Advancing Software Security– The Software Protection Initiative

Mr. Jeff Hughes
AT-SPI Technology Office
AFRL/SN
2241 Avionics Circle
WPAFB, OH 45433-7320
Jeff.Hughes@wpafb.af.mil

Dr. Martin R. Stytz, Ph.D.
AT-SPI Technology Office
AFRL/SN
2241 Avionics Circle
WPAFB, OH 45433-7320
Martin.stytz@wpafb.af.mil

Abstract

In December 2001, the Software Protection Initiative (SPI) was established to prevent the unauthorized distribution and exploitation of national security application software by our adversaries. To achieve this, the SPI has several goals, which are to institutionalize software protection within the software development life-cycle, educate and train the community, develop user-friendly protection techniques, and ensure that protection technology and policy are appropriately applied, balancing mission requirements with security. The focus of the SPI is to improve protections for critical scientific, engineering, and modeling and simulation software running on desktops through supercomputers. Not only does software of this nature represent a significant portion of DoD's intellectual property (IP), it also enables the development of next generation weapon systems.

In addition to the traditional two components of information assurance, namely network security and operating system integrity, SPI adds a new component, a "third leg" to the information assurance triad, based on an application-centric approach to protecting important DoD software. The SPI program is accomplishing these goals by providing military-strength application protection, focused investigation, research and development of advanced technologies for software protection across the entire spectrum of computational hardware.

1 Introduction

In this paper, we will outline existing Information Assurance policy, the motivation for Software Protection from the perspective of DoD, and describe how the Software Protection Initiative (SPI) dovetails with current policy. Additionally, we will describe in greater detail the charter of SPI, and how the SPI program is currently constituted to achieve those goals. Finally, a glimpse into the current research activities undertaken by the SPI program will provide insight into the types of technical activities required to successfully achieve the protection of DoD application codes.

2 Information Assurance

The Department of Defense (DoD) and other Federal agencies developed policy, procedures, and tools to safeguard national security information, as well as the systems on which this data resided.

Report Documentation Page

Form Approved
OMB No. 0704-0188

Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

1. REPORT DATE **JUN 2003**	2. REPORT TYPE	3. DATES COVERED **00-00-2003 to 00-00-2003**
4. TITLE AND SUBTITLE **Advancing Software Security - The Software Protection Initiative**		5a. CONTRACT NUMBER 5b. GRANT NUMBER 5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)		5d. PROJECT NUMBER 5e. TASK NUMBER 5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Air Force Research Laboratory,AFRL/SN,2241 Avionics Circle,Wright Patterson AFB,OH,45433-7320**		8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)		10. SPONSOR/MONITOR’S ACRONYM(S) 11. SPONSOR/MONITOR’S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

The original document contains color images.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**		**46**	

Department of Defense Directive (DODD) 8500.1 defines *information assurance* as follows:

> "Information operations that protect and defend information and information systems by ensuring their availability, integrity, authentication, confidentiality, and non-repudiation. This includes providing for restoration of information systems by incorporating protection, detection, and reaction capabilities."

This definition is broad - broader than the common concepts of *information security* and *computer security*. From the Government's perspective, the definition given above *must* accommodate additional activities required to fully address the threat at the national security level. Software application security (Software Protection) will become another tool in the Information Assurance toolbox that can be used to *protect and defend* our information resources on multiple fronts.

The above definition establishes five criteria that can be used to gauge the level of protection provided by an information assurance activity. These five criteria are:

> **Availability** Timely, reliable access to data and information services for authorized users.
>
> **Authentication** Security measure designed to establish the validity of a transmission, message, or originator, or a means of verifying an individual's authorization to receive specific categories of information.
>
> **Confidentiality** Assurance that information is not disclosed to unauthorized entities or processes.
>
> **Integrity** Quality of an information system reflecting the logical correctness and reliability of the operating system; the logical completeness of the hardware and software implementing the protection mechanisms; and the consistency of the data structures and occurrence of the stored data. Note that, in a formal security mode, integrity is interpreted more narrowly to mean protection against unauthorized modification or destruction of information.
>
> **Non-repudiation** Assurance the sender of data is provided with proof of delivery and the recipient is provided with proof of the sender's identity, so neither can later deny having processed the data.

It is clear from these definitions that traditional Information Assurance focuses on the host operating system, and the security of the data stored on that host. Little emphasis is provided on the highly valued applications that are used to generate or manipulate the data. In the next section, we will describe the importance of protecting these applications from theft, misuse and tampering.

