AFRL-IF-RS-TR-2001-278
Final Technical Report
January 2002

MARKET-BASED SERVICE QUALITY DIFFERENTIATION (MBSQD) (FORMERLY CARTOGRAPHY OF CYBERSPACE)

BBNT Solutions LLC

Sponsored by
Defense Advanced Research Projects Agency
DARPA Order No. J776

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

AFRL-IF-RS-TR-2001-278 has been reviewed and is approved for publication.

APPROVED: Nancy A. Roberts

NANCY A. ROBERTS
Project Engineer

FOR THE DIRECTOR: Michael Talbert

MICHAEL TALBERT, Maj., USAF, Technical Advisor
Information Technology Division
Information Directorate

If your address has changed or if you wish to be removed from the Air Force Research Laboratory Rome Research Site mailing list, or if the addressee is no longer employed by your organization, please notify AFRL/IFTD, 525 Brooks Road, Rome, NY 13441-4505. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document require that it be returned.

REPORT DOCUMENTATION PAGE			*Form Approved* *OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** JANUARY 2002	**3. REPORT TYPE AND DATES COVERED** Final Apr 00 - Jul 01

4. TITLE AND SUBTITLE
MARKET-BASED SERVICE QUALITY DIFFERENTIATION (MBSQD)
(FORMERLY CARTOGRAPHY OF CYBERSPACE)

5. FUNDING NUMBERS
C - F30602-00-C-0088
PE - 63760E
PR - IAST
TA - 00
WU - 02

6. AUTHOR(S)
Mike Frentz, David Mankins, Rajesh Krishnan, John Zao, Ceilyn Boyd, Bill Nelson, Wallace Feurzeig, and Oliver Selfridge

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
BBNT Solutions LLC
9810 Patuxent Woods Drive
Columbia Maryland 21046

8. PERFORMING ORGANIZATION REPORT NUMBER
N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Defense Advanced Research Projects Agency
3701 North Fairfax Drive
Arlington Virginia 22203-1714

Air Force Research Laboratory/IFTD
525 Brooks Road
Rome New York 13441-4505

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFRL-IF-RS-TR-2001-278

11. SUPPLEMENTARY NOTES
Air Force Research Laboratory Project Engineer: Nancy Roberts/IFTD/(315) 330-3566

12a. DISTRIBUTION AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*
This effort was initially part of the DARPA Information Assurance Science and Engineering Tools (IASET) program and then changed to part of the Survivable Wired & Wireless Infrastructure for Military operations (SWWIM) program. The objective was to develop baseline taxonomy on which to build a foundation for the development of a formal science of cyberscience, survey related scientific areas to extract specific relevant concepts which would contribute to the formalization of cyberspace, extend these identified models in the IA domain and validate their usefulness in the context of information assurance. BBN focused on the Economic task of this effort. In particular, they looked at market-based approaches to network quality of service. They designed an architecture to counter the asymmetric threat posed in typical computer attacks by incorporating mechanisms into the network interactions. The particular attack that they focused was a DDoS attack. They were able to generate baseline experimental results from this model that demonstrated the desirability of developing a real-time infrastructure based market-based ideas.

14. SUBJECT TERMS
IA Taxonomy, Cyberscience, DDoS Attacks

15. NUMBER OF PAGES
84

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

TABLE OF CONTENTS

TABLE OF FIGURES

Preface

Cartography of Cyberspace was begun in April 2000 as part of the DARPA Information Assurance Science and Engineering Tools (IASET) Program. IASET was one of a half dozen Information Security-focused R&D programs initiated during 2000 by the DARPA Information Systems Office (ISO). Of the six efforts, IASET had the most theoretical charter – its goal was to begin to develop a science-based foundation for information security by leveraging concepts and formalisms from other branches of science and applying them to the computer science realm. The tenor of the effort was notably aggressive and emphasized "thinking out of the box" approaches to Information Security (IS).

Six months into the effort, the IASET program was terminated as a result of management changes at DARPA. The ISO Information Assurance efforts were redirected from a science-driven long-range focus to instead strive toward bringing about nearer term operationally-driven security improvements. The majority of the ISO Information Assurance work was then transferred to the DARPA Advanced Technology Office (ATO) and the funded projects realigned with the new charter.

