presumably, objects for which public accountability is required would be protected by ACLs prohibiting access from *any* anonymous account, while individuals could freely enjoy the use of the system via anonymous accounts for unofficial or personal business.

This approach would also carry over into the audit subsystem. Identification data, when part of the request context, would be captured as part of the audit trail allowing the actions of a user from different identified accounts to be correlated.

Some have questioned the desirability of anonymous accounts should technology supporting identified accounts become widely available. This question is part of the ongoing discussion regarding the trade-off between hampering the investigative capabilities of law enforcement agencies and providing support for individual privacy. This is clearly a social issue: some administrative domains might choose to support a policy of "no anonymous accounts" so that all of an individual's activities within the domain might be, in principle, subject to correlation, while others might choose to permit registration of anonymous accounts by some or all of the user community. However, this is a social, not a technical issue. I have included the notion of an anonymous account so that the model could be applied to a wide range of social policies.

It is worth noting that under the definitions given, most user accounts today providing access to on-line services are anonymous in the technical sense. (Information such as "user name" is provided by the user, and not verified by the registrar.) Nothing at all prevents a given user from purchasing as many accounts as desired, under whatever pseudonyms or address desired -- as long as the monthly bill is paid! A social decision to prohibit such accounts in the future thus represents a distinct *erosion* of privacy.

5.4 As a Primary Access Control Mechanism

Finally it can be noted that the use of identification data in place of account identifier as a *primary* tool for granting individuals access to data merits consideration. One could envision an "access control rule language" that extended current account names with a modifier indicating that it is the identification data that is to be compared, not the account identifier, in order to permit access. For example, a rule such as

grant (read, write, execute) to FOO

would be interpreted to mean "permit access only if from account 'FOO'" while

grant (read, write, execute) to !FOO (where FOO' is an identified account)

would be interpreted to mean "permit access to the individual owning account FOO from *any* identified account owned by that individual". The benefit intended by this suggestion is strictly operational -- as anyone who has gotten involved with trying to access "private" objects from multiple accounts will know. It is painful to try to set up ACLs in such a way that such objects can routinely be accessed from *all* of your accounts!

6. Implications

In the environment where support for alias accounts is important -- viz., large-scale networks, the issue arises as to how the identification data is to be protected. However, it is always piggy-backed on an already protected entity -- viz., security-critical account records from the account registry, security-critical process contexts, or security-critical request contexts. The identification data is protected in exactly the same way as the authenticand, using whatever mechanism is used to protect the authenticand -- for large networks, typically end-to-end cryptographic sealing to make the record involved tamper-detectable.

Secondly, it should be noted that the authentication subsystem proper makes *no* use of the identification data. This means that inclusion of identification data in the security-critical records that carry it *cannot* disturb the correctness of the authentication system.

In fact, one can go further. Since authentication does not depend in any way on the secrecy of the identification data, there is *no compromise* if identification data is disclosed. This result may seem at first startling. What must be understood is that it is the *association* between identification data and the account record that must be protected within the security perimeter. A malicious user or user process has no way of forging such an association, providing the underlying protection mechanism is sound, even if it knows what the identification data *is*. Authenticators must be protected from disclosure -- identification data does not. What we are trusting, in the final analysis, is that the account registrar properly collected the real identification data from the owner of new account for inclusion in the account record.

Supposing that the security perimeter is breached and the account record compromised, recovery after repair of the perimeter is straightforward. A new authenticator and authenticand must be

generated for the compromised accounts. However, if a cryptographically sealed record containing the biometric profile can be located and validated *anywhere in the system* the biometric profile therein may be safely re-used -- only if no such record can be located must the users biometrics be re-read.

The proposed use of biometric technology is roughly an order of magnitude cheaper than the usually suggested use as an authentication mechanism. If it is used for authentication, then a relatively expensive biometric reader must be located at every log-in point. If the use of the biometric profile is restricted to identification alone, readers are needed only at designated registration workstations as no profile needs to be measured at the time of log-on.

It is difficult, however, to see how requiring the physical presence of the user at the time of an initial registration can be avoided.

7. Acknowledgements

The basic mechanism described in this paper, restricted to the use of biometric technology and cryptographically sealed account records, was invented jointly by the author and George Gajnak while employees of Digital Equipment Corporation. Mr. Gajnak deserves equal credit for the Multiple Account Model, while any mistakes in this paper are my own. We have jointly filed for patent protection internationally for the use of the mechanism to enforce all policies described in this paper. All rights to the invention have been assigned to Digital Equipment Corporation. The contents of this paper and disclosure of this invention should not be construed as a product commitment by Digital Equipment Corporation.

Acknowledgement is also due to the anonymous referees of this paper for their comments and clarifications.

8. Bibliography and References

Surprising little seems to appear in the literature concerning identification and authentication *policies* although specific authentication technologies have received widespread attention.

[CAR88] S. Carlton, J. Taylor, and J. Wyzynski, "Alternative Authentication Techniques", *Proceedings, 11th National Computer Security Conference*, Baltimore, MD, Oct. 1988 pp.333-338. Discusses general characteristics of biometric technologies for use for authentication but does not consider the impact of compromise. No distinction is made between the authentication and identification functions.

[CLA 87] D. Clark and D. Wilson, "A Comparison of Commercial and Military Security Policies", *Proceedings of the 1987 Symposium on Security and Privacy*, Oakland, CA, May 1987, pp.233-248. Contains much basic material on separation of duties policies.

[DOD85] Department of Defense, *DoD Trusted Computer System Evaluation Criteria*, DoD 5200.28-STD, Washington, D.C. 1985 [U.S. Government Printing Office 008-000-00461-7]. As well as defining requirements for explicit denial, the sections on control objectives and background rationale contains basic information about authentication, identification, and audit requirements for the Department of Defense.

[GAS88] M. Gasser, *Building a Secure Computer System*, Van Nostrand Reinhold Company, New York NY, 1988. Everything that is said about the distinction between identification and authentication on pp.20-22 is correct (if non-specific) but no practical application of the distinction is elaborated.

[GAS89] M.Gasser, A.Goldstein, C.Kaufmann, B.Lampson, "The Digital Distributed System Security Architecture", *Proceedings, 12th National Computer Security Conference*, Baltimore, MD, Oct.1989, pp305-319. Description of the DSSA public-key based security and authentication framework for networked systems. No specific support for the identification function is described.

[LIN90] John Linn, "Practical Authentication for Distributed Computing", *Proceedings of the 1990 IEEE Symposium on Security and Privacy*, IEEE Computer Society, 1988, pp31-39 presents a practical and scalable public-key authentication design (an instance of DSSA described in the previous paper). Such a design could be easily enhanced to support the additional identification function.

[MIL87] S.P.Miller, R.C.Neumann, J.I.Schiller, and J.H.Saltzer (Massachusetts Institute of Technology), *Project Athena Technical Plan Section E.2.1*, "Kerberos Authentication and Authorization System" describes a symmetric-key based authentication system for networks that threatens to become an *ad hoc* standard.

ENTERPRISE SECURITY SOLUTIONS PANEL

Overview

Large corporations are rapidly embracing the use of "enterprise-wide" networks to improve their competitiveness. Security concerns have hindered the deployment of these networks, particularly when connecting between companies and across public networks.

The "Enterprise Security Solutions" panel brings together the unique perspectives of leading industry experts. The panelists will describe innovative security solutions for large corporate networks and opinions on future industry directions in protecting information.

Each panel member will present a "theme" topic for which they have specific expertise with the goal of providing a balanced session. Topics to be discussed include: the strategic advantages of Information Security; building firewalls between trading partners, customers, and untrusted third party networks; secure access to enterprise networks from public networks; security authentication servers and security management; and tokens for authentication in enterprise networks. The composition of the panel and each person's theme is:

Paul Lambert Motorola - Chair
Enterprise Security Solutions

Don Sortor J. P. Morgan
Business Issues of Information Security

Peter Browne Motorola
Securing the World's Largest Private Internet

Bill Bosen Enigma Logic Inc.
Enterprise-Wide Security with Token-Based Access Control

Dave Bauer Bellcore
Secure Distributed Computing for Heterogeneous Operating
System Environments

The following are brief position statements from each of the panel members.

Paul Lambert, (Motorola - Chair) - Enterprise Security Solutions

There is no single "magic bullet" to solve the problems of computer security. This is particularly apparent when attempting to look at the broad scope of security issues for corporate enterprise-wide computer networks. Solutions for this environment must confront problems that include workstation diversity, protocol interoperability, personnel policies, export restrictions, and budget limitations. The solutions pursued by corporations to meet these constraints often have little or no relationship to computer security research.

Don Sortor CFE (J.P. Morgan) - Business Issues of Information Security

Implementing enterprise security in an ever-changing distributed environment presents a unique challenge to today's security manager. Where once there was a single machine, with specific physical and logical controls for that environment, we are now in a world where "anything goes".

There are numerous technical "solutions" to secure enterprises, ranging from architectures (OSF/DCE, SESAME, KERBEROS, etc.), to vendor solutions promising single point of entry to an organization's resources, to "real-time" monitors which constantly watch a heterogeneous environment for anomalies which may lead to breaches in security. However, without the proper organizational and procedural controls in place PRIOR to implementing any technical solutions, those solutions are bound for rough waters.

Peter Browne (Motorola) - Securing the World's Largest Private Internet

Motorola has implemented one of the world's largest peer-to-peer private Internets, with over 60,000 people directly connected to each other. The geographic dispersion is worldwide, with over 40 countries represented. The issues of security must be dealt with in a very proactive manner in order to ensure business viability.

A number of initiatives relating to enterprise-level security have had high visibility and success, among them an effort to develop a corporate secure dial-in capability, the building of "firewalls", the development of enterprise security "standards" and educational events focusing on network administrators and end users.

Bill Bosen (Enigma Logic) - Enterprise-Wide Security with Token-Based Access Control

Enigma Logic is a leader in the field of "Enterprise-Wide" computer information security. Since its founding in 1982, Enigma has pioneered the development of security software that works in conjunction with handheld "smart" devices or tokens. This software is available for nearly all of today's major operating systems and networks. During the last three years, Enigma Logic has also been delivering software that provides users with a "Single Sign-On". With this system, users need only a single UserId and password (or token) which they provide just one time. Users can then access multiple systems within the enterprise without having to provide authentication information again.

David Bauer (Bellcore) - Secure Distributed Computing for Heterogeneous
Operating System Environments

Enterprise Security is the new "buzzword" in the business community. No longer workable as a separately handled function, information security requires integration with the very foundations of the business operations. Ultimately, this integration requires application of security technologies into every Information Technology initiative and service. In this part of the panel, approaches to integration of contemporary security technologies into an Enterprise Information Technology architecture and resources will be explored. The goal is to initiate audience discussion into how best to approach Enterprise Security from a technological point of view.

PANEL: Debate of Critical Player Perspectives on MLS System Solution Acquisition Topics

Joel E. Sachs
Arca Systems, Inc.
10320 Little Patuxent Pkwy., Suite 1005
Columbia, MD 21044
410-715-0500

Panel Overview

Acquiring and developing an MLS (multilevel) system solution that results in an accreditable secure solution is not simple; moreover, there is debate and confusion as to what should be specified during the initial phases of an acquisition that will help all parties involved throughout the life of the program. Several MLS system acquisitions have already been deemed less than successful. A number of reasons have been suggested: integration of MLS products is not straight forward, defining mission requirements and mapping them to security and system solution requirements is difficult, and certification and accreditation is difficult and not consistently applied.

This panel is a continuation of similar panels conducted at the last two National Computer Security Conferences that focused on these issues. In past years, the panel discussed and debated a spectrum of issues along the life-cycle timeline associated the acquisition, integration/development, certification and accreditation, operation and maintenance of a MLS system solution. This was achieved through role-playing of the critical players in the acquisition process, as opinions varied depending on one's position within the process. Each of the seven panelists acted on the behalf of an identified role with which they were experienced. These roles included: End-User Organization, Program Management Office, Advising Security Agency / Certification Body, Designated Approving Authority, Systems Integrator, Security Engineering Subcontractor, Vendor.

This year's panel will use the role-playing technique again, will focus on five specific issues, and will address each, one at a time. The issues are:

- What critical items should be in place before release of an Request for Proposal [RFP] and, moreover, which items should be included in the RFP package?
- What should be done to address assurance with regard to (rapid) prototyping, particularly software prototyping? Is this simply an acceptance issue or is it more fundamental? Are prototyped aspects of a system fundamentally non-trustworthy?
- What steps should be followed for developed applications that require trust? When and who should take such steps? Are there additional assurance requirements that should be addressed, and if so, how?
- Should major emphasis be placed on the assessment of security-related aspects of engineering processes for requirements analysis, design, detailed design, product selection, code implementation, integration and test or end-results of these activities? What is the most effective and efficient mix for each and across them all?
- How should the accreditation of a new MLS system solution be addressed with regards to external systems directly or indirectly connected to it? What should be done about the accreditations of these external systems given the introduction of the new one? How should the accreditors of each of these systems work together? How can they add value to the acquisition and development phases of the new system solution?

The first issue focuses on concept definition, preliminary requirements analysis, and acquisition preparation. The next three deal with development and assurance issues. The last issue addresses the multiple accreditor problem, a current technical/political problem in the U.S.

Information is provided below which describes the roles of the critical players along with example issues and concerns for each critical player. A list of 25 questions and issues associated with pre-draft RFP, pre-RFP, pre-award, and post award milestones regarding specifying, procuring, and accrediting MLS System Solutions can be found in last year's panel description in the 1992 National Computer Security Conference Proceedings.

Panel Roles, Descriptions, and Areas of Concern

End-User Organization

The end user organization has a requirement for a system solution. The results of this procurement will be delivered to this organization for their use.

The main concerns of the organization are how to ensure that the end-users get what they want and need, that the system solution will be creditable, that it will fall within its budget and development and delivery schedule. End-user organizations usually understand functional requirements reasonably well but usually do not understand security and assurance requirements and security issues.

Program Manager's Office [PMO]

The PMO is the acquisition agency responsible for writing the RFP, awarding the contract, and supervising its execution. (Typically, a separate organization might be used to develop a system specification for the Statement of Work [SOW]. For the purposes of this panel, the player developing the specification will be considered merged with the PMO.)

The PMO's main concerns are system specification, cost, schedule, measuring the prime contractor's progress and compliance, and assuring steps towards accreditation are being taken. The PMO understands the functional requirements as communicated by the end-users, but is not likely to fully understand the security requirements, issues, and assurance needs that result from the mission and threat context.

Advising Security Agency / Certification Body

The Advising Security Agency is the End-User's and/or PMO's security arm. This agency helps monitor the progress of the program to ensure that security within the program is adequately addressed. The Certification Body gathers the assurance evidence and performs risk analyses on the system. (For the purposes of this panel, these two roles have been combined as often happens in practice.)

The main concern of both of these organizations is whether the delivered system meets the security requirements specified in the RFP and provides the required security functionality and assurance. The certification body must provide enough evidence to allow the accreditor to make a proper decision regarding the system's accreditation.

Designated Approving Authority [DAA]

The DAA is the individual responsible for the operational aspects of the system. It is this individual's responsibility to approve the system for operation.

The DAA's main concern is whether the system meets its operational requirements and its operational risk has been reduced to an acceptable level. Based on the evidence provided during the certification process, the DAA must make a decision whether the operational risk is acceptable given the evidence provided and the system's mission, and accredit or fail the system for operation. The DAA's

accreditation of the system is his indication that he feels the risk is operating the system is low enough or the operational need is high enough to allow the system to operate.

Systems Integrator

The Systems Integrator is responsible for the development and integration of the end-system as well as the management of all the subcontractors involved in the effort.

The main concerns of the system integrator are how to provide the required functionality, security, and assurance within the budgetary and time constraints stipulated in the integrator's proposal. Other areas of concern include how to manage the security engineering effort to produce a functional and usable system as well as how to handle the potential impact of requested changes to the end-system on system operations, security, and assurances.

Security Engineering Group/Subcontractor

Security Engineering is responsible for the security portion of the overall system development. This team is composed of internal systems integrator personnel, a security subcontractor, or a combination of both.

This team's main concerns are: how to relate component policies to the overall system policy, the trust requirements for each component, how to integrate trusted and untrusted systems, how to integrate multiple products into a single secure solution, and how to provide required assurance evidence. They may also be involved in determining the security requirements and policy, determining the appropriate assurance level, and how to provide assurance evidence.

Vendor

Vendors provide products that are used as part of end-user system solutions.

Their main issues are: how to relate their product features to the desired functionality and assurances needed within an MLS system solution and how to advise the systems integrator on the best use of these features and assurances.

Network Security Management -- The Harder Problem

A Panel Discussion

Panel Members:

Ronda Henning, Harris Corporation, Panel Chair

W. Earl Boebert, Secure Computing Corporation

James Galvin, Trusted Information Systems

Maj. Mike St. Johns, ARPA

Mark J. Schertler, National Security Agency

Col. Bill Thomas, USAF

Chairman's Statement:

In the wonderful days of yesteryear, before the advent of personal computers, there was a monolithic mainframe environment. Terminals were hard wired connections, quite possibly attached to a patch panel if connectivity to multiple systems was required. And it was good. There was one system audit trail, one set of user accounts to administer, one place to administer the system security policy. A human being was usually in the loop if you wanted to do anything that could be considered hazardous to the integrity of the security perimeter. Perhaps security wasn't perfect, but it was a relatively manageable entity with a defined perimeter attached to it.

That was then. We are now living in a world where nothing is simple. Network architectures are no longer tied to the mainframe environment, or even client/server architectures. Peer-to-peer networks of cooperative computing are becoming more commonplace. Distributed file systems living in wide area networks are not unusual in today's system architectures.

With more complex internetworks come more complex security policies. Defining where one network stops and another network starts is an arbitrary boundary, defined as "I have no control after we get to Point X." There may be a globally enforceable security policy that can be centrally administered, but, more frequently, the security policy is as distributed as the system architecture.

Which brings us to the topic of this panel. Our attempts at secure networking standards seem to omit or defer discussions of what network security management is all about. Some will say it is a function of the application using the network. Others will argue that the application can only apply the services provided by the underlying infrastructure.

There are no standards defining what network security management encompasses or describing the necessary management attributes for secure networking environments. Is this an oversight, or are the problems in distributed security administration too difficult for today's technologies? Are we focusing on the wrong problems entirely and making it too hard? Are there simple near term steps that aren't being taken? Is there a difference between secure network management and managing network security information? Is that difference theoretical but not practical in an actual operational environment? All these questions were posed to our panel, with the promise that the topic was wide open for them to express their own opinions on the area of most interest to them.

Col. Bill Thomas, USAF

How does an information systems director provide adequate security in an evolving enterprise-wide, client-server based information system? The answers get especially tough when the users of the network put special value on the sharing and availability of information and applications, when ad-hocracy is preferred to autocracy, and when you are committed to meeting the challenge primarily with commercial off-the-shelf tools.

The only core information these extended users may access is that pertaining their workgroup. Information specific to a work group generally may not be shared among groups. Moreover, because the facilities and systems supporting the extended users are not directly supervised by the core security managers, a higher degree of risk management is needed. Finally, many core users are asking for the secure virtual office, a way to extend their work space to virtually wherever they may be.

I believe focus in six key areas, all addressable with COTS technology, may provide a manageable level of security in this environment.

1. Stealthy tools for surveilling network activity at key points. Tools must also reduce the amount of data security managers have to deal with, flag significant security relevant activities, and provide a means to evaluate/limit damage once a security breach is suspected.

2. Graphical network security manager tools for configuring and observing system and network security parameters (passwords, permissions, etc). These tools must themselves be "networkable" to allow security managers across the network to function as one unified entity.

3. Public key based encryption. These can support user identification, authentication, and registry functions as well as the more traditional confidentiality of data while it is stored or transmitted.

4. Security enhancements and options to commercial protocols. The Message Security Protocol (MSP) is an example.

5. Security gateways for operating between domains of the network with differing levels of risk. However, we need alternatives to expensive "trusted operating system" based gateways. Techniques such as remote data caching, data deposit and pick up, and message content and internet address filters that only allow authorized connections are possible alternatives.

6. Object oriented applications and data as a way to address labeling and integrity. This approach would allow each data item to have its own security identity and this identity could remain intact even as objects are linked together into images or documents.

Above all, these technologies must be scalable so that the specific measure taken can be tailored to the cost and performance objectives for the specific risk that needs to be managed.

W. Earl Boebert, Secure Computing Corporation

The phrase "Secure Network Management" admits of several interesting interpretations, ranging from "Managing an Insecure Network in a Way to Get Security" to "Managing a 'Secure' Network in a Way to Avoid Losing the 'Security' You Already Have," with "Managing the Security Services in an Otherwise Insecure Network" and "Designing the Administrative Facilities of Security Services" somewhere in between.

I will speak to the last interpretation: the functions which support and administer security services that operate in an unreliable and potentially hostile environment.

The service of greatest interest is cryptography. There is a misconception that cryptography solves security problems. It does not. It just transforms a problem of protecting a large amount of data into the problem of protecting and distributing a small amount of considerably more valuable keying material.

There is a second misconception that asymmetric or "public-key" algorithms solve the problem of key management. They do not. They just transform the problem of transmitting and protecting secret keys into a problem of managing certificates.

The latter transform is of interest because it suggests that a paradigm shift would be of benefit. Instead of looking at a network which uses asymmetric algorithms as a traditional key management problem, it is useful to view it as a distributed, capability-based operating system which controls access to resources based on distinguished "tickets." Such a view, in my mind, properly diminishes the importance of the cryptography and heightens the importance of such difficult problems as revocation.

James Galvin, Trusted Information Systems.

For anyone with a network with more than a few hosts and a router, the value of network management is apparent. If the network includes remote hosts and routers, then the ability to interact with these devices from a central location is preferred.

Version 1 of the Simple Network Management Protocol was an effective means by which network managers could interrogate their various network devices. However, its support for security was inadequate, since network eavesdroppers could trivially learn the secrets being used. As a result, network managers typically did not use it for controlling their network devices. In other words, while it was possible to determine the status of a device and what the device was doing, it was typically not possible to change the activities of a device.

The SNMP Security Protocols were a significant enhancement to the SNMP. In particular, they introduced a new concept: party identifiers. These party identifiers were a fundamental component of the access control lists and Management Information Base (MIB) views (the subset of the set of all objects accessible via the SNMP), representing a significant departure from the community string model of version 1 of the SNMP and providing the basis for effective security services that were not easily spoofed by network eavesdroppers.

Version 2 of the SNMP incorporated all of the features of the SNMP Security Protocols, enhanced according to implementation experience, and added some new features to the SNMP protocol. Key and secret management, a concern whenever security services are supported, are handled by the SNMP directly. Thus, no additional network services are required. Effective, secure network management is possible.

Mark J. Schertler, National Security Agency

Network security management is a distributed, and, therefore, very complex problem. Comprehensive network security requires the coordination and control of a number of security services. These security services include authentication, access control, data confidentiality, data integrity, and non-repudiation. Each security service can be implemented by a number of security mechanisms. How these security services are utilized depends on a network security policy. The network security manager should be flexible enough to allow the changing of individual security mechanisms, without jeopardizing security. Reasons for changing security mechanisms include changes in security policy, upgrading to an improved security mechanism and replacement with a new or different security mechanism. Interoperability of the network security manager and the security mechanisms' managers and management communications between systems are very important areas of research. Standardization is the way to achieve this interoperability.

The International Organization for Standardization (ISO) Open System Interconnection (OSI) provides a protocol standard for management communications known as the Common Management Information Protocol (CMIP). OSI is also standardizing a number of management services including object management, security alarm reporting, security audit trail recording, and access control. These standards provide a basis for a standardized network security management function, but much work needs to be done

A research effort is underway to utilize the OSI standards where applicable and develop a network security management framework that will provide standard interfaces between the overall network security manager and the individual security mechanism managers. The effort also includes providing interfaces and protocols between network security managers on different systems within a security domain.

Mike St. Johns, ARPA

I've been involved with networking and the Internet since late 1985 when I joined the Defense Communications Agency as a project manager on the Defense Data Network. One of the things that frustrated me then and still frustrates me now is the lack of available security within the networking infrastructure. My emphasis here is not primarily the privacy aspects of security, but more along the lines of integrity, authentication and authorization as tools for ensuring reliable operation of networks.

The Internet is a loose confederation of networks, networking organizations and customers; look up the definition of anarchy in the dictionary and it says "See 'Internet'.". In the past we could count on having our neighbors be good neighbors. They wouldn't knowingly trespass, they'd help you when your network was crashing down around your ears and they'd do what they could to contribute to their share of the common structure in the form of information, assistance or possibly other intangibles. It was rare to think of someone else on the Internet as a possible "enemy." The problems that did occur were generally the result of accidental actions such as configuring your router with the network addresses that were used as an example in the manual. The Morris worm was a notable exception.

This utopian Internet is no more, a victim of its own success. The current "Research and Government" internet which supplanted the original "Research Only" internet is in turn evolving into a true full service Commercial, Government, and Research entity. Commercial organizations are joining the Internet at a greater rate than either the government or research sectors and will within a few years probably have the majority of systems and networks connected to the Internet.

We'd probably already be there except for one thing: Anecdotal evidence seems to suggest the number one reason companies give for not connecting to the Internet is the lack of security. Those companies which do connect often spend lots of time and effort crafting firewalls to protect them from the wilds of the greater Internet. This tends to be self defeating -- the companies limit themselves to a small subset of the available services by firewalling and are slow to adopt new services when offered.