3 Motivation For Software Protection

In the past, the ability to effectively utilize high performance computing (HPC) assets depended upon the availability of high performance computer systems and application software that executed on these systems. The development of high performance hardware and state-of-the-art software technologies enabled advances in defense science and technology and provided the computational capability that is a key component of maintaining US national security. Because of the perceived national value of HPC assets, these assets were protected almost from their very inception. The protection paradigm for HPC codes relied on export control of HPC hardware. The protection provided to the software itself was relatively minimal when compared to the current SPI goals. Traditionally, the strategy employed to maintain the US lead in the HPC arena has consisted of two parts: 1) research and development to ensure the maintenance of superior technology and 2) host compute platform export controls to restrict access to these assets by potential adversaries.

Recently, rapid advances in computer hardware technology have undermined the effectiveness of export controls on high performance computer systems. At this time, there is no reason to believe that this trend is reversible and there are many reasons to believe that it will instead worsen with time. Additionally, there has been an increasing realization that the software, including traditional HPC codes, contains a crucial intellectual DoD property that must be protected. The wide spread availability of powerful computing technology, in conjunction with the realization that DoD software has tremendous intellectual property value, mandates stronger protection of application software. This, coupled with the fact that unprotected software is significantly easier to misappropriate (i.e. copy) than is the theft of the HPC host platforms, along with the ease of creating clustered systems using non-export controlled nodes, forces the DoD to seek ways of providing application-centric security to protect the massive intellectual property.

Another motivation for protecting software springs from the significant investment in software that has been made over the past three decades. This investment has permitted the DoD to gain a significant technological advantage over our adversaries. Protection of these critical applications is essential to maintaining that advantage. Some examples of critical application software that are credited for helping to achieve this technological advantage include electromagnetic modeling software for radar signature predictions such as those within the EMCC, signal and image processing software for sonar and radar, fluid dynamics software for aircraft testing, and software for armor and projectile design. The list of critical national security application includes numerous scientific, engineering and modeling and simulation codes that are crucial to military activity, the DoD must act to protect its software technology from unauthorized use, theft, reverse-engineering, or other exploitation.

4 Software Protection Initiative

As a result of these disturbing trends, the President issued a directive to "identify and invest in additional measures for the protection of critical national security software codes." In December 2001, the Under Secretary of Defense for Acquisition, Technology, and Logistics (USD (A&T) directed the Deputy Under Secretary of Defense for Science and Technology (DUSD (S&T) to undertake the Software Protection Initiative (SPI). The SPI has several goals: 1) Institutionalize software protection as part of the application software life-cycle, 2) Educate and train the community, 3) Develop a wide array of user-friendly protection techniques, and 4) Ensure that protection technology and policy are appropriately applied, balancing mission requirements with security. The SPI program is addressing these goals by providing military-strength application protection, focused investigation, and research and development of advanced technologies for software protection across the entire spectrum of computational hardware.

The focus of the DoD SPI is to improve protections for critical scientific, engineering, and modeling and simulation software running on desktops through supercomputers. This type of software represents a significant portion of DoD's intellectual property and enables the development of next generation weapon systems. The SPI is developing software protection technologies; supporting the insertion of these technologies into application software; defining tools and methods for protected development and distribution of application software; and providing inputs on application software related policy matters in coordination with the OUSD (P)/DTSA.

The Software Protection Initiative is envisioned as the third leg of the information assurance triad. It complements existing information assurance efforts in network security and operation system access controls with an application-centric approach to protecting critical DoD intellectual property. The SPI is about developing practical ways - software, hardware, and/or procedural - to protect high value software. SPI is not simply another approach to improve network security, operating system access control, or software for embedded or weapon systems, but instead to protect the integrity of the software applications themselves.

The primary thresholds for defining practical security are: 1) When the attack requires more effort than independently producing an equivalent software product 2) When the skills, time, or computer hardware needed to successfully attack software are out of reach for the adversary 3) When the skills required provide enough of a deterrent to discourage attempts.

There are currently a number of techniques to afford an application the requisite protection. This protection is carefully matched against the protection needs as defined by the DoD agency in charge of the given application. For example, some applications need to be principally protected from theft, other applications require a limit on the duration or number of cases that can be computed, while still others require a variety of different performance levels, given the trust level of a given user. The SPI program balances these protection requirements with mission requirements, by careful selection of protection techniques, resulting in a unique blend of security "layers" commensurate with the specific application. All this must be maintained, while having minimal impact on computational performance, as well as scaling effectively from the single processor desktop system to the very high node count of modern and future cluster and mesh technologies.

5 SPI Research

The software development community needs additional technologies and tools to protect software against piracy, to ensure code integrity, and to prevent unauthorized usage of critical DoD software applications. The lack of quantifiable protection technology greatly increases the risk of unauthorized exploitation of critical national security software by potential adversaries and threatens to erode the technological advantages we have worked to obtain. Allowing an adversary to obtain these critical applications enables the adversary to save on R&D costs, use the knowledge gained to build more advanced weapons, use the R&D savings to buy more weapons and use insight gained on U.S. weapons' vulnerabilities and tactics against U.S. Forces. To counter this threat, the SPI program addresses software protection deficiencies through the investigation, research and development of advanced technologies for software protection. SPI research is broad in scope and topical in nature in order to foster new, creative, and pioneering solutions and mature technologies.