As a result of the changes in the overall program focus, work on the original "Cartography" charter was stopped in November 2000 and we searched for a near-term operationally-oriented technology objective for the remaining funds that could leverage some of the scientific areas we were researching, but would be better aligned with the new DARPA IS charter. The Cartography effort had proposed to adapt technologies from selected mature (relatively speaking) sciences that have some apparent analogies to information security and compose those fundamental concepts into an overall model for cyberspace interactions. We had been actively researching several broad areas under the Cartography charter: development of an overall taxonomic framework for interactions in cyberspace, investigation of economic models as motivators and moderators of network interactions, and a small research option focused on the investigation of thermodynamic analogies to information processing. A synopsis of the research conducted under the Cartography effort is presented in Part II of this report.

After the management change of direction, we decided that the most promising use of the remaining funds would be to focus only on the "economics" thrust of the Cartography effort. An economics, or marketplace formulation, is a very natural model describing the "motivating force" in many non-intimate human interactions, whether physical or electronic. One of the chief causes of information security problems on networks is the lack of accountability of most electronic interactions when compared to the accountability mechanisms that exist for nearly all routine interactions that occur in our physical lives. Our new focus was to investigate the use of market-driven access control methods to provide non-disruptive, self-compensating methods for controlling access to owned resources on the web.

We began in November 2000 with an emphasis on market-based methods to ensure Quality of Service (QoS) but, after reviewing the many active long term efforts focused on the development of QoS on networks (e.g. IntServ/RSVP or DiffServ), we decided that an enclave-to-enclave protection mechanism layered onto the existing TCP made the most sense in the new 3GS programmatic context by affording the possibility of our being able to provide near-term results (the QoS methods being pursued rely heavily on advances by IETF working groups and involve significant changes to the underlying network technology infrastructure that will require years to evolve and whose motivations are not necessarily well-aligned with military usage).

The renamed effort is called Market-based Service Quality Differentiation (MbSQD) to more accurately reflect the *service quality differentiation* we think may be achievable through the use of compensation based access control mechanisms overlaid on existing networking infrastructure. The goal is to achieve positive control over *owned resources* while still permitting them to be utilized by a largely anonymous user base. The compensation mechanisms that are candidates for this technology include both micropayments (scrip-based) mechanisms as well as Proof-of-Work (PoW) mechanisms that provide some computational "payment" as proof of the earnestness of the apparent requesting entities.

Our goal with MbSQD was to formulate the concepts required to implement such a framework and develop enough of an experimental framework to permit baseline validation of the concepts as much as possible under the remaining Cartography funding. Part I contains a summary of our work on MbSQD. Two appendices are included that document the *ns*-based simulation model that was developed and provide some references to the *ns* network simulation environment.

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Many thanks to: David Mankins, John Zao, Ceilyn Boyd, Rajesh Krishnan, Bill Nelson, Wally Feurzeig, and Oliver Selfridge for each of their unique contributions to these efforts.
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Part I – Market-based Service Quality Differentiation (MbSQD)

Executive Summary

Distributed Denial of Service (DDoS) attacks exploit the acute imbalance between client and server workloads to cause devastation to the service providers. The combination of a distributed gateway architecture and a payment protocol that imposes dynamically changing prices on both network, server, and information resources can defer some cost of initiating service requests back onto the requesting clients. These costs can either be in terms of monetary payments, computational burdens, or some combination thereof. By employing different price and purchase functions, the architecture can provide service quality differentiation and furthermore, select good client behavior and discriminate against adversarial behavior. Though this work is preliminary in terms of the number of experiments conducted to date, judicious partitioning of resources using different pricing functions may very well improve overall service survivability.

The majority of the results presented in the Part I of this report have been submitted to the 2001 Annual Computer Security Applications Conference as a paper entitled, "Mitigating Distributed Denial of Service Attacks with Dynamic Resource Pricing".

1. Introduction

Denial of Service continues to be a pervasive problem for Internet services, as evidenced by the recent denial of service at CERT [**BBC01**]. Such attacks require each attacking machine to perform only a small amount of work, relying on the cumulative efforts many machines to overload the victim machine. Attacks range from simple ICMP ping requests to sophisticated attacks that are difficult to distinguish from a sudden spike in legitimate use (a.k.a. "flash crowd" [**Niv73**] or "the Slashdot Effect").