Commercial concerns with respect to security have some basis in fact. Most of the protocols in use today in the Internet have only the most basic mechanisms for ensuring "secure" operation. That isn't to say they aren't reliable for the environment they were defined for, but that they aren't well protected against accidental misconfiguration or malicious attacks.

My goals for security within the ARPA networking research program are to try and make sure basic security services are available about on the same basis that networking services are: integrated and relatively simple to use. Because of the need for integration, simplicity and ubiquity, I'm not looking for the high assurance military grade systems which have customarily been DoD funded but for approaches which provide some measurable amount of additional protection for a reasonable price/performance penalty; "good" security vs. "perfect" security. This doesn't prevent some of the techniques from being used in high assurance systems, but does provide a baseline from which to start.

The initial emphasis is on protection of the infrastructure from denial of service. This means adding authorization, authentication and integrity services to the routing protocols, directory services, communication link protocols (PPP for example), network management and other protocols and services which have a direct impact on the availability of the Internet. Much of the work necessary for doing this is also applicable to use by end systems, especially the work on authentication and authorization. Future work will follow up on this and provide security assistance services for user and system applications.

PANEL SUMMARY

- Title:** "Application of INFOSEC Products on Wide-Area-Networks"
- Overview:** Classified programs are relying on commercial-off-the- shelf (COTS) INFOSEC products to allow their local area-networks to interconnect securely over public and private wide-area-networks. This panel will explore how this is being accomplished today on a major ARPA program, the Defense Simulation Internet (DSI), using the Network Encryption System (NES) an NSA endorsed commercial-off-the-shelf INFOSEC product. Each panelist has a unique role in this program and will discuss some of the key issues from their own organizational perspective. The Panel Chair has led the Lockheed Corporation in implementation of over 15 NES-based classified networks. Panelists include representatives from the following organizations: Motorola - the INFOSEC product developer, NSA - the INFOSEC product endorsement agency, BBN - the systems integrator for DSI, and Trusted Information Systems - the accreditation support contractor for DSI.
- Chair:** Joyce Capell - Lockheed Missiles & Space Company, Inc.
- Panelists:** Ernie Borgoyne - Motorola Gov't Systems & Technology Group
Blaine Burnham - National Security Agency (NSA)
Russ Mundy - Trusted Information Systems, Inc. (TIS)
Mark Whitney - Bolt, Barenek & Newmann (BBN)

**PANEL CHAIR
JOYCE CAPELL
LOCKHEED MISSILES & SPACE COMPANY, INC.**

Ms. Capell is the Network Technical Specialist for the Computer Security Group at Lockheed Missiles & Space, Co. in Sunnyvale, California. She has pioneered the use of the Motorola NES not only for her company but throughout the Lockheed Corporation. Under her guidance, the Corporation tested the NES over public wide area networks to establish interconnectivity and performance characteristics. The results of that testing program appear in the NCSC conference proceedings, as well as being presented at a separate conference session .

It became apparent to Ms. Capell that users throughout the Lockheed Corporation were experiencing similar NES network implementation issues. As a result she organized a workshop in April of this year to bring people together to resolve these issues. Over 50 Lockheed people attended, representing the technical, procedural, and approval aspects of NES network implementations. NSA, Motorola, and Defense Investigative Services (DIS) also sent representatives to the workshop. To help future NES network implementors Ms. Capell is now is in the process of authoring an "NES User Guideline Document".

PRODUCT DEVELOPER
ERNIE BORGOTNE
MOTOROLA GOVERNMENT SYSTEMS & TECHNOLOGY GROUP

This presentation will focus on some of the NSA support and infrastructure issues affecting the Network Encryption System (NES) family of Type I INFOSEC products.

The NES was developed by Motorola and endorsed by NSA through the Commercial COMSEC Endorsement Program (CCEP). The NES is the first product endorsed by NSA that meets the Secure Data Network System (SDNS) Standard, and uses key material provided by the NSA SDNS Electronic Key Management System (EKMS) Central Facility.

The modular open architecture of the NES, with its strict adherence to standards, allows this security platform to easily incorporate commercially available technology, including trusted software. This open architecture provides a benefit to the system integrator and end-user in that new products can be made available much faster than ever before.

However, the successful deployment of INFOSEC products depends on a solid NSA infrastructure which includes: control and management of the SDNS standard to assure new product conformance and backward compatibility, timely release of the functional capabilities and services that are promised by the NSA EKMS Central Facility, and a stream-lined endorsement process which allows new product spin-offs to be released by NSA for operational use more quickly.

This presentation will focus on Motorola's experiences with the NSA infrastructure, in particular the process for approval of NES product enhancements.

SECURITY TECHNOLOGY CONTRIBUTOR
BLAINE BURNHAM
NATIONAL SECURITY AGENCY

The panel represents all the major responsibilities for the identification, design, integration, certification, and use of automated information systems (AIS) that need to incorporate, at least by directive, security as part of the AIS policy. It is very important for us to share perspective of each others' roles and responsibilities, for there are both significant gaps and overlaps in that perspective. As a consequence, some things are being worked by multiple agents and hence authority conflict is fostered, and some things are not getting worked at all or in the right order and hence falling through the cracks.

Dr. Burnham will try to paint a view of the relationships among the various new constructs that are coming out of the Federal Criteria and how NSA would like to capitalize on those constructs to aid and facilitate the incorporation of trusted technology into the development of secure systems. He will show how these constructs underscore a very significant evolution in the way NSA plans to accomplish its INFOSEC role in the future - in particular, how these constructs foreshadow a fundamental model change in the INFOSEC business model and the NSA INFOSEC relationship with product vendors and system integrators and system certifiers and accreditors.

Overall, Dr. Burnham will present a picture of NSA INFOSEC trying to establish its position in the process of secure systems generation that shows NSA contributing the wisdom of its core technologies and working as a team player to accomplish the end objective. It is important for all the players to understand that, with this relationship, will come the potential for the user to become much more knowledgeable with respect to the risks, residual or otherwise, to which he remains exposed.

**ACCREDITATION SUPPORT
RUSS MUNDY
TRUSTED INFORMATION SYSTEMS**

Mr. Munday will discuss the accreditation process and some of the difficulties encountered in the diverse DSI environment. This will include some suggestions on how to handle accreditations in this environment. Department of Defense (DoD) directives require that automated information systems (AISs) be accredited by a Designated Approving Authority (DAA). The Defense Simulation Internet (DSI) has been accredited under the prescribed process. Since the DSI is a network which interconnects AISs from all of the military departments, several defense agencies and multiple contractor sites, the accreditation is based on the interconnection of accredited AISs described in DoD Directive 5200.28. This required establishing a set of rules for connection and secure operation on the DSI as well as the mechanisms to ensure the continued secure operation of the DSI. A Memorandum of Agreement (MOA) between the DSI DAA for each user site provides the foundation for the secure operation of the overall system.

How this was achieved and by what means will be the topic of this discussion, along with description of some of the difficulties encountered along the way.

**MARK WHITNEY
DSI SYSTEM INTEGRATOR
BBN**

As the integrator for the Defense Simulation Internet (DSI), Mark Whitney of BBN, will discuss how the Motorola Network Encryption System (NES) INFOSEC platform was selected and the challenges BBN has had in integrating the NES into the DSI. The presentation will include the description of the way that DSI uses NESs on a global scale and how DSI manages this aspect of operations.

This discussion will address the real-time performance requirements DSI has for an INFOSEC platform and current operating procedures and constraints of the existing network components. Included will be a short term approach to providing additional secure communications to sites that have higher traffic requirements than the NES can currently handle. This involves using multiple NESs.

Overall network management is a major concern for DSI and secure operations is one of the major areas which will be addressed over the next 6 months. The requirement for an Electronic Key Management System and why it is needed to enhance RED management capabilities and simplify logistics will be discussed. BBN is hopeful that this system will be available soon for implementation to simplify the administrative and operational costs of running a large secure network in multiple Communities of Interest (COI).

DSI and BBN are also anxiously awaiting the delivery of the "Enhanced Network Product Server" (ENPS) from Motorola to help manage the NES remotely from a central network management center. Plans are to integrate the ENPS into the overall secure network management strategy. The requirements for this will be discussed in broad terms.

INFOSEC Design and Certification Initiatives Panel Update

Chair Capt. William Feteck

Speakers

Ruth Willard (NSA)

Brian Koehler (NSA)

Capt. William Feteck (NSA)

This panel session will present NSA's INFOSEC engineering and system certification efforts. To meet customer system security requirements, the Office of INFOSEC Systems Engineering within the Information Systems Security Organization (ISSO) will specify how their work relates to the community.

This panel is an update from a previously well received one presented at last years conference. This panel session will include descriptions of the technical guidance being developed for the user community. Three areas of technical guidance include the Unified INFOSEC Architecture activities, system engineering techniques, and the certification process.

Capt. William Feteck (NSA)

Currently the NSA-sponsored C&A working group is developing a *Certification and Accreditation Process Handbook* which will aid the certifier in performing a certification of a system. This handbook will outline the steps that are needed for a successful certification at an appropriate level of assurance required for the system operating in a particular environment. The initial draft has been distributed for comment and is anticipated that an update will be ready in July for a larger distribution. The process outlined in this document can be used to certify and accredit all systems, ranging from sensitive/unclassified to Top Secret/SCI. Although a standardized process is being developed, the certification effort for a particular system is tailored depending on the level of assurance required for availability, confidentiality, and integrity.

Ruth Willard(NSA)

The NSA Office of INFOSEC Systems Engineering has issued a draft Information System Security Policy (ISSP) Guideline for use and review. The ISSP guideline contains explanations and approaches to help someone write a system level security policy. It is intended to assist the novice or experienced security practitioner in their efforts to integrate mission needs, mission threats, umbrella policy guidance and system-specific security needs into an information system security policy. I will discuss the feedback from 6 months of use and our plans for further versions of the ISSP.

We have also prepared a Security-Relevant Mission Needs Guideline (SRMN). The SRMN provides the user with an approach for recognizing security considerations relevant to the identification of mission needs. It focuses on general statements that describe the protective or secure condition that is required to accomplish the mission. The guideline describes the role of mission needs in the secure development process, delineates the methodology employed in formulating security-relevant mission needs, and focuses on establishing these needs so that a system security perspective is maintained throughout the system development process. I will discuss its status, and plans for future guidelines in other INFOSEC design areas.

Brian M. Koehler (NSA)

Current activities in the INFOSEC Architecture Division of the Office of INFOSEC Systems Engineering are focussed on producing an INFOSEC Domain model that captures the fundamental structure of information security relating abstract security concepts, principles, characteristics, and disciplines. This model is intended to be an implementation independent reference for the design of a diverse collection of architectures that can be represented as generic, logical, or system specific security structures for information processing environments.

The presentation includes a summary of the underlying premise for the research and development of an INFOSEC Domain Model and its origins, highlights of the types of models & modelling techniques used, tools to assist an INFOSEC architect, lessons learned in application to customer INFOSEC problems, parallels to trends in system architecture/software design & reuse, and the importance of measurement criteria to support security architecture design. Relevance of this effort to an overall information systems design and development process will also be highlighted.

SECURITY ISSUES FOR THE SECURITIES INDUSTRY

Sally Meglathery
New York Stock Exchange
212-656-5579

PANELISTS	J. Goodson	Morgan Stanley
	P. Rippley	Bear Sterns

This panel will discuss the security issues that are of concern within the Securities industry.

It is the responsibility of each firm in the Securities Industry to provide adequate security for its automated systems. The security requirements for confidentiality, integrity, and availability of information remain the responsibility of the top management of the firms, no matter which technologies are used to support their business. Issues of interoperability, authorization and authentication, and encryption must be dealt with in a way that provides maximum security for the firm with minimum impact on the user.

How to achieve these requirements in a multiplatform environment is a challenge that the Data Security Officer (DSO) must deal with on a daily basis. This is complicated further in an environment where networking between branches, competitors, and exchanges is necessity. The DSO must ensure the quickest possible turnaround while at the same time ensuring the confidentiality and integrity of the transactions. In addition, since in some cases the users of the systems are not technically proficient, the systems must be as simple and quick to use as possible and security must be virtually transparent.

Historically, the securities industry has not been as concerned with network security as have certain other industries. Perhaps this was logical since traditionally the systems of securities have been quite insular. No more! The implementation of massively distributed systems, the linkage of disparate networks, and electronic connection to customers all over the world have all combined to make network security a serious issue and concern for the firms.

How the DSO faces up to these challenges, and the solutions they have devised will be the topics of discussion. The panelists will discuss their security concerns, the environments they work in, their solutions, and what the industry, as a whole is trying to accomplish in order to ensure the highest level of security with the minimum intrusion to the user.

This discussion should also demonstrate the parallels between the security concerns of the securities industry and those of both sectors of the federal government.

Virus Attacks and Counterattacks

Real-World Experiences

James P. Litchko, Chair
Director of Business Development
Trusted Information Systems, Inc.

Mr. Eric Ratliff
Senior Security Analyst
Input Output Computer Services

Mr. George Wellham
Assistant Vice President
MNC Financial, Inc.

Ms. Louise Mandeville
LAN Administrator and Accounting Manager
Miller, Balis & O'Neil, P.C.

Over the past two decades, viruses have migrated from fiction to our working environment. With the widespread availability of virus tool kits, virus cookbooks, and creative programmers, experts believe that the number of unique viruses now circulating is over 3000 and suspect that new viruses are being developed at a rate of two a day. Some experts project that there will be over 10,000 viruses by the year 2000. With corporate profits and credibility now have a direct relationship to the reliability and availability of information systems over networks, a single virus attack can now result in losses totalling millions of dollars.

What is required is an open discussion of real-world experiences to increase the understanding of the threat and effective prevention and reaction. In the past, few would openly admit that they had been a victim of a virus attack, and fewer would openly talk about the specifics of an attack. All of the articles and discussions about the expected losses and effectiveness of countermeasures are very interesting but very impersonal, and they do not truly answer the real questions:

- What really happens during and after an attack?
- How effective were the countermeasures?
- Where were the pit-falls?
- What was the real impact of the virus attack (i.e., financial, moral, cognitive)?
- Are there effective pre-attack legal countermeasures?
- What were the near-term and long-term effects?

Each of the panelists was a victim and survivor of actual virus attacks on operational computer

and network systems in both commercial and government agencies. During this panel session, each member will describe the attack on his or her system and the actions taken to counter-attack. The corrective actions that will be discussed by the panel will include those that are technical, procedural, organizational, and legal. Through interactive discussions between the audience and the panelists, the panel will provide their perspectives and answers to the preceding questions and to additional audience questions.

This panel was very well received by over 400 attendees during the 1992 National Computer Security Conference and receive very good press coverage. Panelists will provide additional information on what they have learned and what new issues they have confronted since the the last conference.

TERROR AT THE WORLD TRADE CENTER

Sally Meglathery
New York Stock Exchange
212-656-5579

Maryellen Kelledy
Deloitte & Touche

This panel will explore the vulnerabilities to business continuity that were revealed by the bombing of the World Trade Center, and the lessons learned. The relationship of data and information technology to other business functions and the need for comprehensive contingency planning will be examined. Lessons learned from other recent disasters will also be discussed.

On Friday, February 26, when the bombing occurred, Deloitte & Touche was right in the middle of their annual "busy season," the time of year when staff is bustling to meet annual audit and tax deadlines.

As is well known, the bomb was hidden in a rented moving van, strategically parked in the garage under the trade center. The damage caused by the bomb was extensive. It destroyed the command and control center so that there were no emergency lighting or communication systems. Thousands of people were forced to evacuate in the dark.

This occurrence forced the management of Deloitte & Touche, as well as many other businesses resident in the World Trade Center, to activate their contingency plans. Many lessons were learned from our experience, and there were certainly key elements that attributed to our successful recovery that all businesses should address when thinking about business interruption.

But this was just one of several events that hit the downtown New York area in recent years. Many of these events will also be discussed along with solutions for how to successfully recover from a variety of outages such as loss of communications and power.

Maryellen Kelledy has been with Deloitte & Touche for over six years in the Computer Assurance Services Group in New York. Maryellen has extensive experience in the development and implementation of Data Center and Business Continuity Plans for a number of clients within the insurance, banking and brokerage industries. In addition, she has been involved in the development and implementation of Business Continuity Plans for clients directly impacted by the World Trade Center bombing in February of 1993.

Panel: CONTINGENCY PLANNING IN THE 90's

Chair: Irene Gilbert, NIST

**Panelists: Mark Wilson, NIST
Martel Perry, Computer Technology Solutions, Inc.
Sadie Pitcher, Department of Commerce
Tom Terry, AT&T**

Contingency and disaster recovery planning can no longer be viewed as the responsibility of the data center. Statistics indicate that nearly half the computer and communications interruptions in the past few years have impacted business functions that resulted in failure for some organizations to provide a product or service to its clients. For many organizations the data centers were able to recover from unexpected outages while the business areas had difficulty recovering.

In today's business environment, senior executives cannot except anything less than maximum operational efficiency. Managers must shift from the 1980's approach to contingency planning and implement a corporate-wide approach for meeting the organization's business objectives and protecting its assets. A corporate-planning approach focuses not only on ensuring the continuity of business functions but in reducing financial loss and regulatory pressure as well.

Notwithstanding the level of government or organizational culture, a functional approach is favored. This means, like all other business functions, contingency planning should be rooted within the full life cycle of any unexpected outage (e.g., mitigation, emergency preparedness, backup response, and recovery).

What steps are needed?

This panel will explore new technologies and innovative approaches to contingency planning from an organizational, service provider, and user perspective. The planning process is emphasized, enumerating on ways in which to focus senior management attention on developing an organizational strategy and policy for contingency planning. Equally important, the panel will explain how users can develop functional level plans in the context of the organization's policies and procedures.

The panel advocates a team approach for writing plans at each functional level and coordinating the plans with all employees. The panel urges regular staff training and increased awareness of the range of vulnerabilities confronting an organization. All the panelists are experienced in contingency planning and eager to share their knowledge and point out common errors to avoid when planning.

AN ORGANIZATIONAL APPROACH TO CONTINGENCY PLANNING

Mark Wilson, National Institute of Standards & Technology

Recent front-page and evening news coverage of disasters has put a spotlight on the importance and necessity of contingency planning. The flooding of Chicago's Loop, Hurricane Andrew's devastation of parts of Florida, and the bombing of the World Trade Center in New York caused an incredible amount of damage and almost immeasurable financial loss to many affected businesses. However, these are only the high-visibility examples of natural disasters, man-made disasters and politically-inspired disasters businesses face. These represent only the tip of the proverbial iceberg. For example, recent estimates show that there is a business site fire every four and one-quarter minutes in this country. A far more common list of potential threats, including water damage, power and communication failure, earthquakes, tornadoes and other severe storms, and vandalism can cause as much disruption to normal business functions, and can cause as much financial loss to businesses.

An organizational approach to contingency planning stresses a dynamic, business-wide acceptance (beginning with senior management commitment) of the process to reduce risks to critical business functions, and to effectively plan for the response to, and recovery from disasters. Management must understand and accept the costs of contingency planning as being necessary to better protect the business against the inevitable disruption to its operation. A policy statement containing the corporate philosophy toward contingency planning will set the proper business-wide tone in planning for the time "when" a disaster occurs, not "if" one occurs. Management acceptance of a certain amount of risk to critical business functions - an acceptable level of risk - should be understood by all players in the contingency planning process.

The Contingency Planning Coordinator/Manager must work closely with management throughout the planning process, providing briefings and updates on risk analysis, business impact analysis, defining critical applications/functions, analyzing and selecting alternative processing strategies (recovery strategies), and testing or exercising those strategies.

Functional managers (application owners) must work with the Contingency Planning Coordinator/Manager to identify, reduce or mitigate risks affecting their functions, and develop and test disaster recovery/business resumption plans. Functional managers must be responsible for the recovery of their functions following a disaster.

**Business Continuity and Contingency Planning
Versus
Response, Resumption, Recovery, and Restoration
From a Service Provider's Perspective**

**Martel Anse' Perry
Computer Technology Solutions
1400 Shepherd Street, N. E., Suite 257
Washington, DC 20017
(202) 529-8419**

Over the past twenty-five years, management, information systems, and computer security professionals have discussed, changed the names, and reinvented approaches to Business Continuity and Contingency Planning. During this time, our technology environment and computer profession have changed dramatically. When computers were kept in ivory towers on Mount Olympus, the organization felt that the responsibility for operation, maintenance and care of software, hardware, and data was the responsibility of what was then known as the Central Computer Department. With the introduction of personal computers into the work place, the responsibility for business continuity and contingency planning has become even more difficult to assign to the appropriate level in an organization. Management as well as users of information processing systems see the need for planning but are not sure where their respective roles begin and end.

Service Providers, on the other hand, can articulate their roles and responsibilities clearer than management and users of information processing systems. Service Providers are those entities that are responsible for the operation and delivery of services required for information system processing (e.g., power, water, central processing center, communications, and more).

Business continuity planning is the analysis and documentation of critical elements and processes that are needed to operate a business and the implementation of safeguards that can reduce the chance of a business interruption or loss. Contingency planning is the master plan used for preventing and reacting to a business or operational interruption. These two processes should be driven from the executive management level of an organization. The four subcomponents of a contingency plan are emergency response, operation resumption, processing recovery, and system restoration. Each of these action plans detail who, what, when, and how.

From the Service Provider's perspective, the organization being serviced determines the business continuity and the contingency planning approach and requirements. The Service Provider identifies its emergency response and operational resumption plan to the user organization. The users are responsible for determining the processing recovery and system restoration requirements. It is the responsibility of the Service Providers to inform the organization and users of any limitations or capabilities that could affect the business continuity and contingency planning process.

CONTINGENCY PLANNING GUIDANCE FOR APPLICATION OWNERS & USERS

Sadie Pitcher, U.S. Department of Commerce

Application owners, data owners, or functional managers, regardless of the title used, play an increasingly important role in contingency planning. Today, application owners often find themselves "owning" and operating their own systems, either distributed systems, local area networks (LANs), microcomputers, or some complex combination of these systems. Application owners are now far more likely to have an in-house specialist who serves as the system administrator, technical support specialist, customer service representative, and telecommunications technician. Although technical expertise probably exists within the data processing facility of an organization, if indeed, the organization has a DP facility, the application owner is now faced with having to identify and document the existing systems and the telecommunications network; identify risks and vulnerabilities to the systems and network; select, fund, implement, and monitor controls or countermeasures; and plan for the recovery of the systems, critical application(s), and communications network.

The application owner/functional manager will also be directly involved in planning for the recovery of non-information technology (IT) functions. Workplace or office recovery has been identified as a relatively new and serious concern during and after the latest rash of disasters in Los Angeles, Philadelphia, Chicago and New York. Voice communications needs rise dramatically - as much as four times that needed during normal operations. Effective voice communications planning within the organization and with the telephone company will be necessary to avoid business communications paralysis. Vital records and current business/mission-critical records (personnel, payroll, contracts, engineering blueprints, legal documents) must be identified, protected, backed up (where necessary and possible), and included in contingency planning as critical resources.

Application owners will also need to consider "the people issue" in the planning, recovery, and business resumption. The same people that management will rely on to carry out the recovery and resumption of critical functions are those who may be most adversely affected by the disaster, especially a regional disaster which threatens or actually harms families and property. As recent disasters have shown, people suffer from injury and trauma during most disasters. The initial response to a disaster, recovery from a disaster, and resumption of critical business functions will not be "business as usual". Contingency planning must take this into account.

The Evolution of Disaster Recovery Planning

Tom Terry
Bell Atlantic

This presentation covers the concept of value added preparedness planning and provides information on requirements for integrated vendor selection and solution development. A review of methods, past and future, in disaster preparedness planning is included. It provides information on using an integrated approach to managing the response to the corporate demand for full business resumption planning.

Tom Terry is the manager of the Bell Atlantic "CommGuard" business resumption services. He coordinates the functions of all Bell Atlantic companies in meeting customer requirements in the arena of preparedness planning and disaster recovery.

Mr. Terry will present Bell Atlantic's view of evolving disaster recovery needs, based on experiences with over 300 companies in their evaluation of their preparedness needs. He will discuss the focus required for full preparedness, the need for organizations to be certain that they have a complete plan, and the types of services and integrated solutions that have been most useful to customer companies.

ON A BETTER UNDERSTANDING OF RISK MANAGEMENT TECHNIQUES

Chair: Stuart W. Katzke, NIST

Panelists: **Ali Mosleh, University of Maryland**
 Deb Bodeau, MITRE Corporation
 Richard Baskerville, Binghamton University

Risk Management is an iterative process that helps to ensure reasonable steps are taken to protect automated information systems (AIS) and critical business functions. Managing risks requires identification of threats, their impact, and severity under uncertain conditions. Risk analysis is the most widely discussed technique for justifying the cost of security controls and acquisition in management. While many manual and automated methods have been developed to carry out risk management activities, there has been no good way to assure their completeness or accuracy.

The focus of this panel is to provide a better understanding of a very complex subject. Panelists will explore three generations of systems security development methods. Strengths and weakness of first and second generation methods are discussed along with important modeling challenges to designers of third generation risk analysis methods. Considerations for discovering the wide variety of economic facts that systems owners must consider when adopting or rejecting security controls for information systems are presented.

An array of risk analytic techniques that can be used to evaluate risks in computer systems and environments will also be discussed. Discussion centers around how to use these analytic methods and how to properly interpret their results. Even more, major differences in purpose and exposition will be addressed. Relatively new techniques devised in the last decade are explored along with older, more tried techniques.