Since the DoD's current level of focus on software protection is relatively recent, it faces both pressing, near-term research issues as well as longer-term research issues. In the near term, the research goal is the protection of legacy software (i.e. software that is already developed and in use). To support longer-term objectives, research is being funded to focus on the development of techniques and a secure development environments (SDE) that will enable the protection of software from the moment of its inception throughout its lifecycle. The SDE will protect the early stage code development, and ultimately will assist the developer in positioning his or her code for the insertion of software protection techniques.

Given these near and long-term application security research objectives, we can identify four main research thrust areas: 1) algorithms, 2) environments, 3) benchmarks and metrics, and 4) integration. The algorithms area addresses the need to develop improved algorithms and techniques for application security. The environments focus area addresses the need for implementing software development environments that allow us to develop protectable software, maintain software pedigrees, insert protection techniques into software under development, and otherwise protect software throughout its entire development process. The benchmarks and metrics focus area addresses the need to develop means for measuring the strength and potency of different application security algorithms, assessing the ability of an application security technique to protect software on a given computational platform and to measure its impact upon performance. The integration focus area addresses the need for research into the development of techniques for the efficient integration of multiple application security techniques into software and the integration of application security techniques with other information assurance techniques such as network security and operating system security to form robust layered protections.

6 Current Research Efforts

Mitigation of the Differential Analysis Threat

Due to the complexity and componentization of software applications, developers typically update applications in a modular fashion. This eases test and validation but leaves much of the final object code unchanged. As such, when new versions are released, a differential analysis of the new and old version would indicate where differences in the code exist. There is great concern that if software protection hooks were added to applications already being widely used, then a differential analysis would locate the position of those hooks in the source, object, or binary code, allowing it to be defeated. An expensive and impractical solution would be to rewrite the entire application in a different programming language before incorporating software protection techniques. Research is needed to develop technique(s) to effectively counter differential analysis.

Reconfigurable Processors for Software Protection

The SPI is concerned about attacks on critical national security applications by highly motivated, foreign-government-funded organizations with virtually unlimited resources for conducting their software attacks. Disassemblers, debuggers, and virtual execution environments may be used to reverse engineer facsimiles of application source code from processor-specific executable files. Consequently, a number of software applications are vulnerable to these organized attackers.

The most crucial element for these types of attacks is the availability of the opcode instruction set for a specific processor. To operate properly, the attacker must do one of two things: 1) utilize the instruction set of the processor that the software and operating system was designed to operate on. Only then can the attack successfully convert the binary executable into an equivalent mnemonic code for the purpose of extracting legible information. 2) They can dynamically monitor the activity of the software executable during real time operation.

If the binary code can be made unique to only a single processor, the level of difficulty for both attack methods rises considerably. For example, this technology could address a Differential Software Analysis Threat since each version of the binary software would be different from previous versions. To work towards this goal, the SPI is exploring the practicality of designing such a processor whose opcode instruction set would be programmable.

Protecting Software Binaries from Reverse Engineering

There are a number of software applications that are vulnerable to reverse engineering. The SPI is researching new and novel methods for protecting binaries and legacy applications, which are infeasible or impractical to recompile or relink. Ongoing research to develop techniques for processing software binaries to make reverse engineering more difficult include:

- Detecting hostile reverse engineering applications including debuggers and disassemblers
- Detecting falsified operating environments
- Memory and file protection
- Obfuscation, as applied to executables

One of the requirements for protecting software executables is the ability to "lock-down" a particular application to one computer system. Tools such as FlexLM, hard-disk volume IDs, and hardware dongles have been used with modest success. Recently, there has been some interest into utilizing the idiosyncratic signatures emanating from standard magnetic medias such as hard disk drives and swipe cards to explicitly link the software with specific computer platforms. The software uses the "noise signatures" of the system's hard disk drives or swipe cards to determine if it is still operating on the same computer platform. If not, the software will not run or erases itself. The end result is a "locked-down" version of the software that helps prevent illegal uses of the software application.