In a DDoS attack, the perpetrator(s) may spend weeks or months subverting hundreds or thousands of machines by exploiting well-known security flaws. Once the machines are subverted, the perpetrator installs the tools that will implement the attack. On command, the prepared machines (known as "zombies") collectively target a specified victim with a packet storm consisting of repeated datagram packet requests. These packets may take many forms, such as an ICMP ping request, a UDP packet (such as a DNS request), or TCP SYN floods.[1] The packets may also have forged return addresses, allowing one machine to generate requests that appear to be coming from hundreds of machines, and making the sources of the attack difficult to trace. This problem may intensify since newer operating systems are being introduced that may make it easier to spoof IP addresses [Gib01]. Halting these attacks is extremely difficult and typically relies on filtering at the router (or just waiting out the "packet storm").

Packet-flood attacks, such as these, are the most widely diagnosed today [CAIDA01]. As measures to counteract them are put in place, future attacks will most likely include resource depletion attacks that are more difficult to distinguish from legitimate requests. The effectiveness of an attack will be driven by the resource most tightly constrained relative to the attack parameters. The attack could focus on a single server within an enclave (or on the pipes themselves).

In this report, we explore the use of dynamic resource pricing strategies to mitigate various DDoS attacks and to improve service survivability. We propose an experimental architecture and protocol for a Market-based Service Quality Differentiation (MbSQD) system. We have prototyped the MbSQD system using the ns-2 simulator [**NS2**], and have experimented with various price-based controls for DDoS mitigation. These include both proof-of-work based approaches that exact a price in terms of a computational burden, and monetary-like micropayments. The MbSQD system is designed to work as an overlay on existing network security infrastructure, but similar concepts may also be applied at the lower networking layers (e.g. in conjunction with Diffserv/RSVP).

Our experiments thus far have focused on a monopolistic market model where competition among customers desiring to access a particular service creates an effective

[1] TCP SYN is the first message in the three-way handshake that initiates a TCP connection [**Pos81**]. Before SYN attack avoidance strategies were widely deployed, a victimized machine would allocate a TCP control block (TCB) and then wait for the connection to open; a SYN flood attack would eventually consume the memory space allocated for TCBs, and prevent legitimate users from opening a TCP connection.

market for that service. Our initial experiments have focused on mitigating DDoS attacks against a server enclave that has very high bandwidth access to the Internet.

While pricing has been used to influence consumer behavior in the past [**Bak99**][**Fis99**], our work is novel in that it actively uses pricing mechanisms as *discriminants* to penalize adversarial behavior. Apart from providing service differentiation, each discriminant protects the service against a set of adversarial behaviors. We believe that by partitioning resources and protecting them using different discriminants, we can improve the overall service survivability.

This report is organized as follows. In this section, we discuss limitations of existing strategies to deal with DDoS attacks and introduce our proposed approach. In Section 2, we present our proposed approach in detail, as well as provide background information on related work. In Section 3, we describe the MbSQD architecture. In Section 4, we describe the MbSQD simulation experiments and our results. We summarize the lessons learned and provide directions for further research in Section 5. Bibliographic references are included Section 6.

1.1. Existing DDoS mitigation strategies

Detection of subverted machines on local subnets

The first tools used to launch DDoS attacks had recognizable signatures in their control messages. Intrusion detection techniques could therefore be applied to watch for the DDoS control messages, notify a system administrator of subverted machines on their network, and allow them to remove the offending code.

Apart from imposing a considerable burden on network security analysts, this approach does not work well with many of the newer attacks that utilize encrypted commands to hide them from security scanning processes.

Choking off attack packets near their sources

A second approach to palliating DDoS attacks concentrates on eliminating packets near their source. Errant packets with forged return addresses are filtered in the following ways. In *ingress filtering*, routers are configured to drop packets coming in on an interface which the router would not use to route return packets. In *egress filtering*, gateways on a subverted machine's LAN refuse to forward packets that do not originate on that LAN.

Ingress filtering depends on routing paths being symmetric. It may discard legitimate packets under asymmetric routing or while routing is changing to respond to new circumstances. Egress filtering depends on correct configuration of leaf routers at customer sites.