Finally, an alternative model for determining necessary security controls for information systems is presented. This discussion profiles the functionality and attributes of this expert risk management tool. Considerations for coping with uncertainty and probabilistic situations will be addressed. Panelists, based on years of experience as security researchers, will suggest methods for improving risk management activities in industry and government.

THE RISK ANALYSIS BLINDSPOT: THE THREATS IN SECURITY

Richard Baskerville
School of Management
Binghamton, New York 13902-6000

The environment of many large organizations is changing as they compete against smaller or more limber firms in increasingly globalized marketplaces. In response, some large organizations are moving toward new organizational forms that depend on flexible networks of small computers. These dramatically important collections of small computational assets do not seem to be amenable to many forms of security that are traditional in mainframe computer centers.

But how do we know this is true? In its current form, risk analysis does not provide management with adequate information about the costs of implementing a control. For example, risk analysis permits us to calculate the value of physical access controls for a mainframe computer center. With automated support, risk analysis can calculate the value of physical access controls for each node on a large local area client-server network. However, current risk analysis techniques leave us to decide independently whether the costs of securing every office will be excessive.

Risk analysis currently focuses narrowly on the justification, or benefit of controls. Risk management should involve an equally close consideration of the costs of security controls. *Where corporate flexibility is growing paramount, what are the risks that the constraints imposed by security controls will cripple the system functionality? Can an information system strangle on its own security?*

Two areas of research need further development in this arena. First, risk management methodologies need to develop metrics for measuring the impact of controls on the information system, and more broadly, on the organization. Second, security researchers need to develop a new range of effective, low-impact safeguards that retain organizational flexibility while adequately protecting information assets. With the growing movement toward organizational reengineering, and a concomitant shift in the value of information assets, computer risk management may become a primary, not a secondary, element in organizational survival. A broader focus in computer risk management would be an essential advantage for these new organizational forms.

Deborah J. Bodeau
The MITRE Corporation
202 Burlington Road
Bedford, MA 01730
(617) 271-8436

The Role of System Descriptions in Information System Risk Analysis

Before analyzing or assessing risks to an information system, the risk analyst must gather information about the system and its operational environment. As the number and variety of risk analysis tools increases, the risk analyst no longer has to decide what information to gather; this is determined by the tool developer, who defines the parameters the tool uses. Our experience in developing and using the Analysis of Networked Systems Security Risks (ANSSR) tool, as well as in using other risk analysis tools and methodologies, has led us to observe that information-gathering is the most labor-intensive part of the risk analysis process.

This might suggest that the tool developer try to identify only those factors that will be used in assessing risks or communicating the results of the analysis. For example, for assessing disclosure risks due to deliberate attack in ANSSR, key factors include the attributes of legitimate users: how many there are in total and logged-on at any time, how reliable they are, and what their capabilities are.

However, the information gathered to perform a risk analysis can be used by other engineering activities, such as architectural analysis or development of an information system security plan. The tool developer thus must balance the goal of minimizing input parameters with the goal of making the data entered into the tool reusable by other tools or analysis techniques.

Thus, the tool developer must have not only a clear and concise model of risk, but also a model of information systems which addresses how an information system can be decomposed or described, how it depends on or interoperates with other systems, and its operational environment. This model is influenced by the scope of the risk analyses the tool is intended to support. For example, a functional decomposition which ignores architectural and design detail can be useful for analyses of denial-of-service risks, while an architectural description which emphasizes the allocation of security features to components may be more appropriate for analyses of disclosure risks.

Methods and Issues in Risk Modeling for Computer Security

Ali Mosleh

Center for Reliability Engineering
University of Maryland, College Park

ABSTRACT

Over the past fifteen years we have witnessed the emergence of a relatively large number of procedures, methods and computer codes for performing computer security risk analysis. The level of sophistication the methods and the power of the computer codes to implement them have improved over the years. Nevertheless, most methods and tools lack a disciplined approach to identifying, representing, and assessing the likelihood of risk scenarios. In other words what is often presented as a methodology for performing risk analysis is a combination of the developers intuition, ad hoc techniques, as well as rigorous methods. This applies to almost all key building blocks of risk analysis and risk management process as identified in the NIST Framework (e.g., Identification of Scope and Boundary of the Analysis, Analysis of Risk Scenarios, and Development of the Risk Measure.)

This situation significantly limits the ability of the whole discipline to gain acceptability as a serious scientific approach to managing risks associated with information security. The minimum requirement for establishing the validity of the results and recommendations of a risk analysis is the consistency of its results with the input to and basic assumptions of the assessment process. This consistency however cannot be verified unless the models relating the input to the output are at least internally consistent and based on a set of rules (such as logical or mathematical relationships) of application. examples are event trees, decision trees, and fault trees for identification of risk scenarios, and use of probability and calculus of probability for the assessment of their likelihoods.

Clearly the internal consistency is only a minimum requirement. The results of the risk analysis must also be externally consistent. That is they must represent the objective reality both in terms of the nature of possible risk situations, as well as in estimation of their frequencies. As a result, it is not enough, for example, to say that the frequencies of some threats are based on expert judgment without providing information on the method of expert selection, elicitation process, and calibration.

For an effective evaluation of various methodologies with respect to the above criteria it is necessary to catalog different risk analytic methods which have been or could be used in the computer security area. It is also important to identify their domain of applicability, strength and weaknesses, as well as compatibility with each other. The objective of this paper is provide an overview of the risk assessment theory, and discuss the existing modeling techniques for various parts of risk models. Examples of the methods reviewed are influence diagram and graph theory, fault and event tree methods, petri net approach, and decision tables. On the quantitative side the merits of probabilistic and nonprobabilistic methods will be discussed. The domain of applicability, strength and weaknesses of the classical and Bayesian methods are also discussed. Finally the relationship between various techniques and their potential compatibility will be explored.

PANEL SECURITY AWARENESS, TRAINING, AND PROFESSIONALIZATION: STATUS REPORT

Panelists

Mr. Dennis Gilbert, NIST, Chair
Ms. Dorothea de Zafra, PHS/FISSEA
Mr. Richard Koenig, (ISC)²
Dr. W. (Vic) Maconochy, NSA
Mr. Raymond Olszewski, CISS
Ms. Joan Pohly, USAF

PANEL OVERVIEW

Public and private sector organizations understand the need for cost-effective protection of information technology (IT) resources as part of an overall resource management strategy. Relevant security education, awareness, and training for all involved with IT is a vital element of the protection equation. Another vital element is having knowledgeable, highly qualified staff to ensure that security and control measures are consistent with federal directives and with industry and management imperatives. The acceptance of these factors and an evolving consensus regarding a common body of knowledge (CBK) for IT security, point to the emergence of a new career field. A number of independently initiated efforts are currently underway that are intended to provide a better definition of educational requirements, position standards, and testing criteria for security practitioners. Many think that there are opportunities associated with coordinating and focusing these efforts as part of a unified government/industry strategy.

This panel presents a status report on IT security awareness, training, and professionalization efforts from organizations playing a key role in seeking a unified government/industry strategy for better defining and improving these areas. Each organization has a mandate or charter related to either IT security education, awareness, and training, or professional development. Each represents a constituency that will benefit from progress in these areas. The results of a jointly sponsored government/industry workshop, scheduled for July 1993, to address these issues are also discussed.

**Panelist Ms. Dorothea E. de Zafra, MPIA
U.S. Public Health Service
Chair, Federal Information Systems
Security Educators' Association (FISSEA)**

**NEEDS FOR THE NINETIES:
GROWING PROFESSIONALIZATION OF
SECURITY TRAINING AND SECURITY TRAINERS**

FISSEA Overview

The Federal Information Systems Security Educators' Association (FISSEA) exists to provide for the exchange of information regarding - and for the improvement of - information systems security awareness, training, and education programs throughout the federal government, and by its contractors and academic institutions. Membership has grown exponentially, from 40 members in 1989 to over 600 members in 1993, with more than half of that growth occurring within the past year. By means of an annual conference, cooperative development/sharing of successful methodologies and training materials, a newsletter, and other means, FISSEA is a unique vehicle for the cross-fertilization of ideas and expertise among and between systems security professionals and professional trainers and educators. A membership survey this year has provided a profile of members' backgrounds, needs and interests as a guide both to future programming within the association, and to enhanced partnership with other organizations and agencies which are active in the systems security field.

Current Initiatives

The Computer Security Act of 1987 (P.L. 100-235) was quickly dubbed "the full-employment act for training contractors" upon enactment, because of the provision it contains which mandates training for all persons "involved in the management, use, or operation of federal computer systems that contain sensitive information." In government generally, and especially in civilian agencies, the training delivery structure of the 1980's was inadequate to reach such a vast and diverse audience in a cost-effective manner with appropriate levels of training materials and course content addressing the protection of sensitive-but-unclassified systems. Because this deficit has not been addressed through official agency training channels, FISSEA is undertaking the following initiatives:

- **Awareness Briefing Modules** Awareness-level briefings in computer

security basics, with flexibility for adaptation to differing agency environments, are being developed. Each module consists of overheads and accompanying text targeted to a specific audience category. The first modules, for executives, program managers, and functional managers, respectively are currently approaching completion.

- **Update and Revalidation of SPEC PUB 500-172** At the request of the National Institute of Standards and Technology (NIST), FISSEA is taking a fresh look at the governmentwide Computer Security Training Guidelines, issued in 1989, to improve their relevance in light of changing information technology trends and other factors.
- **Enhancement of the Computer Security Training and Awareness Course Compendium (NISTIR 4846)** FISSEA is serving as a resource to NIST in mapping the vendor course descriptions in the 1992 Compendium to NIST's Computer Security Training Guidelines (SPEC PUB 500-172). The results will be reflected in an updated edition of the Compendium, with the addition of more vendors as known.
- **Invitational Workshop on Information Technology Security Training and Professional Development** FISSEA was one of the five sponsoring organizations of this two-day workshop, held in July 1993, to bring government and industry together in addressing the improvement of security training generally and the professional development of security practitioners. FISSEA members Vic Maconochy, Ph.D. and Corey Schou, Ph.D. led the focus group concerned with computer security awareness, training, and education for users, operations staff, and management.
- **NIST-sponsored Information Technology Security Training Course** FISSEA is serving as a resource to NIST in the development, and perhaps delivery, of a three-day, government-focussed course on security issues and management responsibilities. It is intended for functional and program managers. The course is expected to be ready for pilot-testing next spring.

Implications for the Future

Implications of the above initiatives with respect to the state-of-the-art in information systems security training programs will be considered in the panel discussion.

Panelist Mr. Richard Koenig
Program Director
International Information Systems
Security Certification Consortium, Inc., or (ISC)²

Most public and private sector organizations have come to understand the need to cost-effectively protect information systems resources as part of an overall resource management strategy. Information systems security has also come to be seen as a people, rather than a technical problem. This has resulted in broad employee security awareness and training efforts as well as a proliferation of organization-specific professional development programs that are intended to "qualify" or "accredit" information system security staff through required training. The Computer Security Act, of course, spurred action in the federal arena; however, there was already considerable, less formal, private sector activity underway. In total contrast to this apparent widespread progress are the recent cutbacks in several industry segments that have virtually decimated the ranks of many security groups. These factors - as disparate as they may seem - point out that coordinating and focusing the awareness, training and professionalism efforts, as part of a unified government/industry strategy, would provide substantial benefits to the entire information systems security community.

(ISC)² is an independent, non-profit, and tax-exempt consortium which was formed in late 1989 to address professionalism of the information systems security practitioner, with the specific mission of developing a certification program for those in the career field. (ISC)² was conceived as a new, independent entity and as a cooperative effort by a number of organizations and special interest groups, because they felt that there was no clear and undisputed leader in the field who could successfully do it alone.

(ISC)²'s Certified Information Systems Security Professional (CISSP) program was started because it was felt that technical solutions are not the key to achieving the security and control needed in information systems. The key to meeting those needs depends, first and foremost, on the competence and integrity of the individuals who guide the selection, development and cost-effective application of those solutions. Thus, a personal standard of measure is needed both for expertise in the field *and* for ethical behavior related to the use of computerized information and information systems. To gain widespread acceptance in the field these standards of measure will best be developed through the voluntary standards process. (ISC)² is currently developing these professional standards via this process - by working with the National Institute of Standards and Technology and others to achieve a public consensus.

Integral to the CISSP program is the development of a Common Body of Knowledge (CBK). The CBK serves as the cornerstone by defining the knowledge-

base of the field, providing guidance for curriculum development and professional training, and establishing criteria for testing knowledge in the field. Development of the CBK is part of a concerted effort to solicit comments and input that will lead to a subsequent, more refined version. The new taxonomy will then be subjected to broad public scrutiny...to ensure that it, thus, represents a consensus of those working in the field. Such agreement on a CBK is a necessary element for definition of a distinct profession and career field.

Ultimately, these standards must be international in scope because the security and control needs for information systems transcend the traditional boundaries of a country's borders and a continent's coastline. Security and control issues have already had a significant impact on the developing global information society, and this impact will grow dramatically in the coming years. Thus far, our development efforts have been primarily focused in the U.S. and Canada...simply due to limited time and resources.

**Panelist Dr. William (Vic) Maconachy, Ph.D.
National Security Agency**

**AWARENESS, TRAINING, AND EDUCATION IN
INFORMATION TECHNOLOGY SECURITY:
A TIME FOR FOCUS AND CONSENSUS**

As with other professions, there does not exist a single source compendium of all the Knowledge, Skills, and Attributes (KSA's) required in the information technology security (ITS) field. In fact, the ITS career field covers a multitude of specialties. From the organizational-level Computer Security Officer to the Information System Security Architect, the scope and complexity of KSA's increases exponentially. Codification of those KSA's has taken many forms. The International Information Security Certification Consortium (ISC)² has researched and proposed a certification procedure. NIST Special Publication 500-172 lists topics for mastery by job category. The National Security Telecommunications and Information Systems Security Committee (NSTISSC) has issued a directive requiring federal departments and agencies to implement training programs for INFOSEC professionals. To some degree there are awareness and training materials related to these activities. The Federal Information Systems Security Educators' Association (FISSEA) is producing Awareness briefings (in conformance with NIST Special Pub. 500-172). Idaho State University has produced lesson plans covering several topics. The National Security Agency has developed a curriculum of instruction in the technical aspects of INFOSEC. To date, these activities have not been formally integrated and synchronized. This problem has been recognized. It is time for resolution. As partners in building a federal work force which is sensitive to and knowledgeable of information technology security, we need to collectively examine what we are now doing and where we need to be growing in awareness, training, and education.

Panelist Mr. Raymond Olszewski
Deputy Director, Professionalization Directorate
Center for Information Systems Security (CISS)

The newly-formed Center for Information Systems Security will be the vehicle by which the Department of Defense receives professional, state-of-the-art support to help secure vital information assets through products, policy, and personnel. The Professionalization Directorate has responsibility for all the personnel aspects of the DoD's information systems security program. This challenging mission will be tackled with new strategies coupled with existing DoD initiatives.

The Professionalization Directorate has a two-pronged approach to support INFOSEC professionals. The first is to create a career path which allows these professionals to realize advancement in this critical field; the second is to provide the education and training infrastructure necessary to give them the knowledge, skills, and abilities to succeed.

The creation of an INFOSEC career path directly supports the numerous information and security efforts being undertaken by a changing Defense Department. Security technology and its application is a key component which allows programs such as the Joint Staff's C4I for the Warrior and the Defense Information Infrastructure to become a reality. Without INFOSEC professionals to apply and operate the emerging technology, security implications would preclude the interoperability and standards necessary for these vital programs.

A career path alone, however, is not enough. These professionals need an education and training infrastructure to allow them to acquire the skills to become increasingly proficient and be able to progress in this interdisciplinary field. This aspect of our mission also insures current and future INFOSEC professionals can have ready access to education on the latest technology, products, and policies.

I eagerly look forward to the challenge of creating a professional INFOSEC cadre for the Department of Defense. These individuals will help insure warfighters and decision-makers have the right information they need when they need it.

**Panelist Ms. Joan M. Pohly
U.S. Air Force Cryptologic Support Center**

**NATIONAL-LEVEL INFOSEC EDUCATION, TRAINING,
AND AWARENESS INITIATIVES**

The evolution of the communications and computer disciplines into one information systems arena has necessitated the development of training standards for the government information systems community. The target audience is those INFOSEC professionals who have specialized in communications, computer, or emanations security and now must master the entire spectrum. To ensure an educated quality workforce, the National Subcommittee on Telecommunications and Information Systems Security chartered a working group to develop the first minimum training requirements for INFOSEC professionals. The effort will be published as national policy in 1993. This issuance is in consonance with the National Institute of Standards and Technology Guideline 500-172 as well as the reality and intent of the Computer Security Act of 1987. This effort is the work of members from both federal civilian and defense-related agencies.

"How Much Security Is Enough?"

The Accreditor's Perspective

James P. Litchko, Chair
Director, Business Development
Trusted Information Systems, Inc.

Gerald Hendricks, Jr.
NSA Accreditor
National Security Agency

John Martin Ferris
Assistant Director Systems Security
Department of the Treasury

Mr. George Wellham
Assistant Vice President
MNC Financial, Inc.

"How much security is enough?" This is a question that has been asked by every Department of Defense, Civil Agency, and commercial security manager who has ever approved an information system for operational use. How is this determination accomplished in the government and the commercial sector? In what ways does it differ, and what are the driving factors? What are the real criteria used to make that approval? Is it policy guidance, laws, common sense, risk analysis, monetary loss, or political pressure? Or is it all of these or none of these? Finally, what impact has the past decade of technology advancement had on the decision process, and what are the future issues that will impact future decisions.

This panel of information system security experts has a total of 45 years of experience with applying and approving security solutions to operational systems. These panelists also provide perspectives from the three communities that have the most extensive experience with integrating security solutions to support National and corporate security policies.

Jerry Hendricks has over eight years INFOSEC experience in the Department of Defense intelligence community supporting the integration of high-grade security solutions and classified systems deployed throughout the world. Recently, Mr. Hendricks completed a development process that will support accreditation of the installed and future deployments under the Defense Message System Program. Jerry is currently working on the accreditation of the Defense Information System's Network (DISN).

Martin Ferris, Assistant Director of Systems Security for the U.S. Department of Treasury, is

responsible for the development and implementation of policies and standards for the protection of sensitive and classified information processed by computer and telecommunications systems throughout Treasury and its bureaus. In this position, he represents Treasury on a variety of interagency security forums that develop the security policies and standards for the National Security community and that are developing security architectures for the federal law enforcement agencies. Mr. Ferris is also the chairperson of the information security standards subcommittee for the Accredited Standards Committee X9 for the financial services industry. Mr Ferris will discuss the issues in determining "how much is enough" in protecting Treasury automated information systems . The presentation will address the policies and processes required to determine security requirements and the Treasury program initiatives to make the question of "how much" easier to accredit. Emphasis will be placed on the importance of planning to achieve "enough" security.

George Wellham has worked with MNC Financial Corporation for the past eight years. He is responsible for the security evaluation of a financial system that includes over 2000 workstations and 30 networks connected among over 22 sites within Maryland. Mr. Wellham is very open to providing the costs and losses incurred from attacks on the systems for which he is responsible. He will provide the commercial methodology for ascertaining "how much is enough" by presenting the distributed accreditation process that is commonly used in the commercial sector.

In addition to providing perspectives on accreditation, the panel will be open to questions from the audience. This open exchange activity will provide the panelists an opportunity to respond to concerns and hear the perceptions presented by the audience.

Security and Auditability of Electronic Vote Tabulation Systems

**Panel Session
16th National Computer Security Conference**

Technological advances have gradually begun to impact the methods used for collection and tabulation of ballots cast by voters in elections. Vendors and representatives of federal agencies have indicated that computer hardware and software used for this purpose may be error-prone and vulnerable to attack. Considerable effort by the Federal Election Commission resulted in a set of guidelines for manufacture, procurement and evaluation of vote-tallying equipment, but these are not national standards, so many states and municipalities (frequently operating with inadequate computer expertise) have subsequently set their own (often widely-differing) criteria for system selection and certification. Constituents are increasingly discovering that choices regarding the auditability and security of their votes are being made at high levels and without referendum. This panel session will focus on public policy changes required for improvement in both the creation and deployment of computerized election systems.

Topics to be discussed will include:

- Applicability and adaptation of Orange Book classifications to voting systems.
- Utilization of NIST and NAVLAP in the certification of Independent Test Authorities for election equipment.
- A classification of potential voting system threats (as per the Neumann/Parker etymology).
- Tradeoffs in integrity and confidentiality matters.
- Vendor concerns regarding trade secrecy and feasibility issues.

Speakers:

Session Chair - Rebecca Mercuri, University of Pennsylvania

Panelists - Gary Greenhalgh, MicroVote Corp.
Peter Neumann, SRI International
Roy Saltman, NIST

Alternate - Lance Hoffman, George Washington University

An Integrity Model is Needed for Computerized Voting and Similar Systems

Roy G. Saltman

National Institute of Standards and Technology
Gaithersburg, MD 20899

Voting with the aid of computers began in the late 1960s. An important impetus was the rapid expansion at that time of the size of the electorate in such places as Los Angeles, and the consequent need to add large quantities of voting equipment that was relatively inexpensive. One initial answer was the pre-scored punch card. This concept involved the use of "standard" sized punched cards, in which the locations to be punched out to form holes were pre-scored, making them easy to form manually with the aid of a hard-pointed stylus. A vote for a particular candidate or issue alternative was indicated by a hole made at a particular location on the card. With the use of punch cards, large, heavy, and expensive lever voting machines located in each precinct were no longer required.

At that time, data on punch cards was a common method of input to computers; punch card readers accepting the "standard" size punch card and serving as peripherals to computers were in wide use. Vote-tallying programs were written as applications for the business computers to which the punch card readers were attached. The software, hopefully, correctly assigned the choices made that were punched into the cards, assuming that the card readers correctly sensed these choices.

Determination of election outcomes was accomplished with the use of a single computer located centrally. Voters would punch out their ballots at local precincts and drop their ballots into ballot boxes. After the close of polls (and perhaps once during election day), ballot boxes were collected and transported to the central location where the ballots were fed into the central computer for tallying. The County of Los Angeles, the nation's largest county in population, continues to carry out elections in exactly this manner. In a presidential election, Los Angeles County may process between 2.5 and 3 million punch card ballots.

Different techniques for both vote-casting and vote tallying have been introduced with advances in technology. For example, other types of computer-readable ballots have been employed. Among these are mark-sense ballots, in which a mark made on a ballot is automatically sensed by a computer-input unit, and punch cards in which the ballots are not pre-scored. In addition, with the coming of mini-computers, individual vote-tallying computers could be placed in each election precinct and only the summaries from them would need to be transmitted to a central computer to obtain jurisdiction-wide totals. Elections in the city of Chicago, for example, and in many other jurisdictions, now are carried out with one computer in each precinct or small group of precincts.

More recently, systems not using any ballots have been deployed. These, called direct-reading-electronic (DRE) systems, present the ballot on the face of a display unit of a computer located in the voter's precinct. The voter uses a touch-screen or a set of push-buttons to directly enter his or her choices into the computer. This system is the electronic equivalent of lever machines, and has been adopted in jurisdictions in which the use of hard-copy ballots is considered suspect. New York City, for example, has just agreed to purchase a large number of DRE systems for its elections; at least one DRE unit must be placed in each precinct.

In the future, there may be considerable use of non-ballot systems such as DREs. Voting by phone, a technique that is being tried in a few communities, employs no ballot. Voting from a remote terminal, from either a computer or an ATM-like device, similarly uses no ballot. A disadvantage of the lack of a hard-copy ballot is that the primary source document created by the voter to indicate choices is no longer present. Ballots are not available to be recounted, and therefore the computer logic, both hardware and software, must have assured, perhaps "trusted," correctness. There is no external redundancy for verification of reported results.

From the very start, there were concerns about the integrity of the process of computerized voting. The two primary issues have been, and continue to be, (1) the accuracy of computer-recording of the voter's choices, the "vote-casting" problem, and (2) the correctness of the computer software that generates contest results as a function of the votes entered into the computer, the "vote-tallying" problem.

Possibly the first significant airing of concern was in 1969 by two computer scientists in Los Angeles who stated publicly that a vote-tallying program could be rigged by the addition of an undetectable bias routine. A number of studies since then, including two by NIST [1, 2], have made extensive recommendations about audit trails, use of internal control techniques, design and testing of computer programs, physical security, operational security and management control of vote-tallying systems, system checkout prior to and following operation, accuracy and user-friendliness of vote-casting equipment and techniques, and other related aspects of voting that could affect the integrity of, or the confidence of the public in, the computed and reported outcomes. Indeed, the concerns continue with each election in which procedural errors are apparent. The most recent of these widely reported situations have occurred earlier this very year to fill a vacant seat in the First Congressional District of Wisconsin, and for the mayoralty of St. Petersburg, Florida.

At this time, there is no set of generally accepted procedures to assure system integrity because there are no mandatory security standards governing the operation of computerized vote-tallying, even in Federal elections. A recent paper by this author [3] has discussed the reasons for this and has made recommendations for the commencement of a program of work that would remedy the situation.

Additionally, the concept of "trusted systems" has not been applied to computerized voting. This concept was originally developed to respond to the need for assuring confidentiality of separate user sectors in multi-user systems. While confidentiality, as well as availability, are important in computerized voting, special tools of analysis are needed for integrity, the third parameter of the essential security requirements. As indicated above, accuracy of data entry and correctness of software are of primary concern. This is also true of certain other systems identified as "critical," such as any system in which human life or safety is at stake, e.g., air traffic control. Computerized voting systems were named as a potential target of attacks in the report Computers at Risk, published by the National Research Council [4].