Tools to Aid in Protection of Application Software During Development

Another challenge for the SPI is the insider threat during development. Typically, relatively lower security is applied within development environments between development teams as opposed to the distribution process once the application is ready to ship. While currently available software development tools may enhance productivity and accelerate the code-compile-test cycle, they do not provide meaningful assistance to protect software during the development process. Of particular concern is the security of a software application when it is assembled from components, subprograms, and/or objects. This issue is especially salient as most complex codes are developed using available subprograms or objects. Object oriented software composition is increasingly viewed as a promising means for controlling development costs and for enabling the assembly of complex software applications. However, this same technique also permits one component/subprogram/object to compromise the protection of an application during development, or later during execution. It is important to develop tools that assist programmers in establishing, tracking and maintaining the version history or "pedigree" of the software under development. The pedigree documents the history of those who have had access to the source code and unprotected binary application, and records all locations where the source codes and binary executables may have resided. Software protection will use an application's pedigree to determine which protection measures would be appropriate, as well as the vulnerability to various methods of attack such as differential code analysis.

The SPI is exploring software development environments that protect the source code from unauthorized redistribution, while providing those who have authorized access a usable programming environment. Currently, software development environments are focused at programmer productivity and not on source code protection. This effort is a critical theme of the SPI program "Protecting software during the development stage while enabling use by appropriate parties."

7 Conclusion

Application software represents a significant portion of the DoD's intellectual property. It enables the development of next generation weapon systems that allows the U.S. to maintain a technological advantage over our adversaries. Access to these critical applications by our adversaries will allows them to save R&D costs, produce more lethal weapons, and analyze U.S. weapons and tactics, techniques, and procedures for vulnerabilities.

The DoD's Information Assurance (IA) posture allows for mitigation through the integration of people, technology, and operations; the layering of IA solutions within and amongst IT assets; and, the selection of IA solutions based on their relative level of robustness. The security of the entire infrastructure will depend on the security of each component, and as threats and vulnerabilities evolve, security must evolve at an equal or higher rate. In cyberspace, attackers can be anywhere. No geographic safety exists. Placing a wall around the perimeter of a network is not adequate to achieve security.

Software protection provides an additional layer of security that helps to ensure the availability of critical assets and infrastructure. By augmenting the traditional IA concerns of network and operating system, the SPI focus results in a total approach which contributes to DoD asset protection and will serve to maintain the strategic US lead in technologies critical to our nation's interests.

Software Application Security

Martin R Stytz, Ph.D.
Jeff Hughes
AFRL/SNAS
2241 Avionics Circle
WPAFB, OH 45433-7320

SPI - Mission

- Software Protection Initiative (SPI)
  - To prevent the unauthorized distribution and exploitation of national security applications by our adversaries

SPI - Motivation

- High performance computing (HPC) hardware availability (i.e., Linux clusters)
- Decades of investment in high performance software and the research results they embody
- Critical to every aspect of military activity, from training to operations
- Protected software is the foundation for high confidence computing

SPI - Vision

- Establish the Software Protection Initiative as an integral layer of the defense in depth concept for information assurance
- Complement existing information assurance efforts in network security and operating system access controls with an application-centric approach to protecting critical DoD intellectual property

SPI - Strategy

- Develop technologies that provide quantifiable protection of sensitive software
- Determine resilience of technologies to attack or subversion
- Sub-components
  - Continual assessment of technologies
  - Development of techniques and technologies
  - Development of metrics and benchmarks

Application Security Definition

What is meant by “Application Security”

- Anti-Piracy
  - Protected distribution
  - Protected execution
- Code integrity
  - Trusted execution
- Vulnerability reduction
  - Reduction of security flaws
  - Secure development environment

SPI - Scope

- What SPI is: Securing high value application software running on COTS computers.
  - Presently consists of 3 thrusts:
    - Identify and protect existing critical applications
    - Devise secure development environment for future applications
    - Educate the DoD community
- What SPI is not:
  - Network security
  - Operating system access control

SPI - Goals

- Institutionalize software protection as part of the application software life-cycle
- Educate and train the community
- Develop a wide array of user-friendly protection techniques
- Ensure that protection technology and policy are appropriately applied to protect and extend our technological advantage

SPI Activities

- Training
- Protection Technology VV&A
- Metrics Development
- Outreach & Education
- Research & Development

SPI Activity Training

- Goal
  - Train program managers, engineers, scientists, and software developers on how to protect code pedigrees and how to write protectable code
- Key Activities
  - Developing modular short course
    - Formal training courses will be held at SPC and other locations approximately six times per year

SPI Activity
Protection Technology VV&A

- Goal
  - Respond to user/developer feedback and validate usability, scalability, and maintainability of SPI technologies
- Key Activities
  - Assembled broad based VV&A support structure
    - Technology Review Panel
    - Internal Red Team, VV&A
    - External Red Team
    - Insertion Team
    - User Community

SPI Activity Metrics

- Metric Categories
  - Denial of use
  - Denial of exploitation
  - Validate “ilities”: usability, scalability, maintainability, availability
- Values considered for criteria

Cost to “us”	vs.	Cost to “them”
$$ to implement		**$$ to defeat**
Run-time impacts		**Time to defeat**
Skills needed to implement		**Skills needed to defeat**
Extra memory usage		**Size of team needed**
Numerical accuracy		
Schedule impact		
Reliability		