Reducing the impact of an attack

A third approach to mitigating DDoS attacks is to establish resource mirrors so that the attacks affect only a fraction of the users of the service. For example, Akamai provides a service to content providers which causes their content to be cached in "edge proxies" scattered around the edges of the net. A client is directed to the best available edge proxy through interaction with the DNS mechanism. This solution is only available to fairly wealthy content providers, however.
Another possible approach is to reduce the amount of resources consumed during a DDoS attack. At times of contention, NetBSD will accelerate its timeouts of partially opened TCP connections. Linux implementations save no state when receiving a TCP SYN. Instead, the initial sequence number is derived from information that will be available in the return packet from the connecting machine (SYN Cookies).

Monitoring traffic volume

A fourth approach is to look for suspicious traffic destined for a particular destination. The method will be ineffective if traffic flow confidentiality is employed.

Tracing attack packets through the Internet

There are a number of proposals to augment routers so that packets used in a DDoS attack may be traced backwards to their source, so the responsible ISPs or system administrators can be contacted to shut the attacks off at their sources (see for example, [**Sno01**]). Such mechanisms work after-the-fact; however, by increasing chances of being able to identify the perpetrator, they can serve as deterrents.

1.2. Proposed strategy for DDoS mitigation: pushing costs to clients

Denial of service attacks create shortages of a resource such as bandwidth or computing cycles through the creation of an artificial demand. They work because the cost of the transaction falls overwhelmingly on the server.

This characterization suggests an attempt to mitigate DDoS attacks using an economics-based approach to establish a "market" for services to allocate resources more fairly and to push some cost of the transaction onto the attacker.

To make services more robust against a DDoS attack, we propose the following combination of strategies:

1. Increase the barrier to entry by using a pricing-based scheme in which the price of entry varies with the load level. This will throttle the machines used in the attack, thereby forcing the attacker to employ (or subvert) a larger number of machines.
2. Use a differentiated model. Provide prioritized access to classes of users; even if the DDoS attack raises the price so high that the lower priority classes get locked out, the higher priority clients can still access the service.

Use a dynamic, differential pricing mechanism to penalize clients that are responsible for a load on the server. This typically requires flow monitoring and isolation capabilities in line with those of Diffserv [**Ful00**][**Arq**] [**Bla98**].

None of these strategies are sufficient in isolation — situations can be defined where each may contribute to increasing the availability of the network services. In the next section, we describe our proposed approach in detail.

2. Technical Approach

2.1. General Observations about DDoS Attacks

DDoS attacks succeed primarily because the load falls disproportionately on the server. Often a small number of clients are sufficient to launch a successful resource depletion attack. Furthermore, sophisticated DDoS attacks can be virtually indistinguishable from genuine overload, at least at the time of the attack due to the limitations of the information available and the kinds of analysis possible in real-time. Thus, a mechanism is necessary to transfer the burden to the client, as well as to control the damage that any single client can cause.

Second, *few existing systems implement* ***discriminants*** *to identify and penalize adversarial behavior*. The penalty imposed on (or cost to) the attacker is the same as that to a legitimate client. Specifically, there is no way to limit the damage that can be caused by a single or selected group of machines. A wealthy adversary can quickly create a population inversion within the system, replacing legitimate client requests with ones that simply deplete resources. Thus, a mechanism is required to actively select good behavior and discriminate against adversarial behavior; this mechanism should match the response to the level of aggressiveness of the attacker.

Third, *systems must provide* ***multiple service classes****, and must partition and isolate resources effectively across service classes*. Otherwise, an attack launched from one service class can potentially bring down the entire system. Also, it is not sufficient that different classes merely impose a different cost to the client based on the same discriminant. This will not deter a wealthy adversary from simply using the same attack on all classes. It is likely that any single strategy will remain vulnerable to some form of attack and it is only a matter of time before it is discovered and exploited. Thus, in order to be survivable, the system should be partitioned into service classes that use different discriminant mechanisms; this will deter even the wealthy adversary since a number of different attack strategies must be deployed simultaneously to bring the service down entirely.

Fourth, *the pricing mechanisms should take into account the current and projected values of supply and demand* (both network and server resources) as well as other attack indices such as the number of incident reports that match a particular flow, network, host, or user.

Fifth, *price indicators are useful to monitor the onset of attack.* These are required both to trigger appropriate responses to attack as well as to analyze data over a longer term.