A detailed integrity model that would specifically apply to computerized voting systems, and similar systems, would be highly useful. Such a model might be able to indicate to system designers and security analysts some method of logically analyzing the integrity of these systems and might provide some method of determination that a particular level of integrity had been achieved. An excellent start towards the development of such a model is the report Integrity in Automated Information Systems, published by the National Computer Security Center [5]. More work needs to be done.

References

- [1] Saltman, R. G., Effective Use of Computing Technology in Vote-Tallying, NBS Report NBSIR 75-687 (republished as SP 500-30), March, 1975; National Bureau of Standards, Gaithersburg, MD.
- [2] Saltman, R. G., Accuracy, Integrity, and Security in Computerized Vote-Tallying, NBS SP 500-158, August 1988; National Institute of Standards and Technology, Gaithersburg, MD.
- [3] Saltman, R. G., "Assuring Accuracy, Integrity, and Security in Computerized Vote-Tallying: The Role of the U.S. Congress;" Proceedings of the Third Annual Conference on Computers, Freedom, and Privacy, March 24-26, 1993, Burlingame, CA.
- [4] Computers at Risk: Safe Computing in the Information Age, System Security Study Committee, Computer Science and Telecommunications Board, Commission on Physical Sciences, Mathematics and Applications, National Research Council; 1990, Washington, D.C., pg. 8.
- [5] Integrity in Automated Information Systems, Technical Report 79-91, National Computer Security Center; September, 1991, Ford George G. Meade, MD.

Threats to Suffrage Security

Rebecca Mercuri

University of Pennsylvania

P.O. Box 1166

Philadelphia, PA 19105

215/736-8355

mercuri@gradient.cis.upenn.edu

Abstract

An existing risks-assessment classification is examined in the context of vote tabulation systems, in an effort to delineate the avenues for potential misuse and abuse. Some recommendations for security improvements are provided.

Introduction

*In the long history of the world
Only a few generations have been granted
The role of defending freedom
In the hour of maximum danger
I do not shrink from this responsibility
I welcome it*

-- John F. Kennedy

Suffrage, the right to vote, has long been viewed as an integral part of the democratic process. By this method, the people are able to register their opinions (through referendum questions) and can elect individuals who are viewed as capable of reflecting the views of the majority while serving the public at local, state and national levels. One would then surmise that the highest security methods would be required to be applied to any computer hardware and software used in elections, but this is not the case. Vote-tallying equipment is exempt from the Computer Security Act of 1987, despite the fact that it processes "sensitive information" whose "loss, misuse, or unauthorized access to or modification of which could adversely affect the national interest or the conduct of Federal programs..."[NCSC91, Salt93].

Steadily and silently, computer hardware and software has become entwined with virtually every phase of the election system. Voter registration databases are automated in almost all municipalities, ballots are directly cast into microprocessor-controlled units in ever-increasing numbers, punch-card and mark-sense vote tabulation is computerized, and end-of-day results are often electronically transmitted to news services. Current and pending legislation that will further enhance computer involvement with elections includes motor-voter registration, vote-by-phone, and faxing of votes by overseas military and civilian personnel.

Many open questions remain. What degree of accuracy should be demanded in voting systems? Are we willing to pay for increased security? Do auditability and confidentiality issues impose conflicting requirements? These tradeoffs and other matters need to be addressed and revisited as the technology evolves [Sha93]. Some insight, though, can be shed on the threats that computer systems impose on suffrage security by applying an etymology of computer misuse techniques to the election process [Neu89].

Vulnerabilities

Inherent in the nature of electronic vote tabulation (indeed, in all computers) are "gaps" that can be intentionally or accidentally used to subvert the systems. As identified by Neumann, these fall into three categories -- technological, sociotechnical and social:

The *technological* gap is that disparity between the expectations for the hardware and software, and what performance is capable of being delivered. In the case of voting systems, we might demand 100% accuracy from the ballot count, but we know that the mark-sense, punch-card and direct-entry methods all have margins of error. Even if inputs could be guaranteed to be correct, there is still the possibility of a random bit-flip within a memory device, which could affect the final result. This gap also applies to privacy of the balloting system (such as RF emissions from direct-entry machines), resistance to tampering, as well as auditability, configurability, and operability.

The *sociotechnical* gap involves the differences between social policies and computer policies. Social policies generally take the form of laws (computer crime, privacy...) and codes of ethics and commerce. Such laws have lagged behind the rapid advances in technology, and may not adequately address many new issues that could arise. Say, for example, a voting machine is configured so that votes for candidate X are registered to candidate Y. This is not at all fraudulent when performed in a laboratory demonstration, or even in a test procedure. It is fraudulent, though, if intentionally done in an actual election, but there may be no laws pertaining to such misuse. Furthermore, the determination of intent and fraud in the computer setting is difficult, if not (in some cases) impossible to differentiate from simple errors or omissions.

The *social* gap is that between social policies and human behavior. It involves the possibility of misuse during the election process. Should the manufacturer be required to foresee all potential problems and provide traps or flags to preclude them from happening, or should some of the responsibility for procedural correctness rest with the operators?

Neumann and Parker go on to define a set of classes of computer misuse techniques: external misuse, hardware misuse, masquerading, setting up subsequent misuse, bypassing intended controls, active misuse of resources, passive misuse of resources, misuse resulting from inaction and use as an aid to other misuses. Each of these categories applies directly to the electronic vote tabulation situation.

1. External misuse pertains to the observation or theft of information relating to the voting system. It might involve rummaging through discarded printouts, monitoring systems via their RF emission patterns, or visually obtaining a password (by watching the keystrokes of an operator). These actions are generally passive, but the information obtained may later be useful in a more overt system misuse or attack.

2. Hardware misuse can be either passive or active. Passive actions could include the placement of a data collection device within the voting system, or obtaining a discarded ballot-cartridge for the purposes of reverse-engineering. Active misuse includes theft of systems or components, intentional physical damage (dropping, slashing of ballot faces, cutting wires, insertion of glue in keys or switches, dousing with liquids, etc.), modifications (such as Trojan horse devices), power supply tampering, and interference (magnetic, electrical, etc.).

3. Masquerading is deliberate impersonation in order to obtain information or gain access to the system. Individuals might pose as service personnel or authorized operators before, during or after an election. In this way they can collect passwords, tamper with hardware and software, or directly manipulate the programming and tabulation processes. Vendors and election boards may also employ double-agents for other vendors, or persons with hidden agendas.

4. Subsequent misuses can be set up through the use of software Trojan horses that are time-triggered (so that they do not appear in pre- and post-election testing), or input-triggered (through the appearance of a particular data, command or even ballot sequence). Code for such misuse can be written to "self-destruct" following execution so that it does not appear in later system audits. Source code escrows can be rendered useless by involving the compilation or assembly process in performing the actual Trojan horse insertion [Thom84].

5. Controls established within the system for security and auditability may be bypassed both intentionally and accidentally. Exploitation of design flaws in multi-user systems, by using installed trapdoor programs, may enable unauthorized access to election software and data by individuals logged in through separate accounts. Password attacks can be used to obtain "superuser" status, from which audit trails can be turned off or modified to remove traces of system penetration.

6. Authority status may be misused actively within the system by legitimate superusers as well as those who are masquerading as such. Some of these misuses that apply to voting systems include: alteration of data, false data entry, and denials of service.

7. Passive misuse of resources can include browsing of data, global searching for patterns, and access to groups of files that can be used collectively in a more powerful way than when used separately. Within the voting context, the information gathered in this manner can generate statistics that could be used in subsequent attacks on the same system, or on others in remote locations. Direct access to vote totals, population stats, and registration information can be applied in order to shift tallies in swing precincts in subtle ways that would be hard to detect. System-specific information, such as ballot-cartridge programming or vote tabulation, can be transmitted to other municipalities that have similar installations, for use in subverting elections.

8. The lack of timely intervention in the event of a detected or potential problem can be viewed as a form of misuse. This can include inappropriate disposal or handling of election and computer media, non-reporting of an observed system attack, or other breaches of policy and procedure. Here a "cover-up" to save face in light of a system problem can be considered to be a further improper system use.

9. Uses of the system can be indirectly applied to other criminal acts or fraud. This form of misuse would enable individuals to be illegitimately elected who have the intention of performing illegal activities involving misuse of power, such as inappropriate bidding for contracts, misuse of funds, or nepotism in hiring.

It has been asserted (by industry and government representatives) that collusion would be necessary in order to tamper with an electronically tabulated election. The above list of points of attack indicates that this is untrue. System invasion can be done by a single individual, and as audit controls for access and use are typically lax or nonexistent, this can be done in a straightforward manner, often with minimal technical skills or knowledge. Such attacks may be motivated by politics, monetary rewards, power, foreign agencies, and subversion, to name but a few.

Recommendations and Concluding Statements

It is imperative that the electronic vote tabulating systems currently in use, or being proposed for use, be thoroughly examined with respect to the discussion above. A basic set of criteria for such examination has been prepared for this conference [Neu93], along with a suggested setting which should improve the evaluation process [Gre93]. Security assessment will be an ongoing concern -- as the field of electronic vote tabulation matures and evolves, open forums and review sessions will need to continue. Adequate security, integrity and assurance will not be achieved within the ad-hoc and free-market contexts that we have experienced up to now.

Presently, the resources of the National Computer Security Center and the National Institute for Standards and Technology have been largely ignored by the federal, state and local agencies who are responsible for overseeing the election process. The expertise of these organizations, and others in the private sector who have extensive experience in all phases of computer security, must be involved throughout the processes of regulation, procurement, and certification of all systems used in elections. Existing programs, such as the NCSC's Trusted Product Evaluation system, should be applied to election equipment [NCSC85, NCSC90, NCSC92]. It is incumbent upon the federal government (through Congress and the Federal Election Commission) to take a leadership role in establishing and mandating minimal compliance standards, not just suggested guidelines. Government officials, at all levels, must work actively with vendors and municipalities to guarantee that elections are carried out with the highest integrity that technology will allow.

Each of us is charged with the responsibility of guarding the election process -- our checks and balances must not be turned over to machinery. Surely, the role of defending freedom must remain in the hands of those who most cherish its value.

Acknowledgments

The author would like to thank Mae Churchill, Peter Neumann and Roy Saltman for their advice and encouragement in the preparation of this document as well as with the related panel session.

References

- [Gre93] Greenhalgh, Gary L., *Security and Auditability of Electronic Vote Tabulating Systems: One Vendor's Perspective*, 16th National Computer Security Conference, Baltimore, MD, September 1993.
- [NCSC85] National Computer Security Center, *Department of Defense Trusted System Evaluation Criteria (TCSEC)*, DOD-5200.28-STD (Orange Book), December 1985.
- [NCSC90] National Computer Security Center, *Trusted Product Evaluations, A Guide for Vendors*, NCSC-TG-002, June 1990.
- [NCSC91] National Computer Security Center, *Integrity-Oriented Control Objectives: Proposed Revisions to TCSEC*, C Technical Report 111-91, Library No. S238,183, October 1991.
- [NCSC92] National Computer Security Center, *Trusted Product Evaluation Questionnaire*, NCSC-TG-019, May 1992.
- [Neu89] Neumann, Peter G., and Parker, Donn B., *A Summary of Computer Misuse Techniques*, 12th National Computer Security Conference, Baltimore, MD, October 1989.
- [Neu93] Neumann, Peter G., *Security Criteria for Electronic Voting*, 16th National Computer Security Conference, Baltimore, MD, September 1993.
- [Salt93] Saltman, Roy G., *Assuring Accuracy, Integrity and Security in National Elections: The Role of the U.S. Congress*, Position Papers from the Third Conference on Computers, Freedom and Privacy, San Francisco, CA, March 1993.
- [Sha93] Shamos, Michael, *Electronic Voting -- Evaluating the Threat*, Position Papers from the Third Conference on Computers, Freedom and Privacy, San Francisco, CA, March 1993.
- [Thom84] Thompson, Ken, *Reflections on Trusting Trust*, Communications of the ACM, Vol. 27, No. 8, August 1984.

SECURITY CRITERIA FOR ELECTRONIC VOTING

Peter G. Neumann

Computer Science Laboratory

SRI International, Menlo Park CA 94025

1-415-859-2375 Neumann@csl.sri.com

Abstract. Some basic criteria for confidentiality, integrity, availability, reliability, and assurance are considered for computer systems involved in electronic voting. An assessment of the realizability of those criteria leads to the conclusion that, operationally, many of the criteria are inherently unsatisfiable with any meaningful assurance.

BACKGROUND

The election processes of voter registration, vote casting, vote counting, and ballot generation are becoming increasingly automated [Sal93]. Numerous cases of allegedly accidental errors have been reported, along with suspicions of fraud [Dug88, Neu90]. However, the borderline between accident and fraud is murky. Serious security vulnerabilities are commonplace in most voting systems, providing widespread opportunities for computer-system misuse — particularly by insiders [NeuPar89, Mer93]. Indeed, incentives for bribery, collusion, and fraud are likely to be enhanced by the financial stakes involved in winning or losing an election.

At present there is no generally accepted standard set of criteria that voting systems are required to satisfy. This paper proposes a generic set of criteria similar in concept to existing security criteria such as the U.S. TCSEC (the Orange Book, TNI, TDI, etc.), the European ITSEC, the Canadian CTCPEC, and the draft U.S. Federal Criteria. We observe that essentially all existing voting systems would fail to satisfy even the simplest of the existing criteria. Worse yet, each of these criteria is itself incomplete in that it fails to encompass many of the possible risks that must ultimately be addressed. Unfortunately, previous attempts to define criteria specifically for voting systems [Sal88, Sha93, FEC, NYC87] are also incomplete. However, the risks lie in the inherent unrealizability of the criteria as well as in the incompleteness of those criteria.

Copyright 1993, Peter G. Neumann. This paper was presented at the 16th National Computer Security Conference Baltimore, Maryland, September 20-23, 1993.

ELECTRONIC VOTING CRITERIA

Generic voting criteria are suggested here as follows:

- **System integrity.** The computer systems (in hardware and system software) must be tamper-proof. Ideally, system changes must be prohibited throughout the active stages of the election process. That is, once certified, the code, initial parameters, and configuration information must remain static. No run-time self-modifying software can be permitted. End-to-end configuration control is essential. System bootload must be protected from subversion that could otherwise be used to implant Trojan horses. (Any ability to install a Trojan horse in the system must be considered as a potential for subverting an election.) Above all, vote counting must produce reproducibly correct results.
- **Data integrity and reliability.** All data involved in entering and tabulating votes must be tamperproof. Votes must be recorded correctly.
- **Voter anonymity and data confidentiality.** The voting counts must be protected from external reading during the voting process. The association between recorded votes and the identity of the voter must be completely unknown within the voting systems.
- **Operator authentication.** All people authorized to administer an election must gain access with nontrivial authentication mechanisms. Fixed passwords are generally not adequate. There must be no trapdoors — for example, for maintenance and setup — that could be used for operational subversions.
- **System accountability.** All internal operations must be monitored, without violating voter confidentiality. Monitoring must include votes recorded and votes tabulated, and all system programming and administrative operations such as pre- and post-election testing. All attempted and

successful changes to configuration status (especially those in violation of the static system integrity requirement) must be noted. This capability is similar to that of an aircraft flight recorder, from which it is possible to recover all important information. Furthermore, monitoring must be nonbypassable — it must be impossible to turn off or circumvent. Monitoring and analysis of audit trails must themselves be nontamperable. All operator authentication operations must be logged. ([Gre93] analyzes accountability further.)

- **System disclosability.** The system software, hardware, microcode, and any custom circuitry must be open for random inspection at any time (including documentation), despite cries for secrecy from the system vendors.
- **System availability.** The system must be protected against both accidental and malicious denials of service, and must be available for use whenever it is expected to be operational.
- **System reliability.** System development (design, implementation, maintenance, etc.) should attempt to minimize the likelihood of accidental system bugs and malicious code.
- **Interface usability.** Systems must be amenable to easy use by local election officials, and must not necessitate the on-line control of external personnel (such as vendor-supplied operators). The interface to the system should be inherently fail-safe, fool-proof, and overly cautious in defending against accidental and intentional misuse.
- **Documentation and assurance.** The design, implementation, development practice, operational procedures, and testing procedures must all be unambiguously and consistently documented. Documentation must also describe what assurance measures have been applied to each of those system aspects.

Other lower-level criteria from the TCSEC are also applicable, such as trusted paths to the system, trusted facility management, trusted recovery, and trusted system distribution. All of the above criteria elements require technological measures and some administrative controls for fulfillment. The following item requires primarily nontechnological factors.

- **Personnel integrity.** People involved in developing, operating, and administering electronic voting systems must be of unquestioned integrity. For example, convicted felons and gambling entrepreneurs are suspect.

The above set of skeletal criteria is by no means complete. There are many other important attributes that election computing systems need to satisfy operationally. For example, Saltman [Sal88] notes that voting systems must conform with whatever election laws may be applicable, the systems must not be shared with other applications running concurrently, ballot images must be retained in case of challenges, pre- and post-election testing must take place, warning messages must occur during elections whenever appropriate, would-be voters must be properly authorized, handicapped voters must have equal access, it must be possible to conduct recounts manually, and adequate training procedures must exist.

REALIZABILITY

No criteria can completely encompass all of the possible risks. However, even if we ignore the incompleteness and imprecision of the suggested criteria, numerous intrinsic difficulties make such criteria unrealizable with any meaningful assurance.

System trustworthiness

- *Security vulnerabilities* are ubiquitous in existing computer systems, and also inevitable in all voting systems — including both dedicated and operating-system-based applications. Vulnerabilities are particularly likely in voting systems developed inexpensively enough to find widespread use. Evidently, no small kernel can be identified that mediates security concerns, and thus potentially the entire system must be trustworthy.
- *System operation* is a serious source of vulnerabilities, with respect to integrity, availability, and in some cases confidentiality — even if a system as delivered appears to be in an untampered form. A system can have its integrity compromised through malicious system operations — for example, by the insertion of Trojan horses or trapdoors. The presence of a *superuser* mechanism presents many opportunities for subversion. Furthermore, Trojan horses and trapdoors are not necessarily static; they may appear only for brief instants of time, and remain totally invisible at other times. In addition, systems based on personal computers are subject to spoofing of the system bootload, which can result in the seemingly legitimate installation of totally bogus software. Even in the presence of cryptographic checksums, a gifted developer or subverter can install a flaw in the system implementation or in the system generation. Ken Thompson's Turing-Lecture stealthy

Trojan horse technique [Tho84] illustrates that no modifications to source code are required.

- *System integrity* can be enhanced by the use of locally nonmodifiable read-only and once-writable memories, particularly for system programs and preset configuration data, respectively.
- *Data confidentiality, integrity, and reliability* can be subverted as a result of compromises of system integrity. Nonalterable (e.g., once-writable) media may provide some assistance for integrity, but not if the system itself is subvertible.
- *Voter anonymity* can be achieved by masking the identity of each voter so that no reverse association can be made. However, such an approach makes accountability much more difficult. One-way hashing functions or even public-key encryption may be useful for providing later verification that a particular vote was actually recorded as cast, but no completely satisfactory scheme exists for guaranteeing voter anonymity, consistency of the votes tabulated with respect to those cast, and correct results. Any attempt to maintain a bidirectional on-line association between voter and votes cast is suspect because of the inability to protect such information in this environment.
- *Operator authentication* must no longer rely on sharable fixed passwords, which are too easily compromised in a wide variety of ways. Some other type of authentication scheme is necessary, such as a biometric or token approach, although even those schemes themselves have recognized vulnerabilities.
- *System accountability* can be subverted by embedded system code that operates below the accounting layers, or by low-layer trapdoors. Techniques for permitting accountability despite voter anonymity must be developed, although they must be considered inherently suspect. Read-only media can help ensure nontamperability of the audit trail, but nonbypassability requires a trusted system for data collection. Accountability can be subverted by tampering with the underlying system, below the layer at which auditing takes place. (See also [Gre93].)
- *System disclosability* is important because proprietary voting systems are inherently suspect. However, system inspection is by itself inadequate to prevent stealthy Trojan horses, run-time system alterations, self-modifying code, data interpreted as code, other code or data subversions, and intentional or accidental discrepancies between documentation and code.

System Robustness

- *System availability* can be enhanced by various techniques for increasing hardware-fault tolerance and system security. However, none of these techniques is guaranteed.
 - *System reliability* is aided by properly used modern software-engineering techniques, which can result in fewer bugs and greater assurance. Analysis techniques such as thorough testing and high-assurance methods can contribute. Nevertheless, some bugs are likely to remain.
 - *Use of redundancy* can in principle improve both reliability and security. It is tempting to believe that checks and balances can help satisfy some of the above criteria. However, we rapidly discover that the redundancy management itself introduces further complexity and further potential vulnerabilities. For example, triple-modular redundancy could be contemplated, providing three different systems and accepting the results if two out of three agree. However, a single program flaw (such as a Trojan horse) can compromise all three systems. Similarly, if three separately programmed systems are used, it is still possible for common-fault-mode mistakes to be made (there is substantial evidence for the likelihood of that occurring) or for collusion to compromise two of the three versions. Furthermore, the systems may agree with one another in the presence of bogus data that spoofs all of them. Thus, both reliability and security techniques must provide end-to-end protection, and must check on each other.
- In general, Byzantine algorithms can be constructed that work adequately even in the presence of arbitrary component failures (for example, due to malice, accidental misuse, or hardware failure). However, such algorithms are expensive to design, implement, and administer, and introduce substantial new complexities. Even in the presence of algorithms that are tolerant of n failed components, collusion among $n+1$ can subvert the system. However, those algorithms may be implemented using systems that have single points of vulnerability, which could permit compromises of the Byzantine algorithm to occur without n failures having occurred; indeed, *one* may be enough. Thus, complex systems designed to tolerate certain arbitrary threats may still be subvertible by exploiting other vulnerabilities.
- *Interface usability* is a secondary consideration in many fielded systems. Complicated operator

interfaces are inherently risky, because they induce accidents and can mask hidden functionality. However, systems that are particularly user-friendly may be even more amenable to subversion than those that are not.

- *Correctness* is a mythical beast. In reliable systems, a probability of failure of 10^{-4} or 10^{-9} per hour may be required. However, such measures are too weak for voting systems. For example, a one-bit error in memory might result in the loss or gain of 2^k votes (for example, 1024 or 65,536). Ideally, numerical errors attributable to hardware and software must not be tolerated, although a few errors in reading cards may be acceptable within narrow ranges. Efforts must be made to detect errors attributable to the hardware through fault-tolerance techniques or software consistency checks. Any detected but uncorrectable errors must be monitored, forcing a controlled rerun. However, a policy that permits any detected inconsistencies to invalidate election results would be very dangerous, because it might encourage denial-of-service attacks by the expected losers. Note also that any software-implemented fault-tolerance technique is itself a possible source of subversion.

System Assurance

- *High-assurance systems* demand discipline and professional maturity not previously found in commercial voting systems (and, indeed, not found in most commercial operating systems and application software). High-assurance systems typically cost considerably more than conventional systems in the short term, but have the potential for payoff in the long term. Unless the development team is exceedingly gifted, high-assurance efforts may be disappointing. As a consequence, there are almost no incentives for any assurance greater than the minimal assurance provided by lowest-common-denominator systems. (See [Neu93] for a discussion of some of the implications of attaining high assurance.) Furthermore, even high-assurance systems can be compromised, via insertion of trapdoors and Trojan horses, and operational misuse.

CONCLUSIONS

The primary conclusion from the above discussion of realizability is that certain criteria elements are inherently unsatisfiable with assurance that can be attained at an acceptable cost. Systems could be de-

signed that will be operationally less amenable to subversion. However, some of those will still have modes of compromise without any collusion. Indeed, the actions of a single person may be sufficient to subvert the process, particularly if preinstalled Trojan horses or operational subversion can be used. Thus, whereas it is possible to build better systems, it is possible that those better systems can also be subverted. Consequently, there will always be questions about the use of computer systems in elections. In certain cases, sufficient collusion will be plausible, even if one is not a confirmed conspiracy theorist.

There is a serious danger that the mere existence of generally accepted criteria coupled with claims that a system adheres to those criteria might give the naïve observer the illusion that an election is nonsubvertible. Doubts will always remain that some of the criteria have not been satisfied with any realistic measure of assurance and that the criteria are incomplete:

- Commercial systems tend to have lowest common denominators, with numerous serious security flaws. Custom-designed systems may be even worse, especially if their code is proprietary.
- Trojan horses, trapdoors, interpreted data, and other subversions can be hidden, even in systems that have received extensive scrutiny. The integrity of the entire computer-aided election process may be compromisable internally.
- Operational misuses can subvert system security even in the presence of high-assurance checks and balances, highly observant poll watching, and honest system programmers. Registration of bogus voters, insertion of fraudulent absentee ballots, and tampering with punched cards seem to be ever-popular techniques in low-tech systems. In electronic voting systems, dirty tricks may be indistinguishable from accidental errors. The integrity of the entire computer-aided election process may be compromisable externally.
- The requirement for voter confidentiality and the requirement for nonsubvertible and sufficiently complete end-to-end monitoring are conceptually contradictory. It is essentially impossible to achieve both at the same time without resorting to complicated mechanisms, which themselves may introduce new potential vulnerabilities and opportunities for more sophisticated subversions. Monitoring is always potentially subvertible through low-layer Trojan horses. Furthermore, any technique that permitted identification and authentication of a voter if an election were challenged would undoubtedly lead to increased challenges and further losses of voter privacy.