SPI Activity
Outreach & Education

- Goal
  - Cultivate awareness of the threat and the need for application code security
  - Promote software protection as integral to defense in depth for Information Assurance.
- Key Activities
  - Academic centers of excellence program
  - SPI is participating in conferences, providing input for publications, and establishing contacts to increase awareness of the need for software protection

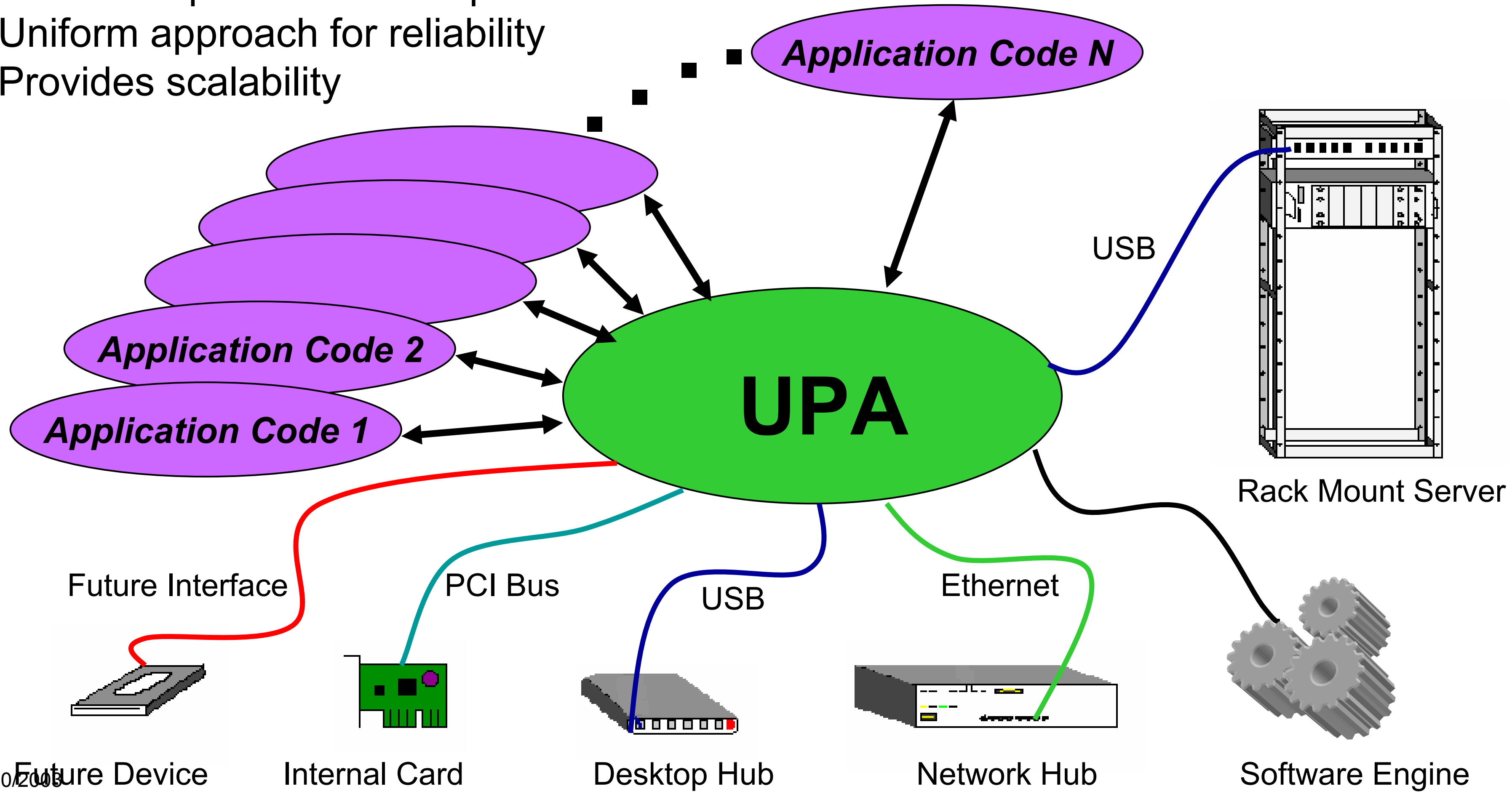
SPI Activity
Research & Development

- Goal
  - Advance software protection technologies on desktops through super computers
- Focus
  - protect developed software
  - Develop protectable software
- Key Activities
  - Protect developed code
  - Develop protectable code
  - Promote usability, scalability, and maintainability
  - Universal Protection Architecture

R&D Activity

Unified Protection Architecture (UPA)

- Minimizes performance impacts
- Uniform approach for reliability
- Provides scalability