- The absence of a physical record of each vote is a serious vulnerability in direct-recording election (DRE) systems; the presence of an easily tamperable physical record in paper-ballot and card-based systems is also a serious vulnerability.
- Problems exist with both centralized control and distributed control. Highly distributed systems have more components that may be subverted, and are more prone to accidental errors; they require much greater care in design. Highly centralized approaches in any one of the stages of the election process violate the principle of separation of duties, and may provide single points of vulnerability that can undermine separation enforced elsewhere in the implementation.

There is a fundamental dilemma to be addressed.

- On one hand, computer systems can be designed and implemented with extensive checks and balances intended to make accidental mishaps and fraud less likely. As an example pursuing that principle, New York City [NYC87] is attempting to separate the processes of voting, vote collection, and vote tallying from one another, with redundant checks on each, hoping to ensure that extensive collusion would be required to subvert an election, and that the risks of detection would be high; however, that effort permits centralized vote tallying, which has the potential for compromising the integrity of the earlier stages.
- On the other hand, constraints on system development efforts and expectations of honesty and altruism on the part of system developers seem to be generally unrealistic, while the expectations on the operational practice and human awareness required to administer such systems may be unrealistic.

We must avoid lowest-common-denominator systems, instead trying to approach the difficult goal of realistic, cost-effective, reasonable-assurance, fail-safe, and nontamperable election systems.

Vendor-embedded Trojan horses and accidental vulnerabilities will remain as potential problems, for both distributed and centralized systems. The principle of separation is useful, but must be used consistently and wisely. The use of good software engineering practice and extensive regulation of system development and operation are essential. In the best of worlds, even if voting systems were produced with high assurance by persons of the highest integrity, the operational practice could still be compromisable, with or without collusion. Vigilance throughout the election process is

simply not enough to counter accidental and malicious efforts that subvert the process. Some residual risks are inevitable.

ACKNOWLEDGMENT

The author is grateful to Rebecca Mercuri for her incisive feedback during the preparation of this position paper, and to Mae Churchill for continual inspiration.

REFERENCES

- [Dug88] R. Dugger. Annals of Democracy (Voting by Computer). New Yorker. November 7, 1988.
- [FEC] Federal Election Commission guidelines.
- [Gre93] G.L. Greenhalgh. Security and Auditability of Electronic Vote Tabulation Systems: One Vendor's Perspective. *Proc. 16th National Computer Security Conference*, NIST/NCSC, Baltimore MD, September 1993.
- [Mer93] R. Mercuri. Threats to Suffrage Security. *Proc. 16th National Computer Security Conference*, NIST/NCSC, Baltimore MD, September 1993.
- [NeuPar89] P.G. Neumann and D.B. Parker. A Summary of Computer Misuse Techniques. *Proc. 12th National Computer Security Conference*, NIST/NCSC, Baltimore MD, pp. 396-407, October 1989.
- [Neu90] P.G. Neumann. Risks in Computerized Elections (Inside Risks). *Comm. ACM* 33, 11, p. 170, November 1990.
- [Neu93] P.G. Neumann. Myths of Dependable Computing: Shooting the Straw Herrings in Midstream. *Proc. 8th Annual Conf. on Computer Assurance (COMPASS '93)*, June 1993.
- [NYC87] Electronic Voting System. Request for Proposal, Appendix G, Security and Control Considerations. New York City Board of Elections, New York City Elections Project, September 1987.
- [Sal88] R.G. Saltman. Accuracy, Integrity, and Security in Computerized Vote-Tallying. NBS (now NIST) special publication, 1988.
- [Sal93] R.G. Saltman. Assuring Accuracy, Integrity and Security in National Elections: The Role of the U.S. Congress. Position paper from *Computers, Freedom and Privacy '93*, pp. 3.8-3.17, March 1993.
- [Sha93] M. Shamos. Electronic Voting — Evaluating the Threat. Position paper from *Computers, Freedom and Privacy '93*, pp. 3.18-3.25, March 1993.
- [Tho84] K. Thompson. Reflections on Trusting Trust. *Comm. ACM*, 27, 8, pp. 761-763, August 1984.

Security and Auditability of Electronic Vote Tabulation Systems: One Vendor's Perspective

Dr. Gary L. Greenhalgh
Director, National Sales
The Microvote Corporation
6524 North Ferguson
Indianapolis, IN 46220

BACKGROUND

Security and auditability of electronic vote tabulation systems is low on the list of vendor priorities. In order to understand why, it is first essential to understand the nature of the marketplace for vote tabulation systems. After a brief look at the marketplace, I will then turn to an analysis and discussion of the Federal Election Commission's (FEC) voluntary voting system standards. Finally, I will provide a suggested solution to problems, shortcomings and concerns presented in these first two sections of the paper.

THE VOTING EQUIPMENT MARKETPLACE

1. VENDORS RESPOND TO MARKETPLACE DEMANDS. To be blunt, most state and local election agencies do not seem to care about security and auditability of electronic vote tabulation systems that they purchase. Indeed, during the past two years, I have written most of my company's responses to Requests For Proposals issued by close to 30 state and local election agencies. Not one RFP mentioned one thing about voting machine or software security or auditability. If this panel concludes that *Security and Auditability of Electronic Vote Tabulation Systems* is a major problem that must be addressed by the election community, then someone must convey this information to the state and local procurement officers.

2. HOW VOTING EQUIPMENT IS PURCHASED. Let's play a guessing game. Which of the following vendors will be successful?

A. Vendor A invests substantial resources in producing a quality electronic voting system with a high level of security and auditability. However, with limited resources, Vendor A cannot contribute to state and local political campaigns or pay attendance fees or host "hospitality suites" at national and state election officials' conventions.

B. Vendor B has an inferior product, but contributes heavily to political campaigns and is a "hale fellow, well met" at election officials' conferences.

It is likely that A will be out of business in 6 months, and B will not only stay in business, but will thrive.

Submitted for: Panel session on "Security and Auditability of Electronic Vote Tabulation Systems," 1993 National Computer Security Conference, Baltimore, MD, September 20-23, 1993.

3. INNOVATION IS DISCOURAGED, NOT ENCOURAGED. Prior to selling vote-counting hardware or software, most states require some sort of formal certification. In these states, certification is often an expensive and time-consuming process (to say nothing of having to provide the same information in 50 different formats). Some states even change the rules or invent new ones during certification. As a result, the average time is one year from initial submission to certification. In addition, if hardware or software is altered or improved by a vendor after the initial certification, a number of states require complete recertification. As a result, vendors will not improve or modify hardware or software unless absolutely necessary. So instead of being praised for innovation, vendors are penalized.

4. YOU THINK WE'RE RICH, BUT WE'RE NOT. Total sales of new vote counting hardware and software probably does not exceed \$10,000,000 in an average year. These sales are divided among some eight vendors. In addition, with budget cutbacks at the state and local level, new sales have declined even further. Thus, it is unrealistic to expect vendors to invest substantial resources into insuring that their hardware and software have a high level of security and auditability, especially when this is not a priority for state and local governments.

Look at the election industry from an *investor's* standpoint. First of all, someone has to put up initial investment capital to develop, test and manufacture equipment and software. In Microvote's case, close to \$1,000,000 was spent between 1982 and 1986 to accomplish these initial tasks. Then, once the system is in production, the equipment and software has to be certified at the state level. Finally, after state certification (and in order to sell at the local level), sales people have to be hired along with a service and support staff to insure that the system functions properly once installed. In sum, it will take a minimum of four years to go from conceptual design to the first sale. Are there any investors out there willing to sit on their money for four years?

5. IT IS ALMOST IMPOSSIBLE FOR A SMALL COMPANY TO SURVIVE TODAY. Several market realities make it virtually impossible for a new corporation to succeed in the election field. First, is the length of time involved in getting a product to market. Second, is the fact that the election market is extremely conservative. I have had election officials tell me that they'd rather buy an inferior product from a known company rather than take a chance with a new company and an admittedly better product. Third, is the simple fact that the market is now dominated by two major corporations who have the sales and other resources to dominate the field.

6. LIKE IT OR NOT, WE ARE GOING TO PROTECT OUR SECRETS. While we acknowledge that there is a compelling public interest in the integrity of vendor hardware and software used in public elections, vendors are extremely wary of revealing confidential hardware and software codes and data. Why? For one thing, certain public officials, after having obtained access to this information, decided to get into the election business. And make no mistake about it, patents and copyrights do not provide adequate protection.

Here is a case in point. As part of a particular state's certification process, a local university was anointed to test voting equipment and software. After testing several vendors' hardware and software, the head of this testing laboratory decided to get into the election business and is now marketing a Direct Recording Electronic (DRE) voting system. What a deal! You get paid by a state to test equipment and you get the vendors to pay for your "education" about voting equipment and software!

7. LACK OF VENDOR OVERSIGHT AND VENDOR DEPENDENCE. Owing to staff and other cutbacks at the state and local level, oversight of vendor hardware and software has

been dramatically reduced. This has been accompanied by an increase in vendor dependence where the vendors are expected to not only service and support the voting equipment, but are expected to perform critical election functions *including election night vote tallying!*

Here's an example of this. Local officials in one state require the vendors, *by contract*, to program and prepare all the voting equipment used in public elections in that state. These local officials do not even bother to pretest the equipment to make sure the programming is done correctly!

8. LACK OF MARKET CONTROLS. It would be expected that a major vendor screw-up, *of which there have been many*, would result in that vendor at least losing some business. But this hasn't happened. Why? Problems are not only covered up, but more than a few election officials have outright lied about such vendor problems. Why the "user" cover-up? Simple. Most election officials are not going to admit that they made a mistake in choosing a voting system. Hence, problems proliferate while, at the same time, certain vendors are allowed to continue to market inferior hardware and software.

9. COST IS THE DRIVER. Simply put, vendors have to make money to survive. And it costs considerable money to develop, test, market and support election systems with the high degree of security and auditability required by the various private and governmental organizations involved in elections. On the other side of the equation are the purchasers. To them, low price is paramount. So here's the \$64 question: how to develop, test, market and support electronic voting systems with a high degree of security that will be purchased by state and local governments at a large enough profit to insure the vendor will stay in business?

VOTING SYSTEM STANDARDS

The FEC's voluntary voting system standards (or, to be exact, the legislative authority to enact these standards) were not the result of any public outcry for such standards. Rather, the standards were initiated on behalf of both myself and a group of election officials as a way to save the Clearinghouse on Election Administration (which I had been directing since it was formed at the General Accounting Office in 1973 and transferred to the FEC in 1976).

The idea for the standards evolved during a number of discussions between a member of the Nevada State Senate, Daniel Demers, Roy Saltman, Senator Cannon, several election officials and myself. As Chairman of the Election Committee of the State Senate in Nevada and a member of the Clearinghouse Advisory Panel, Mr. Demers had become quite upset with the stories about punch card voting system failures throughout the United States. Senator Demers was able to convince Senator Cannon of the need for such standards. In addition, Roy Saltman's initial monograph on *Accuracy, Integrity, and Security in Computerized Vote Tallying* had a great influence on the Senate Rules Committee when it was first published in 1976.

To my recollection, the objectives of the voluntary standards were twofold. First, the standards were to establish baseline criteria that every voting system should meet. I always thought it essential, as an example, that every voting system used in America should record and document all votes quickly, completely and accurately. As such, these standards were never intended to replace testing criteria used by the individual states. Rather, the standards were intended to supplement testing provisions enacted by the states.

Looked at from another viewpoint, the standards should not require or mandate features of any particular type of equipment unless such features were directly related to the integrity of the voting process. Similarly, all types of voting systems, *including mechanical lever voting systems now being used by 30% of America's voters*, were to have standards set and no particular type of voting system was to be "favored" by the standards. Finally, all parts of every voting system used in America were to be "standardized" including paper ballots used in punch card and optical scan equipment and, of course, all support software associated with each type of system.

To digress, readers should remember the state-of-the-art regarding electronics, computers and software when the standards were begun in 1980. There were no DRE voting systems or PCs at that time, only mini- and mainframe computers. So self-sufficiency using dedicated computers was, for most election jurisdictions, unrealistic. In addition, election software was nowhere near as sophisticated as software used today. For example, almost everyone's software at that time was written around a particular piece of vote counting hardware. Database management election systems, electronic integration of absentee and "at the polls" election results, and other applications were only on the drawing boards.

The second objective of the standards was more subtle. In sum, we wanted these standards to be an incentive for new companies to get into the election business. Why? At that time, the market was dominated by one company. Many of us felt that the products of this election company were weak, were supported by old software, and were "hard sold" in such a way as to discourage competition. We looked at the standards as, in effect, providing targets for new companies to compete with this one vendor.

Have these objectives been achieved with the standards that were enacted in 1990? For the first objective, the record is mixed. For the second objective, the answer is clearly no. Here are my reasons:

1. In one of the earlier draft versions of the standards, optical scan voting equipment appeared to be favored over DRE voting equipment. But precinct count optical scan equipment, according to the earlier version of the standards, did not have to have a public and protective counter (an ongoing election-day record of how many voters had voted on the equipment since it was started on election morning and a record of the total number of votes cast on the equipment since it left the manufacturer) while DRE equipment was specifically required to have such counters. Fortunately, reason prevailed (as well as some rather pointed letters from DRE users to their Congressional delegation) and the standards for optical scan equipment were upgraded to require such counters.

As a result, as a DRE voting equipment company, we, quite frankly, have some continuing concerns about the neutrality of the standards.

2. In certain areas, the standards do not go far enough. For example, the software standards must be strengthened to mandate high level languages for all software used to tally and record votes in elections. We discovered, fairly recently, that one of the largest vendors in America uses assembly language for its source code! In addition, many vendors have now moved to DBMS and market their systems as complete "self-sufficient" packages. The software standards should directly address election DBMS packages sold to election agencies.

3. The standards should not merely suggest certain key features, but should make them mandatory *if such features are directly related to the integrity of the election*

process. For example, a real time and event clock, that logs every activity that occurs on election day, must be made mandatory in every voting system. In addition, every voting system must have a complete set of self-diagnostics that describe any problem, in English, with complete documentation for correction of the problem.

4. Full system documentation must be made mandatory. Such documentation must include a complete hardware manual covering all pre-election set-up, election day operations, post-election procedures, maintenance, repairs and training. Software documentation must include both a training manual and "internal" documentation. This should be supplemented by poll worker training and related voter information manuals and instructions.

5. The standards must also address minimal requirements that should be met by the "users" of voting equipment. Included here should be minimal staffing levels, staff expertise, documentation, security requirements, training and related technical and engineering requirements (storage and pre-election set up facilities, electrical requirements and election day operating requirements).

6. Another critical software standard relates to the ownership of the software sold to public agencies. The standards must clearly address the issue of copyrights to the software and such standards must require that the company selling the software own all related copyrights outright *and have firm, written ownership agreements with the authors of such software*. Why is this important? If the software authors leave a voting equipment company, that company must be secure in its rights to continue to market, service and support such software.

7. The standards should require all vendors to *warrant and support* all versions of vote counting hardware and software sold to any public agency, for a period of at least five years. The standards should also spell out the terms of the warranty and what that warranty must cover. One major vendor has now discontinued service to certain local election jurisdictions for equipment sold only a year ago and yet another vendor has a standard 90-day warranty!

8. The standards must also address all associated proprietary parts, support equipment and election day "consumables." Each vendor must be required to provide these items for a period of at least ten years, at a reasonable price. Similarly, if certain parts, support equipment or consumables are required for warranty purposes, vendors must be required to clearly document this information prior to any sale.

9. The standards must be updated, frequently, to reflect changes in the industry. For example, several election jurisdictions are now using multiple sites to tally election results. Standards should be set for such remote communication of election results. Also, several vendors are now using off-the-shelf ballot printing and other devices as part of the system sold to election jurisdictions. Standards must also be set for this equipment.

Finally, we all must recognize both the importance of these standards and the need for strict neutrality and complete integrity and professionalism on the part of the FEC and all organizations anointed with a role to play in the implementation of these standards. Unfortunately, we have seen signs that such strict neutrality, high integrity and complete professionalism have not been observed. As a result, we at Microvote are very reluctant to have our equipment tested and certified by any testing laboratory until we are assured that the equipment will be tested fairly, and that the laboratory, after testing our equipment, won't suddenly decide to get into the election business and compete with us.

THE SOLUTION

Now that we have defined the problems with the marketplace, with the standards, and with their implementation, let us look at a solution.

In my view, the solution involves the creation and staffing of a high-level Institute that incorporates the interests of three groups: the "users" (state and local election officials); the public interest community (the federal government, universities and other organizations) and those private vendors who manufacture, sell, service and support electronic vote tabulation systems. A federal agency (with the possible exception of NIST) is not the proper vehicle to provide this forum. Why? Certain state and local officials simply do not trust the federal government. And, to be frank, certain federal agencies tend to get carried away with their authority, as there is no check on their activities.

Also, with rare exceptions, vendors have been treated, by many professional election organizations, like we have the plague. We are rarely called upon to provide real input (except when money is needed!) and our motives are frequently suspect. But in creating this Institute, it must be recognized that we are all partners in the election industry, with the same objectives, but with different perspectives. Every responsible vendor wants to produce a reliable election system that provides excellent service at a reasonable price. At the same time, responsible vendors realize that failures of electronic election systems hurt all of us. These objectives are not much different than those of the users or the professional election community.

A number of vendors, including Microvote, are also concerned with the integrity and neutrality of the selected "testing laboratories" (those authorities who will test voting equipment for conformity with the FEC standards), as well as with the integrity and neutrality of those organizations who select these testing laboratories.

The Institute would provide a number of critical services:

1. Become the central focal point for testing and certifying voting systems for conformity with the FEC standards. Up to now, this process has taken entirely too long and there is suspicion that it may even have become tainted by possible misallocation of funding.

Another important function is standardizing state certification requirements. Vendors now waste considerable time and money trying to get certified by 50 different election agencies. The Institute could provide a central certification service that might be accepted by these state agencies. This would result in lessening burdens on state election offices, lower vendor costs, and more timely certifications. Also, when vendors make changes or enhancements, such would be tested once by the Institute and approved for use in subscribing states.

Let me provide a specific example. In those states that require straight party voting during general elections, two types of "cross-over" voting are now used. One of these types allows a voter to cross-over in one action, while the other type requires the voter to make two separate actions. In 1992, several states moved from one type of cross-over voting to the other. The Institute, had it existed, would simply have certified that the equipment in question could handle both types of cross-over voting. That certification by the Institute would have saved many vendors considerable time, money and effort in obtaining certification at the state level.

2. Update and augment the FEC voluntary standards to cover areas outlined above as well as others documented in the professional literature.

3. Take all the extant material on security and auditability of electronic tabulation systems, and other critical subjects, digest it, *and get it into the hands of the election community in a form they can use*. Copies of these documents should be sent, free of charge, to every state and local election agency in America.

What would this do? If the marketplace becomes sensitive to these issues *by incorporating security and auditability requirements into relevant RFPs*, the vendor community will respond.

4. Take an aggressive role in sensitizing election officials to the need to directly address security and auditability matters. How? By accepting an active speaking role before the various national and state meetings of election officials where such ideas can be actively promoted. Vendors attend these meetings. They will get the message loud and clear.

5. Actively encourage innovation in the election field. All of us have plenty of good ideas for better products and services. However, there really is no effective forum for the exchange of ideas between users, the public interest community and the vendors. We would appreciate and use an Institute comprised of technical and other experts with whom we could exchange ideas.

6. Provide one location for vendors to escrow software and deposit other confidential data. Our immediate concern is the proliferation of sources for these data and the fact that some of us have had our confidential information used for profit motives. The Institute would provide one central repository with established procedures for accessing confidential data and information by state and local governments, and other parties.

7. Provide a source of technical assistance for state and local election offices. The facts are simple. State and local election offices are never going to obtain adequate staff or other technical resources to deal with vendors and to provide vendor oversight. The Institute can provide at least some of these oversight and support services especially when state or local agencies choose to purchase new electronic voting systems.

8. Provide a source for users to turn to for information and facts on the performance of the various vendors in the field. While this is, admittedly, a very sensitive area, it is essential to recognize that there are several systemic problems with certain electronic vote tabulation systems. Vendors who have had continuing problems will not make necessary changes until these problems are, in effect, exposed.

I feel it essential that this Institute, or some other organization, continue Roy Saltman's excellent monograph series on automated vote counting systems. Indeed, it is now time for a new publication analyzing problems encountered by certain optical scan and DRE voting systems, as these are now the major growth vendors.

9. Provide an active and ongoing research forum for critical technical election issues, such as voting by phone, early voting and other new developments in the election field.

Thank you for the opportunity to share some thoughts with you.

Panel on Protection of Intellectual Property

Intellectual property is fast becoming an important national asset for fueling our nation's economy and its international competitiveness. As our manufacturing base erodes further from competition with labor-rich and lower-paying third world countries, the services and information sectors of U.S. commerce grow in size and importance as a component of our exports and as a base for future new growth industries. Leadership in intellectual property development has been achieved through generations of education and cultural standards and cannot be easily duplicated; the products--the expression of ideas and invention--can, however, be easily copied.

A little noticed fact is that there are more personal computers than plows and more computer user-operators and programmers than farmers. Yet, ethical behavior has not kept pace with technological change and growth. While no one would think it legal to steal a farmer's crop products, far fewer people would consider unauthorized copying of fee-based electronic media products as stealing. Such products, whether for entertainment, commerce or education, have value to people and laws exist to protect author royalties and motivate new development. But the ubiquity and ease of the personal computer as a copying device provides a convenient private method for circumventing those laws.

Access control technologies are advancing to provide protection of information through user authentication and encryption techniques. This panel will explore some of these new technologies and industries' (e.g. cable, paper and electronic publishing, software development are multibillion dollar industries) economic and legal concerns regarding intellectual property management.

Panelists have been chosen to give the subject of protection of intellectual property in-depth coverage with respect to major current activities in the developmental, operational and legal areas. The panel is chaired by a person with extensive experience in the development of and controlled access to highly sensitive databases, and the panelists selected include the General Counsel of the Software Publishers Association; an attorney specializing in the area of intellectual property and partner in a well-known Washington, DC law firm; a key architect of the state-of-the-art National Research and Educational Network (NREN); a nationally known leader in the cable industry who is spearheading advanced technological efforts to support expanded cable services; and an executive in a firm involved with protecting intellectual material with improved access control and information privileging armamentaria.

Chairman: Gerald S. Lang, President, Harrison Ave. Corp.

- Dr. Aleksander T. Futro, Dir., Technical Assessment, CableLabs, Inc.
- Dr. Steven Wolff, Dir., NCRI Networking Comm & Research Infrastructure Div, NSF
- Michael T. Platt, Partner and head of Intellectual Property Law Section, Dickinson, Wright, Moon, Van Dusen & Freeman
- Ilene Rosenthal, General Counsel, Software Publishers Association
- Burton G. Tregub, President, SPIDA, Inc.

OUTLINE OF PRESENTATION BY MICHAEL T. PLATT*
ON
PROTECTION OF INTELLECTUAL
PROPERTY EMBODIED IN COMPUTER SOFTWARE

I. Trade Secrets

- A. Scope of protection.
- B. Definition.
- C. Requirements for establishing trade secret.
- D. Possible conflict between state trade secret laws and federal patent and copyright laws.
- E. Effect of "reverse engineering."
- F. Remedies available for misappropriation of trade secrets.

II. Copyrights

- A. Scope of copyright protection.
- B. How copyright in a work is achieved.
- C. Advantages of copyright protection vis-a-vis other forms of protection.
- D. Effect of "reverse engineering."
- E. Remedies available against copyright infringers.

III. Utility Patents

- A. Patentable subject matter.
- B. Advantage of patent protection vis-a-vis copyright and trade secret protection.
- C. Remedies available against patent infringers.

IV. Design Patents

- A. Patentable subject matter.
 - 1. Icons.
- B. Advantage of design patent protection vis-a-vis other available types of protection.
- C. Remedies available against design patent infringers.

*partner and head of the Intellectual Property Law Section of the law firm of Dickinson, Wright, Moon, Van Dusen & Freeman, 1901 L Street, NW, Suite 800, Washington, D.C. 20036

**Don't Copy That Floppy:
Software Piracy Is A Bigger Problem Than You May Realize**

Ilene Rosenthal
General Counsel
Software Publishers Association

Many businesses don't think software piracy is a serious problem. Davy McKee Corporation, a construction engineering firm headquartered in Pittsburgh, PA, learned the hard way. On November 19, 1990, their Chicago office received a surprise visit from representatives of the Software Publishers Association accompanied by U.S. Marshals. After meeting with the firm's executive management, the audit team went into action, searching the hard disks of their computers for unauthorized copies of software. Following this, the court required Davy McKee to perform an audit of all personal computers located at all of the company's other offices, including those in Pittsburgh, Tulsa, Houston and two locations in California. In February, 1991, Davy McKee agreed to a settlement of \$300,000 with the SPA. The settlement also required them to destroy all unauthorized copies of software and to institute formal internal copy control procedures at all of their offices.

This scenario is taking place, with increasing regularity, at companies large and small across the United States. Managers at Davy McKee, like many others, didn't think about the fact that software piracy is against the law, and their company was at risk every time an employee made an unauthorized copy of software. But the practice has become so widespread, with the software industry losing over a billion dollars annually in the United States alone, that the industry is fighting back. And corporate America is learning the hard way that piracy doesn't pay.

The unique dilemma that software piracy poses for information managers is that copying software is so easy to do and so difficult to control. In fact, the software industry is the only industry in the world that empowers every customer to become a manufacturing subsidiary. Every computer user has all the equipment necessary to make a perfect copy of a software product. The industry's challenge is to convince users that just because they can make a copy doesn't mean that the law permits them to do so.

Many employees are often confused about what is expected of them when it comes to software use, and many companies do not articulate a clear software policy. But one thing is clear. Federal law states that it is illegal to make unauthorized copies of software except for archival or back-up purposes. Companies and individuals who break this law can be liable for as much as \$100,000 for every software copyright that they infringe.

The Scene of the Crime

Unquestionably the environment in which software piracy is most prevalent is the corporate workplace. More than half of the revenues lost from piracy are a result of "softlifting", a crime often committed by employees who are otherwise law-abiding citizens, and who make copies of software to use in the office or to take home. People who would never think about stealing a candy bar from a drug store have no qualms about copying a \$5000 software package. The scene of the crime is not only corporations, but schools, non-profit organizations and government agencies - even law enforcement agencies.

Software piracy's negative impact is more far-reaching than most people realize. In addition to the billions lost annually as a result of domestic piracy, international piracy costs the software industry between \$8 billion and \$10 billion each year in lost revenues. In fact, an entire computer platform has been lost to software piracy. In 1985, the Atari ST became so identified as a pirate's machine that software developers refused to write programs for it,

and it has all but disappeared from the market. And the consumer lost a low-priced computing option.

An often overlooked cost of piracy is the cost to the user himself. When users copy software, they miss out on many of the valuable benefits of purchasing authorized software. These include a variety of user manuals and tutorials, customer telephone support and notification of and information about upgrades. Services like these are crucial to the value of software product.

Perhaps the highest cost of software piracy, and ultimately one that we all will pay, is the cost to the U.S. economy. Over the last 10 years the U.S. software industry has become an important national resource. Approximately 75% of the software used in the world today is developed in the United States. By depriving the software industry of billions of dollars in revenue, software pirates jeopardize U.S. leadership in the important high-tech market by slowing down the development of new products. In a time when many claim that the United States is losing its competitive edge in technology, it is critical to recognize the role that the software industry plays in maintaining the U.S. position in the global marketplace.

Taking Aim at Pirates

Over the past three years, the SPA has collected more than \$5 million in penalties from software pirates and generated a substantial amount of new sales for the software industry as a whole. Recoveries from settlements are used to fund future litigation as well as anti-piracy educational efforts.

The SPA targets pirates based on tips received from a variety of sources, primarily its anti-piracy hotline (1-800-388-7478). About 10 calls a day come in from disgruntled or former employees, even temporary employees. The SPA's first three raids in New York were based on evidence provided by a single temporary worker.

While the SPA has filed lawsuits against more than 150 companies, it receives many more reports that do not lead to legal action. Many of these are resolved with a cease-and-desist letter. Addressed to the president of the company, it identifies the software the company is suspected of pirating and warns the company to cease and desist illegal software usage.

For more serious offenders - more than 200 to date - the SPA requests an audit. Many companies prefer this option as an alternative to litigation. During the audit process, an SPA representative observes as the file directories of each PC in the company are printed and the directories are compared to purchase records. Before this procedure, the company agrees to pay to the SPA the retail price of all unauthorized software found during the audit. It also agrees to destroy the illegal copies and repurchase all software that is necessary for the company to operate legally.

In cases the SPA believes are appropriate for litigation, it will often obtain an ex parte TRO (often referred to as a search and seizure order) from the court. This order empowers representatives of the SPA, accompanied by U.S. Marshals, to enter the premises of an organization and conduct a surprise audit of the company's PCs. Based on the evidence gathered from these raids, the SPA will negotiate a settlement with the defendant. To date, the SPA has never lost a case, and recent settlements have been high - with several companies paying between \$200,000 and \$300,000 in penalties.

The SPA has been leading the fight against piracy in North America since 1988. At that time, a dozen SPA members formed the Copyright Protection Fund to pursue legal action against pirates and raise public awareness about the problem. Current participants in the Fund include Adobe Corporation, Aldus Corporation, Apple Computer, Autodesk, Inc., Borland International, Inc., Brightwork Development, Central Point Software, Claris Corporation, Fifth Generation Systems, Funk Software, IBM Corporation, Lotus Development Corporation, Micrograx, Inc., Microsoft Corporation, Novell, Inc., Software

Publishing Corporation, Symantec Corporation, WordPerfect Corporation, Xerox Corporation. Each serves as a plaintiff in cases of litigation involving pirated copies of their software.

The Copyright Protection Fund fights software piracy on behalf of the entire PC software industry, not simply the twenty-one members of the Fund. Therefore, Fund participants take action against businesses and individuals who pirate not only business applications but consumer and educational software, as well. In cases where consumer and educational software is found to be pirated, those publishers are invited to join as additional plaintiffs in the suit.

Getting Legal

It's clear that piracy lawsuits and audits are cause for companies to give more serious thought to the software practices of their employees. The only way to protect your company is to educate employees, rigorously enforce anti-piracy programs, and conduct periodic audits.

When General Motors called the SPA to request information about this issue, the two organizations worked together to develop GM's comprehensive anti-piracy program. In fact, GM was way ahead of the game. The company conducted its first internal audit in 1985.

Unfortunately, many companies believe that a written policy against software piracy is adequate protection from a lawsuit. However, anti-piracy policies are useless unless accompanied by a program of stringent enforcement. By neglecting software piracy, management, in effect, condones it.

Top management can even encourage piracy, sometimes unknowingly, by trying to cut costs. One company had a written policy decreeing that there was no reason for the

accounting department to have word processing software. This policy ignored the fact that sooner or later, every employee had to write a letter.

Policies regarding software use need to be part of the employee orientation program, and they should be prominently displayed in the workplace. Another important point for management to remember is that money must be budgeted for software whenever new hardware is purchased. If a PC is purchased for a new employee, software must be purchased as well.

Managers must also address "back door" piracy, where employees bring a favorite software program to work and make copies of it for co-workers. Technically, the company is still responsible for these copies, and in the case of a lawsuit, they would be treated no differently from standard office software.

Local area networks are growing in popularity within the corporation, and as a result software piracy takes on a new twist. While piracy on a network does not involve the traditional form of copying, it still means the unauthorized use of software. This is a growing concern for software publishers, considering that more than 15% of all software now runs on networks.

Software publishers are wrestling with the problem of how to properly license software for a network so that the end users can stay legal with relative ease. Currently, software can be licensed to a network in three different ways. Software may be licensed to a machine, to an individual, or to the number of concurrent users.

Because network license agreements vary from publisher to publisher, managers should check with the publisher of the software they use whenever a question arises regarding proper network use. The important thing to remember is that, since the company will ultimately be held responsible for any copyright violations, the management must be sure

that employees understand how to use their own network software legally.

In the long run, the battle against piracy in corporate America can only be won by helping corporations get legal, stay legal, and educate their end-users. A strategic ally in the battle is, of course, the information manager. It should be his or her responsibility to educate users about the risks of software piracy and to closely monitor the software practices of employees.

The SPA lends a hand with a free Self-Audit kit. More than 60,000 Kits have been distributed in the last 2 years to MIS managers, small business owners, corporate trainers and network administrators. Designed to help organizations manage their internal software practices, the Kit includes an auditing software program called SPAudit. SPAudit is designed to help inventory commercial software on a company's hard disks. It automates the software audit process by searching for over 625 of the most commonly used programs in business today and printing out a list of all programs found. Users can then compare the printout to purchase records. SPAudit comes in both an MS-DOS and a Macintosh version.

The Self-Audit Kit also includes two pamphlets: *Software Use and the Law*, a summary of the Federal Copyright Law as it applies to software, and *Is It Okay to Copy My Colleague's Software?*, a question-and-answer guide that helps end users to understand legal software use. In addition, the Kit contains a suggested corporate policy statement and a sample memo to employees.

Recently, the SPA released "It's Just Not Worth The Risk", a 12 minute videotape on software piracy. This video is a useful tool for instructing business users about the legal use of software products and is available for \$15.

Ultimately software piracy is a managerial problem that *every* company with PCs must

address. Everyone struggles with the problem, but managers who ignore this issue may find themselves in the same unfortunate position as did the managers at Davy McKee Corporation. The primary objective of SPA's anti-piracy campaign is to provide the necessary resources to help companies avoid that type of scenario.

The SPA encourages organizations to monitor software resources and educate employees about legal software use. The SPA's Self-Audit Kit and anti-piracy videotape simplify these tasks. These materials may be obtained by calling the SPA at (202) 452-1600, or by sending a written request on company letterhead to the SPA, 1730 M Street, N.W., Suite 700, Washington, DC 20036. There is a \$15 charge for the videotape.

PANELIST STATEMENT

Panel: "Protection of Intellectual Property"

Sixteenth National Computer Security Conference

Panelist: Burton G. Tregub, President, SPIDA, Inc.

Panel Session Schedule: Wednesday, September 22, 1993, 2:00-3:30 PM

Title: The Future of Intellectual Property Protection
-A Smart Card Technology Solution-

Personal computers and microcomputer-based products are growing more pervasive and portable in all consumer and business market and application sectors. Information and consumer programming networks are proliferating in utility. Announcements are being made almost daily about important new personal electronic delivery and storage media devices.

The density and cost effectiveness of information distribution media continues to increase in order to satisfy new requirements for user portability, decentralized information availability and system transportability. Market demands for user mobility, decentralized information availability and system transportability require that network connections and storage media remain standardized in order to lower the costs of utilization. Standardization simplifies the ability of unauthorized personnel to copy the data carried by a network or on a media, leaving information providers and publishers much more susceptible to financial losses by increasingly sophisticated attacks.

Whether for physical delivery or for distribution through cable or satellite operator or telephone common carrier, major investments are being made to develop an incredible selection of knowledge or entertainment based intellectual products. The multimedia industry will introduce tools and procedures to allow on-demand composition of new intellectual products and material, partially from existing works.

Encouraging this trend is essential for rapid development of our country's intellectual property products and services. Properly empowering and rewarding producers and authors is essential in order to stimulate creativity and align it with consumer demands and profitability.

Quoting from a 1992 Smithsonian Institute Symposium, "For the multimedia publisher, securing the rights to each work is imperative regardless of whether the work will be fully used, partially reproduced, included in a collage, or manipulated into a "new" work...Fears of artists and copyright owners are growing as they recognize how easily their works may be reproduced en masse and/or manipulated beyond recognition. Once their work is digitized, it may be reproduced into practically "perfect" pirate copies. These pirate copies may then be stored on databases,

Title: The Future of Intellectual Property Security--A Smart Card
Technology Solution
(cont'd)

downloaded from a network and used over and over again. Further, because transferring the pirated copies leaves no trail, pinpointing the infringers is nearly impossible. The ease with which digitized works may be pirated should be considered within the licensing framework...Creators and copyright owners should welcome multimedia as a high-tech opportunity to disseminate their works, while at the same time keeping in mind that the potential for uncontrolled distribution, unlimited reproduction and gross manipulation is real."

The growth of intellectual property must be stimulated; it is the most valuable national resource to fuel this country's international competitiveness in the 21st century, and is as important to our economy as the development of real property has been in our past. Technologies and products have been applied piecemeal to protect access and integrity of all forms of electronic intellectual program material. This has often led to complex and costly integration and operation of incompatible products.

Current password solutions for providing accurate control for multi-user data delivery and integrity have been well publicized with regard to vulnerabilities, even from amateur hackers. Reliance on "fair use" doctrines contains significant risks and expenses for complete enforcement, particularly where valuable information services are provided. To maintain protection against unauthorized use, mobility or media portability are often precluded outside a closed system or independent of a host network, thereby constraining information availability. This is not acceptable; it is, at best, a very poor compromise.

New advances in smart card technology facilitate exercising a chain of portable accountability that economically and transparently links each copyright owner to each user. This will open up the supply boundaries for program producers and authors and expand availability for users. Unlike other specialized products which require the user to maintain different models for each application, the smart card's ability to concurrently host separate unique identification codes allows it to manage multiple applications and delivery systems. The integrity of pricing and access control for each application can be separately retained.

Smart card technology can add the dimensions of absolute confidentiality, security, and usage metering while maintaining total portability and on-demand accessibility for physically delivered or downloaded electronic material. This provides inherent compatibility with pre-packaged electronic storage material (e.g., CD-ROMS, VCR video tapes), or material composed and downloaded to a user's own storage system (e.g., PC hard disk, flash card or CD-R media). Smart card solutions can easily be

Title: The Future of Intellectual Property Security--A Smart Card
Technology Solution
(cont'd)

integrated with portable digital assistants, desktop PC's and cable-TV set-top boxes. Subscribers and information can be completely mobile; secure access and operations are not tied to specific access ports or equipment locations.

The systems can overlay all network architectures and encryption techniques, providing an open, multiple-source competitive environment for continually improved performance and decreasing costs. Multiple programs or information materials, each with separate pricing strategies, can be combined on a single medium or on a media set. Each subscriber can have individual access rights, dependent upon their requirements and the fees paid. Such rights can be changed dynamically without reissuing target storage media. Off-line transaction auditing of program material usage can capture additional revenue fees and market feedback information.

Through electronic signature verification, personal smart cards can provide verification that the program material has not been altered since being published by the author; any user can verify, through the smart card transaction, that a "computer virus" has not been injected nor has the author's work been modified by a third party.

Through new designs for smart card system solutions, the same advances in semiconductor technologies that allow electronic information to be so easily and quickly copied and modified without a trace, can now be applied to protect author's rights and provide the motivation and economic rewards to produce new products to benefit society. Additionally, users and consumers ultimately benefit from an increased diversity of products, from the lower distribution costs that electronic media systems can provide, and from a totally portable and accessible point-of-purchase/point-of-use environment that smart cards can enable.

THE PRIVACY IMPACT OF TECHNOLOGY IN THE 90'S

Session Chair
Wayne Madsen
Computer Sciences Corporation
Integrated Systems Division
Rt. 38 West
Moorestown, New Jersey 08057

Recent developments in digital communications, storage media, microchip, automatic identification and data base management system technologies point to a growing impact on individual privacy. This panel shall discuss some of the relevant issues involving the impact of new technologies on privacy. The panel shall also address some of the domestic and international legislative and legal developments in the field of information technology security and privacy.

---**---

Paper Presentation
Christine Axsmith
Mantech
Washington, DC

Topic: The OECD Guidelines for the Security of Information Systems: A Look to the Future

The Organization for Economic Cooperation and Development (OECD), an international organization comprised of the leading industrialized nations in the world (European Community, Japan, Australia, Canada, New Zealand, Sweden, Norway, Switzerland, Austria, Finland and the United States), has issued a set of information system security guidelines. This represents, after the European ITSEC initiative, the second international effort to harmonize information security standards.

This presentation shall discuss the impact of these guidelines on information security requirements in the United States. Particular attention will be paid to the extradition of computer criminals between OECD member states as well as required changes in Federal rules of evidence and information security legislation. Special focus will be brought to the Digital Signature Standard (DSS) and its relation to these issues.

---**---

Speaker: Gerard Montigny
Privacy Commission of Canada
112 Kent
Ottawa, Ontario, K1A 1H3, Canada

Topic: Today's Technology-Tomorrow's Privacy

One of the Canadian Privacy Commissioner's themes this past year has been "Future Challenges to Privacy in Canada." To this end the Privacy Commission has undertaken an initial study of information technology and its impact on privacy today and into the 21st century. Although this study continues, a number of interesting and relevant privacy issues have emerged. This presentation will attempt to highlight some of the key aspects of the study. This will include such areas as secondary use of personal information, electronic highways, personal communications networks, wireless communications, e-mail and bulletin boards, surveillance and monitoring and smart cards. The presentation will also include an update on what is happening in the privacy front in Canada.

Key words: bulletin boards, electronic highways, electronic mail, monitoring, personal communications networks, personal information, privacy, secondary use, smart cards, surveillance, wireless communications.

---**---

Speaker: John Hamlet
Deacon House
18 Park Avenue
Beverly, New Jersey 08010
jhamlet@everglades.motown.ge.com
COMPUSERVE: 76430,2334
PRODIGY: PDHK43A

Topic: Privacy and Biometrics

Biometric identification is becoming a more prevalent part of an individual's interaction with his or her surroundings. Rather than identifying individuals based on what they possess (an identification card) or what they know (password), many information technology systems are relying on processes that identify people based on what they are.

This presentation defines biometric terminology, reviews several works on the subject, provides a brief history of biometric development, discusses the accuracy of biometric identification techniques and analyzes areas where the use of biometrics may endanger individual privacy.

Key words: authentication, biometric identification, DNA analysis, eye retina analysis, facial analysis, fingerprint analysis, hand geometry analysis, keystroke analysis, signature analysis, voice analysis.

---**---

Speaker: Julien Hecht
Miles and Stockbridge
Attorneys at Law
10 Light Street
Baltimore , Maryland 21202

Topic: Legal Liability of Disclosure of Private and Inaccurate Data

Personal data is increasingly used to make critical decisions affecting an individual's financial, medical, and social well-being. When such data becomes corrupted or when it is purposefully or accidentally released to unauthorized recipients, individuals can suffer immeasurable harm.

This presentation shall focus on the civil and criminal liability associated with the purposeful or accidental disclosure of private or inaccurate data. Particular items to be addressed include tort liability, statutory liability, contract liability and strict liability.

Key words: civil liability, contract liability, criminal liability, data inaccuracy, data disclosure, statutory liability, tort liability

---**---

Speaker: Juhani Saari
International Baseline Security OV.
Kauniainen, Finland

Topic: Privacy Implications of Smart Cards

This speaker is the Chairman of the European Smart Card Application Technology Association (ESCAT). He has been at the leading edge of smart card applications in the international banking, transportation, point-of-sale, EDI and debit card arenas.

---**---

Stewart Baker
General Counsel
National Security Agency

Topic: Limits on the Use of Keystroke Monitoring for Security

As concern grows about hacker attacks on computer systems, more and more companies and agencies are taking advantage of key-stroke audit or monitoring software. Such software allows managers to review the activities of system users and to identify unusual behavior that may signal an unauthorized intruder. (This technique is related to the method used by Cliff Stoll to foil a German hacker in "The Cuckoo's Nest.")

But the use of key-stroke monitoring for security purposes is fraught with legal peril. Companies and managers that install such software must act with care to avoid falling foul of criminal statutes that protect the privacy of electronic data. And even when monitoring has been structured to avoid criminal liability, companies would be well advised to adopt a "code of conduct" for those employees who exercise this extraordinary power.

Panel Summary

Electronic Crime Prevention

Robert Lau, *Chairman*

National Security Agency

Speakers

Special Agent Jim Christy, *AFS/OSI*

Special Agent Jack Lewis, *Secret Service*

Special Agent Mark Pollitt, *Federal Bureau of Investigation*

During this session the audience can talk to federal investigators and law enforcement officers who understand the impacts of electronic crimes to organizations and individuals.

Special Agent Jim Christy will discuss Computer Crime programs in the Department of Defense. Special Agent Jack Lewis will discuss programs in the Secret Service. Special Agent Mark Pollitt will discuss programs in the Federal Bureau of Investigation.

They will then discuss: 1 - the cooperative efforts that they use to enforce their programs; 2 - the laws for which their organizations are required to enforce; and, 3 - the techniques used to investigate electronic crimes.

The audience will be able to talk with the speakers and discuss the impacts of electronic crimes against organizations and individuals.

INTRODUCTION TO THE FEDERAL CRITERIA: USER OVERVIEW AND UPDATE

SPEAKERS: CDR Debbie Campbell
Eugene Troy

NSA
NIST

OVERVIEW

This session will provide the participants with a general overview of the Federal Criteria project, its goals, scope and driving principles. The history of the project will be presented, as well as the planned milestones for its completion. The discussion will include how the criteria fits into the trusted systems arena, including international harmonization of all security criteria. Terminology will be clarified so that all participating in Federal Criteria discussions will have an equal understanding of the underlying principles. Lastly, the status of the Federal Criteria will be provided, including a summary of comments on the December, 1992 version and the June, 1993 workshop. The session will provide the participant with the basic knowledge for attending all other sessions pertaining to the Federal Criteria for Information Technology Security.

SUMMARY

The original goal of the Federal Criteria project was to create a U.S. national standard for computer and information system security that: 1) protects previous investment in trust technology; 2) adds value to current criteria; 3) develops a framework for defining new customer requirements; and 4) promotes international harmonization of criteria. This standard was intended to provide information on how to specify requirements for Information Technology (IT) product security, to include a fundamental structure for stating those requirements and a set of common "building blocks" to assist in the development process.

A first draft of this document was released in December 1992. This was followed by a seminar in January 1993 at which the concepts and goals of the project were presented. As a result of this seminar over ten thousand copies of the draft Federal Criteria (FC) were distributed for comment. In February 1993 an equivalent briefing was provided to the European Community (EC). This further expanded the audience of the document, as well as the project as a whole. The draft FC received an extensive review and over 20,000 comments were submitted to NIST and NSA for revision to the document. Reviewers were then invited to a two-day workshop in June 1993 to discuss several issues raised in the comments. These discussions were recorded and are being used as another input to the document.

The project has since been elevated to the international level. The United States, Canada, and the European Community have agreed to work together to develop the Common Criteria (CC) which will harmonize all the existing criteria. This effort is expected to begin in early fall of 1993 and be completed in the spring of 1994. Specific inputs include: 1) the Information Technology Security Evaluation Criteria (ITSEC) and the experience gained to date with the ITSEC in the form of suggested improvements; 2) the Canadian Trusted Computer Product Evaluation Criteria

(CTCPEC); the draft Federal Criteria for Information Technology (FC) and the comments received on the draft FC document, including the results of the FC invitational workshop; and 4) the Trusted Computer System Evaluation Criteria (TCSEC) and experience gained over the past ten years in conducting trusted product evaluations. The resulting CC will then undergo extensive international review and testing before becoming an international standard.

FEDERAL CRITERIA FOR INFORMATION TECHNOLOGY SECURITY: PROTECTION PROFILE DEVELOPMENT A TUTORIAL

SPEAKER: Janet Cugini
MAJ Mel DeVilbiss

NIST
NSA

OVERVIEW

This tutorial will provide the participants with an understanding of volume one of the Federal Criteria for Information Technology Security - Protection Profile Development. This tutorial will provide information on the concepts of security functional requirements, development assurance requirements, and evaluation assurance requirements as they relate to the construction of a Protection Profile. The process of formulating the Protection Profiles will be discussed, including threat analysis and requirement dependency analysis. This tutorial will give the participant a general understanding of Volume one of the Federal Criteria for Information Technology Security.

SUMMARY OF TUTORIAL MATERIAL

The Federal Criteria for Information Technology Security is currently published in two parts: a description of how to develop and register protection profiles and a set of sample protection profiles. This tutorial will provide the information contained in the first of these volumes.

The Federal Criteria has four main objectives: to develop an extensible and flexible framework for defining IT product security in an ever-changing technology; to enhance existing product security criteria; to facilitate international harmonization of product security criteria; and to preserve the fundamental principles of product security. The first objective acknowledges that the information technology industry is operating in a rapidly changing environment. A security criteria must allow for changing technology and therefore must be flexible enough to take into account these dynamic parameters. The Federal Criteria also acknowledges that there has been substantial work in the field of criteria and aims to add to the field. This addition is the concept of protection profiles. Protection profiles are bundled sets of criteria that define a level of trust for a given set of products and registered at a central source for use by developers, evaluators and consumers. Protection profiles can be added to the registry as technology progresses. Related to this objective is the objective to harmonize the United States product security criteria with those of other nations. The goal is to have one criteria (set of protection profiles) to which vendors can build products and have a world-wide market for those products. Lastly, the Federal Criteria acknowledges that historical knowledge-base and investment must be maintained. The work performed under the TCSEC is valuable, still applicable, and must be maintained.

The first volume of the Federal Criteria provides the structure of protection profile. It contains information which describes the use and threats to which the profile is meant to be applied. It also contains a list of the functional requirements, development assurance requirements, and evaluation assurance requirements that are included in the profile. These three sections are loosely equivalent to a class of the TCSEC. The protection profile is then analyzed to ensure that the

profile is technically sound, useful in the marketplace, evaluable, substantially different from other protection profiles in the registry, and consistent in form and detail to the other protection profiles. A dependency analysis is performed to ensure that the grouping of requirements are logical as a unit.

The first volume of the Federal Criteria also provides a set of components that can be used to build a protection profile. These components include functional areas such as identification and authentication, audit, access control, intrusion detection, and security management. The development assurance components include areas such as interface definition, modular decomposition, configuration management, and functional testing. Evaluation assurance components include areas such as test analysis, independent testing, design analysis, and operational support review. Components are rated, so that some are stronger than others. For instance, there are four components provided for access control. The second access control component extends the first, the third extends the second, and the fourth extends the third.

FEDERAL CRITERIA FOR INFORMATION TECHNOLOGY SECURITY: REGISTRY OF PROTECTION PROFILES A TUTORIAL ON INITIAL PROTECTION PROFILES

SPEAKER: Dave Ferraiolo
Lynne Ambuel

NIST
NSA

OVERVIEW

This tutorial will provide the participants with an understanding of volume two of the Federal Criteria for Information Technology Security - Registry of Protection Profiles. There are currently two families of Protection Profiles in the Registry: Commercial Security Requirements (CS) and Label Based Protection Requirements (LP). An overview of each of these families will be provided, as well as a detailed discussion of the requirements included in each of the protection profiles within the two families. This tutorial will give the participant an understanding of the content of each of the Protection Profiles, the source of their requirements, and their intended uses.