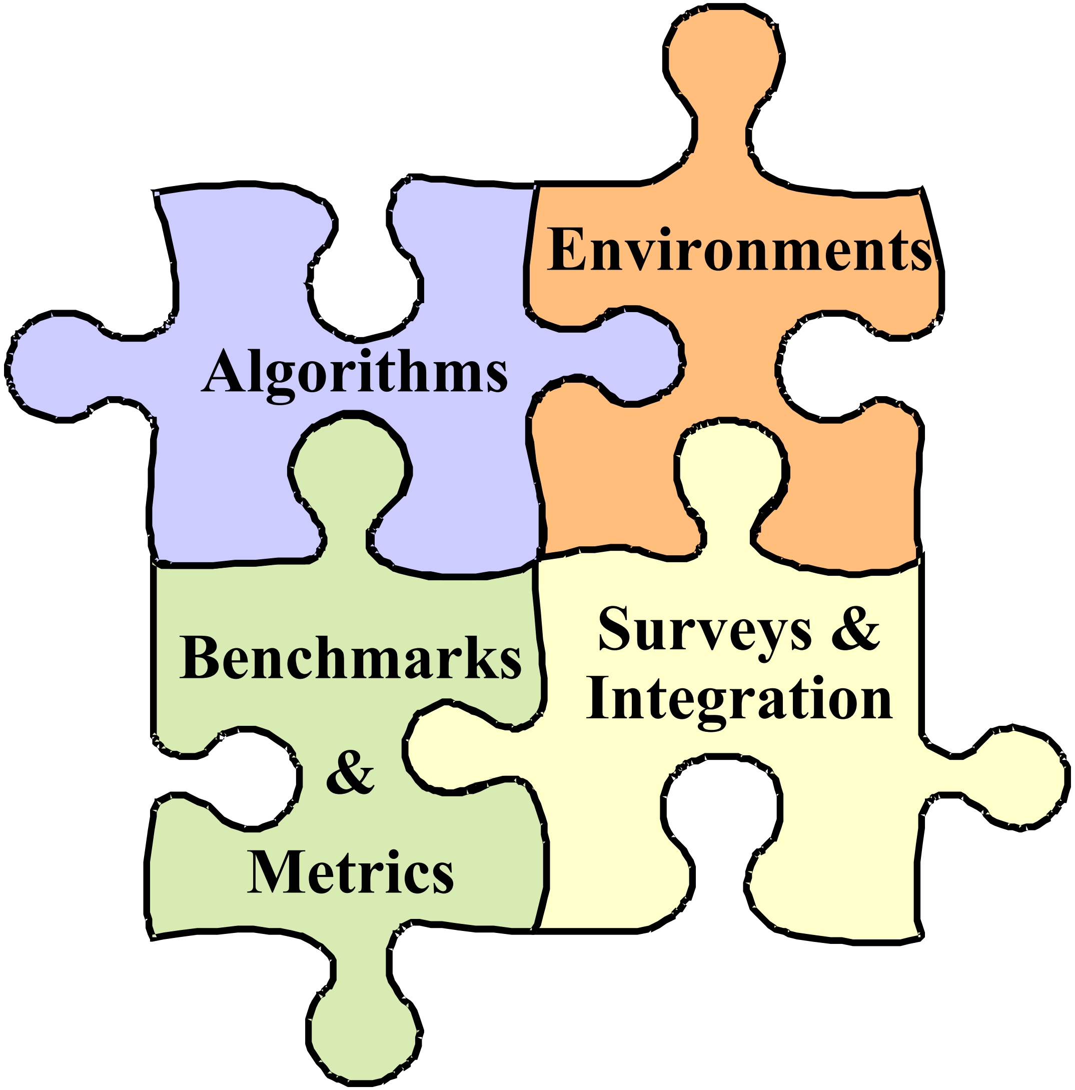


R&D Strategy
Vision*

- Protection
  - Application security without development or performance penalty
  - Protection techniques tailored to the criticality of the code, the operational and threat environments, and computational power
  - Scalable and customizable protection
- Detection
  - Self monitoring of protected software for
    - Malicious activity
    - Code integrity
- Reaction
  - Array of autonomous self defense measures for protected codes
    - Modification of code/data
    - Self destruction
    - Reporting

Secrets and Lies: Digital Security in a Networked World, Bruce Schneier, John Wiley and Sons, Inc., 2000