TUTORIAL CONTENT

The Federal Criteria for Information Technology Security is currently published in two parts: a description of how to develop and register protection profiles and a set of sample protection profiles. This tutorial will provide the information contained in the latter of these volumes.

The first volume of the Federal Criteria provides the building blocks for developing and analyzing protection profiles. The current second volume contains example protection profiles based on existing product security criteria. The CS protection profiles are based on the Minimum Security Functionality Requirements (MSFR), which is derived from the TCSEC C2 class, as well as the Bellcore Standard Operating Environment Security Requirements, the Commercial International Security Requirements, and the Computers at Risk report. The LP profiles are based on the multi-level security classes of the TCSEC.

The CS1 protection profile contains the equivalent requirements to the C2 class of the TCSEC. This provides an equivalent rating for those products that have been developed and evaluated under the TCSEC C2 class. The CS2 protection profile adds several specific requirements derived from the MSFR. Added features include access control lists, the separation and use of privileges, specific identification and authentication rules for all trusted subjects, added audit capabilities, as well as several assurance enhancements. The CS3 protection profile adds the concept of role-based access controls to the commercial security protection profiles. In addition, it provides for stronger authentication mechanisms, such as authentication generators or biometric scanners. In addition, several assurance requirements have been added to the commercial security requirements in this protection profile..

There are currently four LP protection profiles, one for each of the TCSEC multi-level classes. The intent was to translate the TCSEC classes into protection profiles in order to protect past, (as

well as present) investment in products developed and evaluated under the TCSEC. The classes have been transferred largely intact, although the requirements have been reorganized to fit into the structure of protection profiles defined in the Federal Criteria. However, there is also a set of interpretations of TCSEC issues that have become part of the criteria for evaluating products. The LP protection profiles include clarifications of some of the requirements based on these interpretations. The end result is a set of profiles that both preserve investment in security products based on the TCSEC and provide a sound criteria for developing and evaluating multi-level security products in the future.

It is understood that these two families of protection profiles are only examples and that protection profiles for other types of security products will be developed. The initial protection profiles will serve as examples as to form and content for later protection profiles. They also ensure that well-accepted criteria for security features and assurances are adequately covered in protection profiles under the Federal Criteria.

Federal Criteria: Protection Profiles for Technology of the 90's
Distributed Systems
NCSC Conference Panel Summary

1. Panel Makeup

Chairman: Robert Dobry, NSA

Panelists: Jeremy Epstein, TRW
Ken Cutler, MIS Training Institute
Virgil Gligor, University of Maryland

2. Panel Summary

The Federal Criteria (FC) introduces the new concept of a protection profile. A protection profile is defined as an abstract specification of the security aspects of a needed Information Technology (IT) product. The profile is product independent, describing a range of products that could meet this same need. The required protection functions and assurances for an IT product are bound together in a protection profile, with a rationale describing the anticipated threats and intended method of use. The protection profile becomes the specification of the requirements for the design, implementation, and use of IT products.

Protection profiles are assembled from functional and assurance components. A functional component is a set of rated requirements for protection functions to be implemented in an IT product. An assurance component is a set of rated requirements for development and evaluation activities conducted by producers and evaluators during construction and independent assessment of an IT product.

As technology evolves, existing protection profiles will be modified or new profiles developed in response to a specific need for information protection. Modifications to existing profiles will be accomplished by allowing the functional and assurance components from which the profile is constructed to evolve to encompass the new technology, or by adding new components altogether. In either case the profile must still be applicable to the IT product for which the profile was originally developed. Consumers or producers with a unique security need could develop a new protection profile to fit that need. More typically, groups of developers, or sponsors, having similar needs could combine to develop one protection profile that meets their common needs. These profile sponsors could represent the Government or the private sector.

The protection profile is intended to respond to both the pull of consumer needs and to the push of advancing technology. Ultimately, the protection profile is to be a common reference among consumers, producers, and evaluators.

3. Protection Profiles for Distributed Systems
Virgil Gligor

The FC limits its scope to IT products which it defines as a hardware and/or software package that can be purchased as an off-the-shelf product and incorporated into a variety of systems. The FC specifically states that it addresses IT product

requirements only. "The composition of multiple IT products into an IT system is beyond the scope of this standard."

Although the current version of the FC focuses on IT products the trend in today's technology is towards networked distributed systems: distributed systems being systems comprised of several interconnected sub-systems with each sub-system sharing the resources of all the other sub-systems. The operating system for the distributed system is dispersed over the entire system instead of residing in one component.

Panel discussions concerning distributed systems will be divided into three main areas. The first section will concern itself with the inherent differences between the characteristics of product security and system security. The next section will deal with the ongoing efforts to develop a Distributed Systems Criteria. The final section will deal with how a distributed systems criteria would fit into the frame work of the Federal Criteria.

The inherent differences between the characteristics of product security and system security result from how the security issues that must be addressed in each case are defined. Because product security is only concerned with technical safeguards and assurances, the security issues are well defined and their characteristics can be easily objectively measured. A great deal of precision is obtainable in a product standard. System security, on the other hand, must take into account the interactions between the technical security in the products which comprise the system and the non-technical security in the system environment. These interactions are not currently well understood and would be the focus of the discussions.

The ongoing work to develop a distributed systems criteria is based on previous work, primarily a white paper on Trusted Distribution Systems put forth by Virgil Gligor in 1991. Criteria statements based on this white paper were developed. These criteria statements were then used to develop a distributed systems framework which identifies the required policy aspects. From this framework actual components were then developed. A working group was then formed to refine the material into an initial draft of a distributed systems security criteria. The discussion will focus on reporting on these efforts, with emphasis on any points of contention encountered.

Fitting distributed systems criteria into the framework of the FC could take two tacks. One proposal is to develop a family of protection profiles specifically for distributed systems. These profiles would be developed based on requirements currently found in the FC and new requirements developed specifically for distributed systems. The other proposal is for requirements to be added to the existing components which make-up the LP profiles so that these profiles would also be applicable to distributed systems. The discussion will focus on which of these two possibilities would best fit the need. One of the major factors to be considered is which of the possibilities would prove more compatible with the ongoing harmonization efforts. Time will be provided for this discussion.

Dr. Gligor, having worked in the area of distributed operating systems since 1978, is recognized as one of the foremost experts in the field. Having published over 60 scientific papers in the areas of security, reliability, and distributed systems, he has a vast knowledge of the security requirements for distributed systems. He will be relating this experience and providing insight into how the protection profiles for distributed systems were developed.

4. Protection Profiles for X-Window Implementations Jeremy Epstein, TRW

The office of the 90's is comprised mainly of workstations. These workstations generally have a windowing capability so the worker can view, and even work on, several files at one time. This type of working environment was not envisioned in the formulation of classes for the TCSEC. The issue was raised, however, once the Defense Intelligence Agency (DIA) formulated a criteria for Compartmented Mode Workstations (CMWs). The initial protection profiles provided with the Federal Criteria do not include one for windowing environments. This panelist will discuss which functionality and assurance components need to be developed for this specialized environment.

Mr. Epstein is recognized as one of the foremost experts in the field of trusted windowing environments. He will be relating his experience in this area and providing insight into how past experience could best be incorporated into a family of protection profiles for X-Window workstations.

5. Protection Profiles for Secure Workstations Ken Cutler, MIS Training Institute

The need for secure workstations is just as acute in the commercial world as it is for defense applications. A primary difference is the primary concern of the commercial world with the integrity of the system. Although the Federal Criteria includes profiles supposedly geared towards commercial products, these profiles in many cases contain requirements that most commercial vendors feel unnecessary. This panelist will discuss which functionality and assurance components need to be included in profiles that better meet the needs of the commercial world.

Mr. Cutler has over 19 years experience in the field of information security, auditing, quality assurance, and information services. He is a well known international speaker on information security and auditing topics. He is the primary author of the widely acclaimed Commercial International Security Requirements (CISR), which offers a commercial alternative to military security standards for system design. He will be providing insight into how the process of developing protection profiles will be handled within the commercial sector.

A PANEL DISCUSSION ON THE VETTING AND REGISTRATION OF PROTECTION PROFILES

PANEL CHAIR: LYNNE AMBUEL NSA

PANEL MEMBERS: SVEIN J. KNAPSKOG CHAIR, ISO SC27 WG#3
 LAWRENCE G. MARTIN NSA
 TBD

OVERVIEW

This panel will discuss the possible ways in which Protection Profiles could, and should, be reviewed for correctness, completeness and usability, and registered for use by trusted product developers and customers, as well as evaluators. Several possibilities exist. This panel discussion will ask several questions of the panelists, as well as of the audience. The purpose will not be to provide definitive answers but to stimulate discussion. The questions will include: who will provide the means to verify that a Protection Profile is valid? Do NIST and NSA need to verify each Protection Profile? Who will be trusted to provide Protection Profiles? What makes a Protection Profile worthy to be listed in the registry and what will be the criteria for refusing a Protection Profile? Will there be an appeal process for a protection profile that is rejected? Where will the registry or registries reside? These are all questions that must be answered in order for the concepts in the Federal Criteria to be put into practice. Panel members will present their views and will encourage the audience to provide input into this politically sensitive issue.

SVEIN J. KNAPSKOG - INTERNATIONAL VETTING AND REGISTRATION

Vetting Protection Profiles as a part of the analysis and registration process is seen by the potential users of such profiles as an absolute necessity. The Federal Criteria has, and will continue to have, a significant impact of the formulation of an international set of criteria. The vetting procedures for the criteria (e.g. protection profiles) will, for all practical purposes, be equivalent. Administratively, they will still be different, because of the organization doing the vetting will have to be recognized as an international organization, or a procedure on how to evaluate (and accredit?) national authorities handling the vetting procedure on behalf of the international community (ISO) must be established. However, there is no real conflict of interest here, a standard, or protection profile, that meets the U.S. Government and commercial security needs will also address many or all of the security needs of governments and commercial entities of other nations.

All the headlines contained in Chapter 3.5 of the Federal Criteria, should be properly dealt with in a vetting procedure. These include technical soundness of the protection profile, usefulness of the protection profiles, whether the requirements of the protection profile are evaluatable, whether the protection profile is adequately distinct for other registered protection profiles, and the

consistency of the protection profile with other protection profiles in form and level of detail. During the panel discussion, handling of the vetting process in a harmonized way in an international environment, preferably within the ISO administrative structure, will be discussed.

LAWRENCE G. MARTIN -

THE RISKS AND LIABILITIES OF VETTING AND REGISTERING PROTECTION PROFILES

The Federal Criteria is the first criteria that does not mandate specific sets of functionality and assurance classes or levels. Instead it provides a set of building blocks which allows the reader to develop their own functionality and/or assurance class or level into a Protection Profile. Unlike predefined classes or levels, the new Protection Profile must be analyzed for technical soundness, usefulness, evaluation capability, distinctness, and consistency. It is only upon the completion of this step (called vetting) that the new Protection Profile can be considered for registration.

The first draft of the Federal Criteria did not adequately describe the process of vetting new Protection Profiles. Yet, this concept is central to the success of the Protection Profile concept presented in the Federal Criteria. If it is not understood how Protection Profiles are registered, there can be no protection profiles for developers to build to nor ones for consumers to use in procurement requests. There must be a way to populate the registry in order for the goals of the Federal Criteria to be met.

There are several complex political and legal issues related to the registration of Protection Profiles that need careful and thorough study. Many questions must be resolved before the Federal Criteria (or Common Criteria) can be finished. It must be decided exactly whether, and how, this concept will work because the entire foundation of the Federal Criteria is predicated in this concept. The project will succeed or fail based on the degree of difficulty and the political/legal perceptions, as well as the actualities, of this vetting and registration process.

**EVALUATION PARADIGMS
AN UPDATE ON THE TRUSTED PRODUCT EVALUATION PROGRAM
AND THE TRUSTED TECHNOLOGY ASSESSMENT PROGRAM
A PANEL DISCUSSION**

PANEL CHAIR: Stephen Nardone, NSA

PANEL MEMBERS: Pat Toth, NIST
Judy Dye, DoD
Christina McBride, NSA

The purpose of this panel is to present the status of the two major efforts underway within the United States to develop and/or refine the process for evaluating a given product against a trust criteria. The two working groups will present the status of the programs, and future activities planned to incorporate their findings into the evaluation process. There will be ample time for questions and comments.

Trust Technology Assessment Program

In January 1993, the National Institute of Standards and Technology and the National Security Agency formed the Trust Technology Assessment Program (TTAP) Working Group. The goal of TTAP is to provide a commercial evaluation capability for lower level of trust products. The TTAP Working Group was given six months to:

- * study the feasibility of the TTAP concept
- * determine if TTAP meets the needs of users in the government and private sectors
- * further develop the requirements to establish TTAP

The presentation will review the findings of the Working Group and its recommendations for the establishment of TTAP.

Trusted Product Evaluation Program

In October 1993, the National Security Agency announced the results of a Total Quality Management effort to improve the TPEP. The Steering Committee will present the status of the implementation of process action team recommendations. Topics will include Security Design Advice; Intensive Preliminary Technical Reviews; Documentation Guidelines; Evaluator Resource Management; Vendor Involvement; Combined IPAR/Testing TRBs; Relevant Evaluation Reports; RAMP; Experimental Evaluations. Vendors participating on the panel will present their experiences working under the new process.

**Panel
European National Evaluation Schemes**

Ellen E. Flahavin, Chair, NIST

Klaus Keus, BSI (German Information Security Agency)

Jean Viale, SCSSI, French Government

Dr. Rainer Baumgart, RWTUV, Germany

Andrea Cummings, Logica, CLEF, UK

In the late 1980's different national IT-Security Evaluation Criteria were produced in several nations in Europe (e.g. UK, Germany, France). Using these criteria in combination with needed guidelines for the evaluation first national approaches of Evaluation and Certification Schemes were established.

Based upon these criteria and experience a first issue of an European harmonized IT-Security Evaluation Criteria (ITSEC) was published in June 1991. A first draft of the Evaluation Manual (ITSEM) describes the evaluation methodology.

These criteria form the foundation for the approach of a first common European Evaluation and Certification Scheme. The contributions of the involved partners (sponsors, Certification Bodies(CB), evaluators) are enhanced by Accreditation and Licensing Schemes for third party laboratories known as IT-Security Evaluation Facilities (ITSEF).

The European approach based on the European Standard set EN45000 builds the basis for European Multinational Agreement for Certificates which is currently being developed.

The practical experience of many evaluations and certifications have proven the practical use of the complete scheme during the trial period of the ITSEC.

Many accredited ITSEFs have demonstrated their competence through the evaluation of a wide range of IT-products and IT-systems.

In addition to the classic evaluations of operating systems, the ITSEC is used in state-of-the-art application environments. There is a growing worldwide interest in the use of smartcard systems in the information security area. RWTUV (a German ITSEF) evaluated the SIEMENS-NIXDORF multi-application smartcard SCC V2.0 with the quality level Q4 against the German IT-Security Criteria. For the German DBP-Telekom's electronic-signature system "TELESEC" based on smartcard-technology RWTUV is performing several ITSEC E4 level evaluations. In Germany, certification of smartcards is one of the focuses for future applications, e.g. in health systems and medical and social assurance environments. This panel will include a brief overview of smartcard architecture and standards, followed by

information about the evaluation experience in smartcard systems and components. The panel will also discuss functional aspects with regard to the ITSEC.

Other applications, such as database systems (DBMS) have been evaluated in the UK, partially with international cooperation. Widely accepted use of the ITSEC in the European Evaluation and Certification Scheme demonstrates the existence of a competent evaluation community which continue to grow as other European EC members and EPTA Nations participate.

The panel will also discuss the wide range of products evaluated in the UK under the ITSEC scheme. The presentation will then assess the possible results of applying the draft Federal Criteria to these products, and hence conclude whether any modifications would need to be made to the Federal Criteria to allow the evaluations to be carried out. The presentation will draw on the experience of the ITSEC product evaluations by UK CLEFs since 1991.

This panel will present a view of evaluation from the Certification Body and ITSEF's perspective.

**Panel
The Process**

Patricia Toth, Chair, NIST

**Carl Weber, Tandem Computers, Germany
Dr. Geof Haigh, ASK Ingres, UK
Ingo Hoffmann, SNI, Germany**

As described in the panel "European National Evaluation Schemes" the European Evaluation and Certification Scheme is based on; the ITSEC (Information Technology Security Evaluation Criteria), the ITSEM (Information Technology Security Evaluation Manual) and the European Standard set EN 45000.

This common approach defines roles of involved parties; the sponsors(manufacturers and developers), the national Certification Bodies (CBs) and the evaluators - ITSEFs (Information Technology Security Evaluation Facility).

This panel will present practical experiences with the European National Evaluation Schemes from the sponsor themselves.

They will present their experiences gained during the evaluations. They will overview practical experiences with the ITSEC and demonstrate the flexibility of the scheme and its criteria.

Tandem Computers was one of the first American manufacturers which performed an evaluation in Europe. The evaluation was based on the German "Green Books" and was performed in 1989. The evaluation of Tandem's operating system GUARDIAN/SAFEGUARD was performed by the German ITSEF "CAP debis GEI" is partially used as input for reevaluations according to the ITSEC.

A great deal of practical experience with evaluations and certifications in the field of applications (e.g. database systems) has been gained in the UK.

The American manufacturer INGRES, one of the market leaders in the field of DBMS, will provide background information on the development and evaluation of the UK certified INGRES/ES. Dr. Haigh will discuss the transition from the TCSEC evaluation to the ITSEC evaluation. He will also give his impressions of management issues from his perspective as a project leader. He will describe the motivation for seeking an ITSEC evaluation and the impact of the evaluation on the US market and future TCSEC evaluations.

Mr. Hoffmann from the German computer manufacturer SIEMENS-NIXDORF (SNI) will describe the flexibility of the European approach to evaluations. As the project leader for the SECTOS evaluation in the UK and several product evaluations in Germany, Mr. Hoffmann will present his experiences.

**Panel
International Harmonization I**

Yvon Klein, SCSSI, Chair

Panelists:

Pat Toth, NIST, USA

Julian Straw, SISL, U.K.

Klaus Keus, BSI, Germany

Wolfgang Kurth, BSI, Germany

Daniel Lovenich, BSI, Germany

Information Technology security evaluation is the crossroads of many interests which are too often presented as conflicting.

It is the only possible gateway between the user requirements and the sponsor specifications. Security should never be considered as a constraint, and the evaluation should only be the yardstick to measure how the evaluated product fulfil the claims of the sponsor, and satisfy the requirements of the user.

Technically, the security evaluation is the boundary between two different cultures, procurement and production, which have developed their own techniques and standards. Harmonization is the way to improve common understanding on this complex problem in the area of IT security.

Evaluation is the application of criteria with a well defined method on the elements resulting from the development process of the product or system.

For the development, harmonization is limited to the high level requirements on the process. Due to the variety of domain of application, it is impossible to standardize the detailed level of the process or the form of the final results.

For evaluation, the efforts of harmonization first covered the criteria to ensure the conformity of these requirements with the national or regional constraints.

To apply these different sets of criteria, evaluation methods have being developed reflecting specific national rules and regulations.

The market of IT seems to unite worldwide. This is evidenced by users requiring trans-borders application, and by the IT industry needing better return on investment by larger unified market. This generates a formal request for highest level of harmonization, passing from the national or regional approach to the international expression of needs and constraints.

Harmonization activities are in progress in the domain of criteria, by the publication of new sets of criteria as the Federal Criteria, in the domain of methods by the publication of publicly available method guideline as ITSEM, and in the domain of Quality and Security, by the publication of appropriate "Code of Practice" for Secure IT product development.

An International Comparison

The European Commission, the United States and Canada have been engaged in a Joint Task to study the Evaluation Process. The purpose of the Joint Task is to build a common understanding of the different processes followed in the evaluation of IT security products, systems and services.

The objectives of the task include:

- * To support the development of international standards for IT Security Evaluation Criteria and methods.
- * To create a framework in which to explore the concept of security evaluation criteria and supporting methodologies.
- * To explore the options for international mutual recognition of security evaluation certificates.

The Draft Federal Criteria and the ITSEC : Towards Common Criteria

The draft federal Criteria were published for comment in January this year. The contribution of the ITSEC to the development of these new criteria is acknowledged, and a stated aim is to advance the process of international criteria harmonization.

This paper draws comparisons between the ITSEC and the Draft Federal Criteria with the following specific objectives:

- a) Identification of major differences between the Criteria. These differences are addressed to determine which represent:
 - i) difference of approach or emphasis;
 - ii) potential barriers to international harmonization.
- b) Expression of the ITSEC assurance levels in terms of Federal Criteria development and evaluation assurance building blocks.

This provides a means of assessing protection profiles in terms of ITSEC assurance levels for comparison, and draws attention to detailed differences between the criteria. particular attention is paid to a comparison of E3 and LP-1.

Other areas examined include:

- a) the applicability of the criteria to both systems and products;
- b) a comparison of "security target" and "protection profile";
- c) a comparison of the range of products which can be evaluated under the two sets of criteria, with comments on the approval process for protection profiles;
- d) identification of parallels in Federal Criteria with the ITSEC concepts of Effectiveness;
- e) the issue of dependencies between functionality and assurance.

Quality Assurance in the ITSEC Evaluation Environment in Germany

The field of IT-Security is influenced by far-reaching requirements concerning quality aspects.

These requirements are implemented according to several standardized criteria and instruction which are responsible for different aspects and phases during the life-cycle of an IT-product.

Fundamental Basic Criteria as the ISO Standard set 9000 (quality assurance in design/development and production), the EN/DIN 45001 (installation and behavior of an Evaluation Facility (ITSEF)) or EN/DIN 45011 (installation and behavior of a certification Body (CB)) build the basis for a quality oriented evaluation environment.

The IT-Security specific criteria as the ITSEC and the ITSEM define the specific requirements for the development, the evaluation and certification of IT-Security products and systems.

In this paper we present a brief survey of the dependencies and relations between the Fundamental Criteria and the IT-Security Specific Criteria with respect to evaluation and certification of IT-products including IT-Security requirements.

PANEL

INTERNATIONAL HARMONIZATION II -- GOALS AND PROGRESS TOWARD THE COMMON CRITERIA

Chair: E. Troy, NIST

Panelists:

P. Cormier, CSE Canada
C. Ketley, CESG UK
Y. Klein, SCSSI France
H. Kreutz, GISA Germany
M. Tinto, NSA

The statement attached below describes a new international project to develop a common IT security criteria that will be an alignment of existing national criteria. This project was officially announced by NIST and NSA officials during the Federal Criteria Invitational Workshop, held at Turf Vally on June 2-3, 1993. The project is a joint activity of the governments of the U.S., Canada, and European nations. Six government IT security officials from these nations have formed the Common Criteria Editorial Board (CCEB), and have begun to carry out the effort.

This panel will provide the six members of the CCEB a forum to discuss the project from their own perspectives. They will describe the nature of the work, the input documents, the timetable, the intended product, and the way the draft will be reviewed by the public and subjected to trial use prior to adoption.

NORTH AMERICA AND EUROPE AGREE TO DEVELOP COMMON CRITERIA

SUMMARY (June 2, 1993)

North America and Europe have agreed to develop a "Common Information Technology Security Criteria" (CC).

Security criteria are needed to develop trusted information technology (IT) products that can be used to help protect important information of the government and private sectors. IT security criteria common to Europe and North America will help broaden the market for these products and further lead to economies of scale. In addition, common criteria will help achieve the goal of mutual recognition by North American and European nations of IT product security evaluations.

The effort, which is expected to begin in early Fall of 1993 and be completed in the Spring of 1994, will use the ISO Subcommittee 27, Working Group 3 draft criteria documents (Parts 1-3) as an

initial framework. Specific inputs will include the Information Technology Security Evaluation Criteria (ITSEC), the Canadian Trusted Computer Product Evaluation Criteria (CTCPEC), the draft Federal Criteria for Information Technology Security (FC), the experience gained to date with the ITSEC in the form of suggested improvements, the comments now being received on the draft FC document, and the results of the FC invitational workshop planned for 2-3 June 1993.

The resulting common criteria are expected to undergo extensive international review and testing by performing evaluations of "real" products against the criteria prior to being fully accepted for use within Europe and North America. When mature enough, the CC will be provided as a contribution towards an international standard to ISO Subcommittee 27, Working Group 3.

BACKGROUND

The agreement grew out of a 4 February CEC-sponsored workshop in Brussels on the Federal Criteria that was attended by many European security professionals. The general European response to the workshop was that alignment of criteria between Europe and North America is now both feasible and opportune.

This idea was taken up and endorsed by the EC Senior Officials Group for the Security of Information Systems (SOG-IS) in their meeting on 11 February, clearing the way for EC participation in the work required to achieve common IT security criteria.

As a result of informal meetings held thereafter, a proposal was made to proceed with a joint project to develop common criteria. This proposal was then given preliminary approval by EC member nations and North American government senior officials.

PLANNED DEVELOPMENT PROCEDURE -- THE EDITORIAL BOARD

Current plans call for the establishment of a six member Editorial Board (EB) consisting of three members from North America and three from Europe. The EB will be composed of senior IT security experts who have had experience designing IT security criteria and have the authority and autonomy to make decisions with regard to the contents of the CC. The EB will be requested to complete their work within a six month timeframe. The main tasks of the EB are to obtain a clear understanding of the similarities and differences between current criteria and to develop a first-draft CC for presentation to the participating government bodies. The EB will be instructed to use the material identified above as the primary material from which to develop the CC. The CC is to represent a synthesis of the best concepts

and components contained in the original material. The EB is to avoid inventing new criteria.