Research Areas



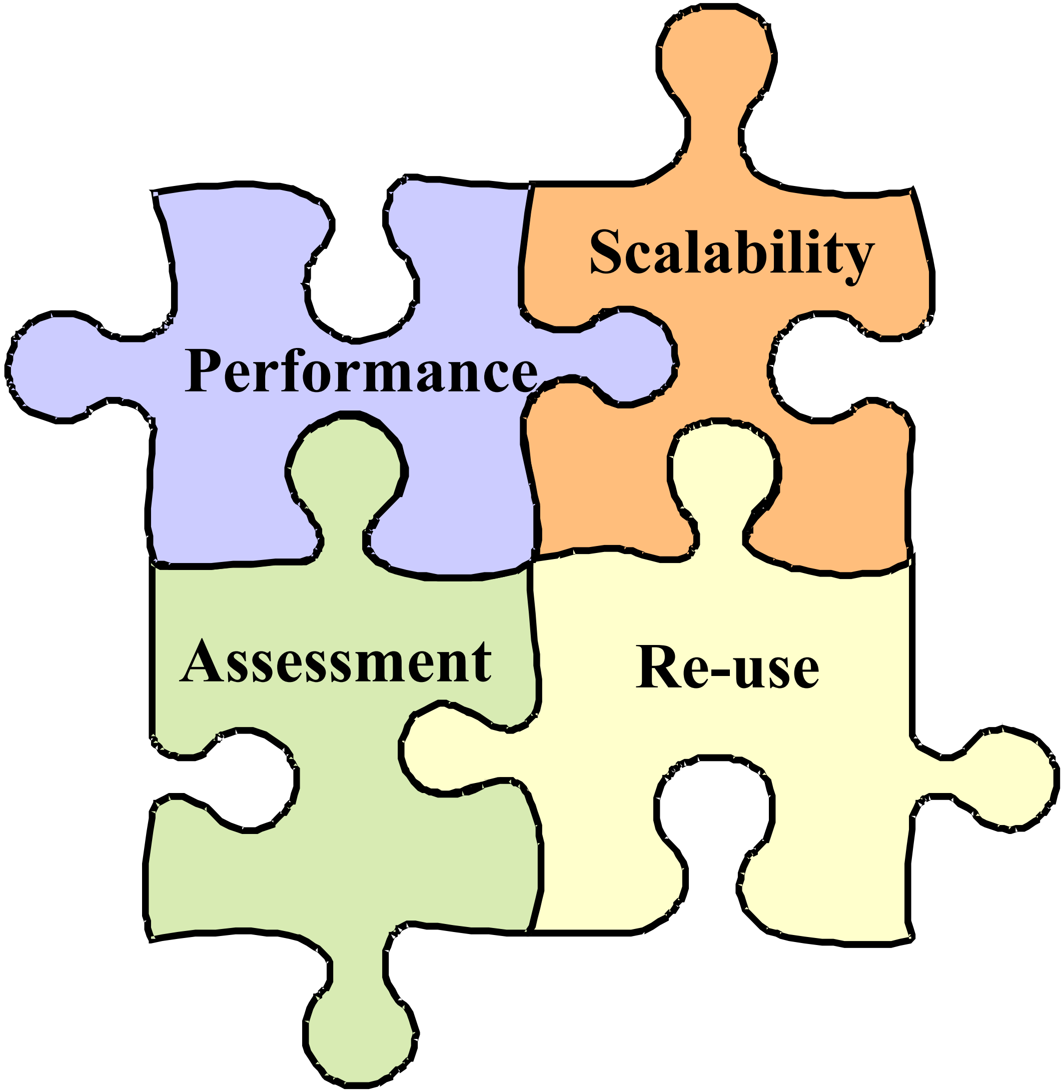
Research and Development Dimension of the Challenge

- Framework for Analysis
- Technology for achieving SPI goals
- Quantification of code complexity
- Performance metrics

Framework for Analysis

- Goals of attacks
  - Reverse engineering all or parts of a code
  - Allowing limited or unrestricted execution
  - Tampering with the code
- Type of effort
  - Human effort (from expert to ordinary skills)
  - Generic tool availability (COTS, open source)
  - Specialized tools (what is possible by skilled adversaries?)
  - Number of allowed executions
  - Time and availability of code required for attack
  - Level of mathematical or logical symbolic analysis

Research Issues



R&D Technology Hardware Approaches

- Freestanding, obfuscated code only
- Obfuscated code with an “authentication” host on the network
  - Kind of network
  - Kind of host processing
- Obfuscated code with on-board hardware module/hardware
  - Proprietary
  - TCPA, Palladium, COTS
- Other approaches and combinations of the above

R&D Technology Software Approaches

- Obfuscation of code
  - At source code level
    - source restructuring
  - At executable level
    - Obfuscating compiler
    - Post-compilation obfuscation
  - Three address code representation

- Opaqueness of code and/or procedures
  - Obfuscate procedures
  - Complexity of parameters passed, nesting
  - Use of special hardware for function evaluations

Quantification of Code Complexity

- At source, intermediate and/or executable levels
- Basic block count and size
- Structure of program control flow graph
  - At basic block level
  - Static analysis
  - Dynamic analysis for a “typical” execution
  - Loop structure and depth
- Data structure complexity
- Procedure call depth, count, parameter passing (indirection, etc.)

R&D Metrics

- Performance metrics
  - Possible levels of protection as a function of code complexity and attack effort
  - Performance loss as a function of protection
  - Preprocessing effort required for protection
  - Cost of protection – hardware, management, etc
  - Cost of versioning, updating, etc.

Research Objectives

- Protect developed code
- Develop protectable code

Protecting Developed Code

- Continually assess the state-of-the-art
  - Develop capabilities to maintain security edge in face of technological advances
- Areas of concern/research interest
  - Decompilers
  - Watermarking
  - Compilers
  - Multiprocessors
  - Disassemblers
  - Obfuscation
  - Debuggers
  - High Performance Computing

Developing Protectable Code

- Continually assess state-of-the-art
- Areas of concern/research interest
  - Secure development environments
  - Automatic pedigree generation and validation
  - Automatic developer logging and profiling
  - Software development methodology modification
  - Virtual machine wrappers
  - Multiprocessors

R&D
Current Efforts

- Development of topics
  - Obfuscation and Watermarking
  - Tampering & Reverse Engineering
  - Architectural Degradation
  - Tamper Detection & Response
  - Binary Code Transformation
  - AT Protection Thru Obfuscation

R&D
Protection Research Avenues

- Benchmarks, metrics, and test suites
  - Autonomous red team
  - Ontology and lexicon
- Secure development environment
  - Architecture through maintenance phases
- Black box application of protection technologies
- Cross authentication of components
- Improved watermarking and obfuscation

R&D Strategy
Protection Research Avenues (cont)

- Autonomous, secure assembly and verification of security capabilities
  - Composable protection techniques
- Data
  - Container-based protection of data
- Inherently secure programming languages
- Multiprocessor software protection
- Operation on untrusted hardware

R&D Strategy
Detection Research Avenues

- Autonomous attack detection and defense
- Ontology and lexicon
- Comprehensive threat description and threat models
- Voting schemes to “detect” subverted software or nodes
- Continuous or pushbutton verification that the software is not changed
- Security gauges

R&D Strategy
Reaction Research Avenues

- Adaptive defense
- Variable precision and accuracy
- Benchmarks and test suites
- Autonomous recovery and repair
- Isolation of subverted nodes
- Secure migration of subverted processes
- Pedigree to track back to developer

Strategic Issues

- Education
- Technical Thrusts
- Government-wide Coordination
- Risk Management

Strategic Issues
Education

- Issue
  - Education is critical to cultivate awareness of the threat and the requirement for application code security across the DoD
- Discussion
  - Education is required at all levels
  - SLAG and IPT must have reps from key DoD and government agencies and assist in education process
  - SPI will encourage commercial entities to issue statements supporting the initiative
  - Web will be a key education tool
- Way Ahead
  - Involvement is essential…..

Strategic Issues
Technical Thrusts

- **Issue**
  - SPI must identify and protect existing critical codes and develop secure software development tools/environments for future applications
- **Discussion**
  - Currently rely on the inherent obfuscation provided by current higher order language compilers
- **Observation**
  - Ensure R&D investments address these core issues and backstop the technology risk
  - Developing comprehensive and integrated R&D strategy to meet short and long term objectives

Strategic Issues
Government-Wide Coordination

- Issue
  - Critical applications are shared among government organizations
  - Software protection policy, techniques, and procedures must be consistent
- Discussion
  - All actions must be coordinated at senior levels
    - National Cyberspace Policy
    - DoE, NASA, and others
      - Sharing of applications and procedures
      - Common requirements definition
- Way Ahead
  - Include key government organizations in activities

Strategic Issues
Risk Management

Issue

- SPI program success requires balancing multiple, competing factors

Discussion

Established DoD acquisition process	**vs.**	**Typical academic-based software development process**
Compartmentalization of facts	**vs.**	**Education**
Strong protection measures	**vs.**	**Ease of use**
Directed compliance	**vs.**	**Voluntary implementation**
DoD only	**vs.**	**Government-wide implementation**

Observation

- R&D to enable usability, scalability, and maintainability
- Definitive policy to institutionalize SPI
- Education and coordination to encourage compliance

Conclusion

- Application software represents a significant portion of the DoD’s intellectual property
  - Significant investment in both time and money
  - Enables development of next generation weapon systems
- Protecting critical application software **a**llows U.S. Forces to:
  - Maintain a technological advantage over our adversaries
  - Extend the operational life of critical systems

Conclusion, cont.

- Software protection is an integral layer of the defense in depth concept for information assurance
  - Compliments network security and access controls
  - Provides application centric technology to reinforce application security policy