TECHNICAL GROUPS TO PROVIDE SUPPORT

The EB may establish and utilize special Technical Groups (TGs), as needed, to help develop specific technical areas of the CC. These TGs will operate under the direction of the EB for the time needed to perform their assigned tasks. They will be staffed in a representative way, in a pattern like that of the EB.

PUBLIC REVIEW AND TRIAL USE

Following completion of the first draft criteria, the governments involved will jointly review the CC. When they mutually determine that the CC is ready for further review by the IT security community at large, they will initiate an extensive review cycle to obtain comments from all interested parties. This cycle is expected to result in additional versions until convergence is achieved. The CC will then enter a trial period to allow the specification and evaluation of vendor offerings against the CC. Upon completion of the trial period, the CC will be revised if necessary to gain final adoption by the participating governments.

RELATIONSHIP TO ISO INTERNATIONAL STANDARDIZATION

During the process of CC development and trial use, the associated governments will work through their respective national standards bodies to help keep the ISO draft standard in relative synchronization with the CC. An issue requiring further study and consultation is how to maintain the necessary level of momentum in ISO, yet avoid finalization of an International Standard prior to achieving generally acceptable common criteria for Europe and North America.

FEDERAL CRITERIA USER FORUM A PANEL DISCUSSION

PANEL CHAIR: Caralyn A. Wichers NSA

PANEL MEMBERS:

Peter Callaway, IBM Corporation
Noelle McAuliffe, Trusted Information Systems
Charles Menk III, NSA
Marvin Schaefer, CTA Inc.
W. Olin Sibert, Oxford Systems Inc.

The purpose of this panel is to allow information system security users (e.g. vendors, evaluators, and integrators) to discuss their opinions of the Federal Criteria (FC) effort. The draft Federal Criteria for Information Technology Security identified the following four principle objectives:

- a) Develop an extensible and flexible framework for defining new requirements for Information Technology (IT) product security.
- b) Enhance existing IT product security development and evaluation criteria.
- c) Facilitate international harmonization of IT product security development and evaluation criteria
- d) Preserve the fundamental principles of IT product security.

North America and Europe have agreed to develop a "Common Information Technology Security Criteria". An information technology security criteria common to Europe and North America will help broaden the market for products and help achieve the goal of mutual recognition by North American and European nations of product security evaluations. The US will provide an updated version of the FC based on comments received and discussions held at the invitational workshop to the Common Criteria Editorial Board. At this time the format of the updated version of the FC has not been decided.

Each panelist will be allowed an opportunity to provide opinions on the Federal Criteria draft. In addition, panelists will describe areas for improvement which can then be forwarded to the Common Information Technology Security Criteria.

Charles Menk III, NSA

From an evaluator's perspective, I would have to say that the concept of the Federal Criteria carries the potential of answering a long-standing desire for a concise evaluation criteria that is capable of addressing most of the aspects associated with Trusted Computer System Evaluations. However, in its current form, the Federal Criteria falls short of our expectations and goals.

The Trusted Computer System Evaluation Criteria (TCSEC), although the most appropriate

criteria of its time, has been overtaken by the technical advances of the 20th century. This is evidenced in the development of the Computer Security Subsystem Interpretation, Trusted Network Interpretation, Trusted Database Interpretation, and the Compartmented Mode Workstation Evaluation Criteria and the continuing issuance of Formal Interpretations to address the real demands of our customers our dynamic INFOSEC environment.

The FC does address specific aspects associated with enforcing data security, where the TCSEC has traditionally failed. Specifically, the Federal Criteria concept is flexible and has the potential for being much more concise than the TCSEC. We are more knowledgeable about Computer security today than we were in 1985 and the Federal Criteria has provided a vehicle through which we can construct an evaluation criteria design to meet the needs of the 21st century.

W. Olin Sibert, Oxford Systems Inc.

In over a decade of doing evaluations against the TCSEC, much has been learned about how its requirements match up with the features and assurances of commercial products. Although on the whole, the TCSEC has stood up well, many difficulties have been encountered: ambiguous requirements; areas where the required level of vendor effort is out of proportion to a targeted security level; important security properties not covered by requirements; requirements that preclude otherwise reasonable and secure implementation options; and requirements that are not appropriate to types of products that have evolved since the TCSEC was written. These are all undesirable from an evaluation viewpoint, since evaluators don't want to make unreasonable demands or unnecessary efforts, and equally bad for product developers, who don't want to waste effort, either. The Federal Criteria presents an opportunity to address these problems. I will discuss how well it does so: areas where it is successful as well as those where further improvement is needed.

Peter Callaway, IBM Corporation

This abstract is based on IBM's critique of the first draft of the Federal Criteria, published in December 1992 and reviewed in March 1993. These views may change as a result of the June workshop, follow-on drafts, or drafts from the international harmonization effort between North America and the European Community.

With regard to furthering the state of the art of security evaluation criteria and related processes, the FC represents a significant step forward although many practical problems are introduced that need to be addressed. Examples are creation of Protection Profiles; management and ownership of a registry of Protection Profiles (to avoid proliferation and market complexity) and complexity of evaluation and lack of exploitation of recent Trusted Product Evaluation Program (TPEP) and RAting Maintenance Phase (RAMP) improvements.

Furthermore it is not clear how the universally stated goal of harmonization has been facilitated since the relationships between FC concepts, structure and components and those of the ITSEC were left to the readers to work out for themselves. For example, the FC breaks down assurance criteria into developmental and evaluation, whereas the ITSEC divides assurance criteria between correctness and effectiveness. No attempt to map one scheme to the other was offered.

Noelle McAuliffe, TIS

From my experience as a vendor of Information Technology (IT) products, as well as a consultant to other IT vendors, I found the promise of the Federal Criteria a welcome announcement. The Orange Book is an excellent foundation; however, innovative advances in technology and lessons learned during the past decade highlight the need for a more extensible and flexible criteria.

In theory, the concept of Protection Profiles, introduced in the FC, will support this objective. Unfortunately, I must say "in theory" because the success of this concept will be determined in part by the quality of the requirements that comprise the profiles. The requirements need to be concise, understandable, usable, and most importantly technically proven and sound. Their rationale should be clear. Furthermore, the requirements should focus on describing what the vendor needs to demonstrate or accomplish rather than dictating how that should be achieved. Deviations from these characteristics will result in substantial delays in the development and evaluation stages of a product's lifecycle.

The initial set of requirements presented within the FC needs further work so that they may meet these characteristics. It is my view that initially these requirements should closely reflect those included in the Orange Book, deviating only in cases where community consensus has previously been obtained. The Protection Profile concept can then be used to extend the set of requirements to include those representing new technology and lessons learned. This assumes that the vetting of a profile will, among other responsibilities, ensure that the new requirements embody the characteristics described above.

The goal of establishing "an extensible and flexible framework for defining new requirements" is a critical need. Once achieved, this improvement will certainly serve to increase the availability and use of trusted computer systems. Protection Profiles have the potential for meeting this need, although their success is highly dependent on the quality of the requirements they integrate.

Marvin Schaefer, CTA INCORPORATED

It is the lot of the system security integrator to build a trusted *system* or to assess the security properties of composed *systems* that are, in turn, built from products and systems. The 'or' above is *not* an exclusive or: analysis and assurance are integral parts of the job. Analysis includes taking risk and threat into account; assurance entails determining that the composed system, used consistent with its specified procedural and environmental constraints, meets the requirements of its users. Systems are intrinsically different from products, since they are tied to a known application rather than to a generic range of applications.

The TCSEC was originally intended to address system contexts, both in terms of acquisition specification and in terms of certification support. However, from the time of its earliest application to system contexts, it was necessary to interpret the words of the TCSEC to fit the context of the system. In many cases, this application was not immediate or straightforward: reasoning, and sometimes heated discussion, were required to resolve those cases where the fit was not perfect.

Well, however difficult those exercises may have been, the experiences of producing *generic* interpretations were far more difficult. This is because the generic interpretation needs to be based on invariant principles rather than on particular attributes of the context of use or of system composition. While it is easy to find fault with the TCSEC in the light of a decade of use, it has been more difficult to precisely identify and characterize the deficiencies of the TNI and the TDI, as the problems in the latter tend to be far more subtle in their implications. None the less, certain *system* analyses by integrators have identified cases in which strict adherence to the requirements of the TNI has produced compliant, but fundamentally flawed (i.e., exploitable), security architectures.

Well, users have become more sophisticated and have far more sophisticated needs now than they did a decade ago. While there has always been an identified need for any system to process data efficiently to provide assured service, and to provide various forms of integrity, even these assured features are no longer adequate to satisfy the needs of the user community. This is because the rapid advances in technology, with resultant support for graphical interfaces, object oriented multimedia support, hybrid distributed processing complexes, etc., have become assimilated technologies, and users are unwilling to abandon these essential features just so that they can protect data from unauthorized release. If there is any doubt about this, please consult any recent RFP for a major procurement: a year ago, RFPs began demanding secure GUIs -- six months earlier, the term was not in wide use. 'GUI' is not listed in the glossaries of the TCSEC or of the Rainbow Series.

Despite the use of a lot of the above modern application vocabulary, the draft Federal Criteria, the ITSEC and the CTCPEC do not deliver closed-form solutions for addressing these requirements on single platforms or in distributed systems contexts. There is no guidance available to tell the integrator how to assess the specific INFOSEC properties (i.e., the composed security policy) of a system that interconnects two evaluated B2 components produced by different vendors. To my taste, the book that comes closest is the CTCPEC -- but not even it meets my daily integration needs.

The fundamental difficulty is not one that could be corrected by getting better criteria writers. Research in INFOSEC has not kept pace with the rapid advances in information technology. Until the research has been done, producing new criteria will not resolve any of the fundamental problems. It would be naive to assert that these fundamental problems could be resolved by producing a criteria that readily admits to the addition of new protection profiles derived from bundling together a set of feature and assurance components.

However, while the Federal Criteria could not possibly, without periodic revision, achieve its stated goals for addressing the INFOSEC requirements of systems of the present, it is fair to ask whether it adequately addresses the needs for assessing or integrating products as components. I and some colleagues in other companies have conducted a simple set of thought experiments to see if the Federal Criteria is an improvement of the TCSEC. We conclude that without major rethinking and reworking it is not. On the one hand, we find that the so-called vetted profiles do not establish equivalence between the Division B and Division A products and the LP counterparts: evaluated B3 and A1 products have been confirmed to fail some technical LP-2 requirements; worse, penetration scenarios have already been found to demonstrate a lack of security in

LP-3 and LP-4-compliant architectures!

I certainly support the need to improve on the existing TCSEC. However, the task is subtle and fraught with potential difficulties. Unfortunately, the voluminous tomes comprising the December 1992 draft do not meet this challenge.

NCSC/NIST National Computer Security Conference

Tutorial Series On Trusted Systems

Presented by:

R. Kenneth Bauer
Joel Sachs
Dr. Eugene Schultz
Dr. Gary Smith
Dr. William Wilson
ARCA
 8229 Boone Blvd.
 Vienna, VA 22182
 703-734-5611

Dr. Charles Abzug
LtCdr Alan Liddle, Royal Navy
Information Resources Management College
National Defense University
 Fort Lesley J. McNair
 Washington, D.C. 20319
 202-287-9321

Schedule

	Monday - 20 Sept 1993	Tuesday - 21 Sept 1993	Wenesday - 22 Sept 1993
0900 - 1030		Trusted Networks <i>R. Kenneth Bauer</i> ARCA	Trusted Networks <i>Dr. Eugene Schultz</i> ARCA
1100 - 1230		Trusted Databases <i>Dr. Gary Smith</i> ARCA	Trusted Databases <i>Dr. William Wilson</i> ARCA
1400 - 1530	Threats & Security Overview <i>LtCdr Alan Liddle</i> IRMC, NDU		
1600 - 1730	Trust System Concepts <i>Dr. Charles Abzug</i> IRMC, NDU	Trusted Integration & System Certification <i>Joel Sachs</i> ARCA	

Description

These tutorials are based on courses and seminars given by either ARCA or the Information Resources Management College of the National Defense University. Arca's Information Security Seminars focus on several topics, including Computer, Network, and Database Security and incorporate experience from applying Arca's security engineering and consulting services on MLS systems solution developments. The Information Resources Management College includes security in its information management courses, particularly through an intensive Automated Information Systems Security Course which is taught at the graduate level.

The tutorials will be presented in lecture format with question and answer periods. While there is a logical flow between the tutorials, each tutorial will be presented as a separate unit so that conference attendees can attend any or all of them. The tutorials are intended to introduce many and varied security topics as opposed to exploring them in-depth. Brief descriptions of each tutorial identified above follows:

Trusted Systems Concepts focuses on the fundamental concepts and terminology of trust technology. It includes descriptions of the Trusted Computer System Evaluation Criteria [TCSEC] classes, how the these classes differ, and how to determine the appropriate class for your operational environment.

Threats & Security Overview focuses on the general threats to automated information systems, assets requiring protection, and an overview of security disciplines (operational, communications, computer, physical, and administrative) as well as their interrelation.

Trusted Networks focuses on basic points in network security and gives an overview of the TNI. Topics include network security concerns and services, trusted network components, the TNI and its Evaluation Classes, system composition and interconnection, and cascading.

Trusted Database Systems focuses on security from a "database view" and gives an overview of the TDI. Topics include DBMS specific security requirements, vulnerabilities, and challenges; database design considerations; implementation issues; and use issues.

Trusted Integration & System Certification focuses on issues in integrating MLS solutions using trusted products, the development of the certification evidence, and the accreditation process. Topics include system-wide security and assurance, security trade-offs, and development methodologies.

A TUTORIAL ON COMPUTER VIRUS PREVENTION

by Aryeh Goretsky

Manager, Technical Support
McAfee Associates, Inc.
3350 Scott Blvd. Bldg 14
Santa Clara, California
408-988-3832

With damage from computer viruses becoming a greater risk into the 1990's and beyond, countermeasures must be taken to prevent, or at least reduce the risk, of infection and damage on the IBM PC and compatible platforms. This presentation will be broken down into five sections: what a computer virus is; preventing computer viruses; isolating and identifying computer viruses; recovering from an infection; and evaluating the damage done by the virus and what changes are necessary to prevent it from happening again.

In the first section we will examine what computer viruses are and what areas of the computer are vulnerable to them. We will also look at the tell-tale symptoms of virus infection, and discuss the types of damage they can do.

In the second section, we will look at steps that can be implemented to greatly reduce the risk of computer virus infection.

In the third section, we will learn how to check for a suspect computer virus using DOS commands, utility programs, and anti-viral software.

In the fourth section we will discuss recovery procedures for a virus-infected system.

In the fifth section we will evaluate the damage done by the computer virus and look at ways to prevent a similar incident from happening again.

Lecture time will be approximately 75 minutes followed by 15 minutes of questions. Handouts will be provided.

Panel Summary

Getting Your Work Published

Jack Holleran, *Chairman*
National Security Agency

Speakers

Dr. Harold Joseph Highland, FICS
Editor-in-Chief Emeritus, Computers & Security
Distinguished Professor Emeritus, State University of New York

Professor Sushil Jajodia, *George Mason University*
Editor, Computer Security Journal

Mr. Donn Parker, *SRI International*
Author and Consulting Editor,
Journal of Information Systems Security I-4

Dr. Charles Pfleeger, *Trusted Information Systems (UK) Ltd.*
Author

Ms. Deborah Russell, *O'Reilly & Associates, Inc.*
Acquisition Editor & Author

One of the features of the National Computer Security Conference is "peers discussing problem-solving techniques with peers." This panel discusses the insights that a prospective author might need to succeed in publishing his or her work.

Mr. Parker will discuss quality, integrity and growth beyond today's folk art articles and papers. Professor Jajodia will discuss the types of papers that the new Computer Security Journal is reviewing for acceptance. Dr. Pfleeger and Ms. Russell will discuss book publishing: Dr. Pfleeger from the author's perspective (i.e., how the author chooses a publisher); Ms. Russell from the publisher's perspective (i.e., how a publisher decides what to publish. Dr. Highland will provide information about how to submit an article to a professional, refereed journal and some insights into selecting a book publisher and submitting a manuscript for consideration.

Each of the panelists has contributed to or authored books in the Information Security community. All of the panelists have reviewed books or articles for publication.

After the discussion, the panel will address questions from the conference attendees.

EXECUTIVE SUMMARY

**PANEL:
INFORMATION SYSTEMS SECURITY STANDARDS
THE DISA PROCESS**

Chair: Mr. Bill Smith, CISSP
Defense Information Systems Agency
Joint Interoperability & Engineering Organization

Panelists:

Mr. John Keane, DISA/JIEO
Mr. Craig Sutherland, DISA/JIEO
Dr. Dale Nunley, NSA
Ms. Marilyn Kraus, DISA/JIEO

Abstract. The purpose of this panel is to report on the Defense Information Systems Agency's (DISA) Information Technology (IT) Standards Program. Major topics will include discussions from senior leaders on the DoD "Technical Architecture Framework for Information Management" (TAFIM); the "Defense-Wide Information Systems Security Program (DISSP) Goal Security Architecture"; "Information Systems Security (INFOSEC) Standards Activities at the National Security Agency (NSA)" and the "DoD Information Technology Standards Management Program".

Background. Significant events have occurred within the past year in the way the DoD implements its Information Technology (IT) Standardization Program. Changes have occurred throughout all echelons and some responsibilities have been redefined. The Defense Information Systems Agency (DISA) was designated in September 1991 as the DoD Executive Agent for coordinating and integrating the DoD's IT Standards Program in support of the Assistant Secretary of Defense for Command, Control, Communications and Intelligence (ASD(C3I)). Panelists represent major DISA elements involved in INFOSEC Standardization: the Center For Information Systems Security (CISS) [manages the Defense-Wide Information Systems Security Program (DISSP)]; the Center For Technical Architecture (CTA) [manages the Technical Architecture Framework for Information Management-TAFIM]; and the Center For Standards (CFS) [DISAs Lead Standardization Activity for selected IT standards areas]. These Centers of Excellence are elements of DISAs Joint Interoperability and Engineering Organization (JIEO). The Centers work closely among themselves and with the National Security Agency's (NSA) Information Systems Security Organization to coordinate on-going and emerging INFOSEC standards development activities as well as national INFOSEC policy and guidance.

This panel session will discuss the activities of DISA in its DoD Information Technology (IT) role focused on Information Systems Security (INFOSEC) Standardization and provide an overview of INFOSEC standardization activities at NSA. A Roundtable Discussion will conclude this panel with audience participation.

Panel Summary

Technical Architecture Framework for Information Management.

This discussion provides an overview of the DoD Technical Architecture Framework for Information Management (TAFIM). The TAFIM provides the integrated guidance that governs the evolution of the DoD's technical infrastructure. It does not provide a specific system architecture. Rather, it provides the services, standards, design concepts, components and configurations which can be used to guide the development of technical architectures that meet specific mission requirements. Security services and standards are integral to the TAFIM, as highlighted through the discussions of the other panelists. The TAFIM provides a total technical documentation structure for designing and implementing efficient and effective information systems that support improved functional processes. Proper application of the Technical Architecture Framework can: 1) ensure integration, interoperability, modularity, and flexibility; 2) guide acquisition and reuse; and 3) speed delivery and lower information technology costs.

The Role of Standards in the DISSP Goal Security Architecture.

The DISSP Goal Security Architecture (DGSA) is a generic security architecture from which mission-specific architectures can be derived. This security architecture has been coordinated with the developers of the TAFIM and will become an integral part of the generic DoD information systems architecture. There are four key operational requirements driving the DGSA: open distributed processing systems, support for multiple security policies, appropriate security protection, and common security management. Security standards are critical to the realization of the objectives of the DGSA.

To date, most security solutions have relied upon an add-on component approach, which inhibits system interoperability, application portability and is costly. The DGSA relies upon and promotes the concept that security is an integral feature of information systems. This concept means that security is a constant consideration in the information system development cycle and is dependent on security mechanisms as embedded technology. As a result, security standards are critical to achieving interoperable, portable and cost-effective solutions in two ways. First, information preparation and exchange applications and protocols need to be modified so that their standards include security services as part of their implementation. Second, security-specific mechanisms, protocols and supporting functions will have to be developed as standards if operational requirements and goals are to be met.

The development of standards is a formal and lengthy process, but it pays many dividends. To affect standards development and reap the attendant benefits, the DGSA developmental team must play an active role in various standards organizations. The team must promote the development of security standards and present them to the standards organizations. Further, the team must devote resources to the implementation and use of these standards, ranging from proof-of-concept experimentation to specification of operational solutions.

Some NSA Activities in the INFOSEC Standards Process.

This discussion describes some of the National Security Agency's (NSA) responsibilities and activities in the area of Information Systems Security (INFOSEC) standards, emphasizing support to the joint DISA/NSA Defense Information System Security Program (DISSP). It describes NSA involvement directed toward fulfillment of the responsibility to provide INFOSEC standards in support of the DISSP Goal Security Architecture and DoD Information System (DIS) programs. Also discussed is the application of NSA information systems security experience, knowledge, and leadership in support of the DISA/JIEO Center For Standards (CFS) standards process. Security labeling is an example of one area in which this support is provided. The status of the effort to develop a common security label acceptable to both industry and government is discussed. Details of the security labeling work were first presented at the 15th National Computer Security Conference.

DISA's Information Technology Standards Management Program.

This discussion briefly describes the DISA/JIEO Information Technology (IT) Standards Program as the mechanism to lead, manage, integrate, and coordinate efforts centrally to achieve and implement Information Technology Standards in DoD Information Systems. The role of the Center For Standards (CFS) in the Defense Standardization Program (DSP) will be discussed. Central to CFS INFOSEC standards management is the coordination with other JIEO Centers, NSA, the National Institute of Standards and Technology (NIST), and recognized international, national and federal standards bodies and committees. It describes CFS involvement directed toward fulfillment of the responsibility to support the DISSP Goal Security Architecture, the DoD Technical Architecture Framework For Information Management, and Information Technology (IT) programs through the specification, adoption, development, certification, and enforcement of IT standards for DoD use.

Roundtable Discussion.

At the conclusion of the individual panelist presentations, a "Roundtable Discussion" period will be held to answer questions from the audience. The chair will moderate this period. In view of the anticipated audience interest in the topics discussed, the chair will initially solicit written comments from the audience to guide this discussion. The audience is encouraged to interact in this discussion period and provide feedback to the panelists on the topics presented.

Security Requirements for Cryptographic Modules

Chair: Lisa Carnahan, NIST

Panelists: Miles Smid, NIST
David Balenson, Trusted Information Systems, Inc.
Roberta Medlock, Mitre Corp.
Patricia Edfors, Dept. of Justice
Don Thompson, Jones Futurex, Inc.

Cryptography can be used to provide integrity and confidentiality for information processed and stored by automated electronic applications. Cryptography can also be used for the generation and verification of electronic signatures in these systems. For application developers who have determined a need to utilize cryptography to protect electronic information, the next step is a perplexing one. How does a developer or implementor select a cryptographic module? What requirements should be met to provide proper security features for a module implementing security? Federal Information Processing Standard (FIPS) 140-1, Security Requirements for Cryptographic Modules addresses this issue. A cryptographic module which is designed to meet FIPS 140-1 incorporates cryptographic algorithms and functions specified in related FIPS. In addition, this standard specifies four security levels to provide for a wide spectrum of data sensitivity and a diversity of application environments. Each security level offers an increase in security over the preceding level. These four increasing levels of security will allow cost-effective solutions that are appropriate for different degrees of data sensitivity and different application environments. Table 1 provides an overview of the requirements specified in FIPS 140-1. This standard is expected to be the foundation for NIST's current and future cryptographic standards.

The purpose of this panel is to familiarize the audience with the specifics of the standard and the issues surrounding it. Panel members will discuss the development of the standard, the use of the standard in real world applications, as well as the FIPS 140-1 validation program. The panel may be of interest to parties in both the private and public sectors. This includes project managers, application developers, and integrators in federal agencies and industry who are incorporating cryptographic security into applications.

	Security Level 1	Security Level 2	Security Level 3	Security Level 4
Crypto Module	Specification of cryptographic module and cryptographic boundary. Description of cryptographic module including all hardware, software, and firmware components. Statement of module security policy.			
Module Interfaces	Required and optional interfaces. Specification of all interfaces and of all internal data paths.		Data ports for critical security parameters physically separated from other data ports.	
Roles & Services	Separation of required and optional roles and services.	Role-based operator authentication.	Identity-based operator authentication.	
Finite State Machine	Specification of finite state machine model. Required states and optional states. State transition diagram and specification of state transitions.			
Physical Security	Production grade equipment.	Locks or tamper evidence.	Tamper detection and response for covers and doors.	Tamper detection and response envelope.
EFP/EFT	No requirements.			Temperature and voltage.
Software Security	Specification of software design. Relate software to finite state machine model.		High-level language implementation.	Formal model. Pre- and post- conditions.
Operating System Security	Executable code. Authenticated. Single user, single process.	Controlled access protection (C2 or equiv).	Labelled protection (B1 or equiv). Trusted communications path.	Structured protection (B2 or equiv).
Key Management	FIPS approved generation/distribution techniques.		Entry/exit of keys in encrypted form or direct entry/exit with split knowledge procedures.	
Crypto Algorithms	FIPS approved cryptographic algorithms for protecting unclassified information.			
EMI/EMC	FCC Part 15, Subpart J, Class A (Business use).		FCC Part 15, Subpart J, Class B (Home use).	
Self-Tests	Power-up tests and conditional tests.			

Table 1: Summary of security requirements