In our work, we argue that dynamic resource pricing based on fungible and non-fungible payment strategies can be used to build a solution to mitigate DDoS attacks. In this section, we describe in detail our dynamic pricing based approach to satisfy the requirements that we presented.

Micro-payments can provide a useful side benefit by providing a uniform means of resource accounting, pricing, and arbitration. Careful consideration must be applied to make sure that the micro-payment mechanisms do not impose an undue performance penalty. Specifically, in the absence of attack, the performance should be nearly comparable to a system that does not use the payment mechanisms. There is prior work on how pricing can be used to influence consumer behavior, how to integrate pricing mechanisms with OS and network resource management mechanisms. In this report, we instead focus on how pricing strategies can be used to mitigate DDoS, and improve overall service survivability.

2.2. Types of Micro-payments

In this section, we provide some background on micro-payments and proof-of-payment strategies that are used for pricing resources.

Fungible vs. non-fungible micro-payments

There have been a number of digital payment and micro-payment schemes proposed to support digital exchanges [**Riv97**]. These have been primarily proposed to support digital commerce, but some researchers have also looked at the use of payment schemes as a means of mitigating denials of service.

Fungible (or transferable) digital payment schemes range from anonymous cash schemes such as David Chaum's Digicash [**Digi**] [**Cha83**], to electronic checks and payments [**Pay**], postage [**Hash**], to bartering services (as in Mojo Nation [**Mojo**]).

Cash-like schemes do not require on-line verification --- the server can validate the coin by examining it. They therefore have low latency with respect to coordination with external servers, however the validation process typically requires significant computation or memory usage overhead for the server itself. As a result, high integrity cash-like payment schemes may not be compatible with fielded servers.

Many of the alternative fungible payment schemes are analogous to a check or credit card transaction and require some type of on-line verification of payment --- a server must connect online with a bank and verify. On-line verification is susceptible to high latency and provides an alternative critical path target for DoS.

Scrip-based systems are an attempt to reduce the latency of verification by making the verification a purely local operation on the server. These systems, such as Compaq's Millicent [**Milli**], are intended for ultra-micro-payments (on the order of thousandths of a penny). In Millicent, a server issues (or mints) its own scrip to be used by clients to pay for services. Since the server issues the scrip, it can verify it with very low latency (possibly requiring as little as a table lookup). Clients obtain a quantity of scrip from network *scrip brokers* using one of the high-overhead bulk-payment schemes geared for larger expenditures. Millicent allows a server to give a client *change* (which the client may later redeem with their broker).

Another way of escaping the need of on-line verification is to extract a payment in the form of "work" or computation [**Dwo92**][**Jue99**][**Jak99**][**Hash**]. The server sets a computational task to the client that must be solved before the server will proceed with the client's task. To be useful, the task must be computationally hard to solve, but a solution must be simple to verify, e.g., factoring a large number --- the size of the number is determined by the prices of the service, which in turn reflects the load (demand) on the server. Once a solution is provided (showing that the client has done more work than the server will end up doing in honoring the request), the server will then expend its resources on the client's request.

Convertible vs. non-convertible currencies

A *non-convertible* currency scheme has a limited scope where it can be used and cannot be exchanged for other types of currency. A *convertible* currency on the other hand can be exchanged for other types of currency.

The former is useful to permit priority access to specific resources for a particular subset of known potential users (e.g. a military squadron). The latter has advantages in situations requiring high priority access for a dynamically changing subset of potential users drawn from a general population.

Infrastructure issues

Our system architecture described in Section 3 supports both proof-of-work (POW) as well as these different kinds of micro-payment systems.

The broker-based architecture has the advantage of not requiring any change to existing end-systems. In the case of POW, the client brokers can pass the problem to the client or it can splice the TCP connection and act as a proxy that responds to POW challenges.

Protocol Issues

The protocols must be efficient in terms of number and size of messages exchanged and must use as few round-trips as possible especially over network paths with high bandwidth-delay products.

An overwhelming number of applications and services on the Internet rely on the TCP protocol. This includes email (SMTP), remote login (SSH and TELNET), file transfer (FTP), web (HTTP), and peer-to-peer information exchange protocols (Napster). Inter-domain routing protocols such as BGP-4 also use TCP to exchange routing information. Therefore, it is useful to consider pricing support for TCP.

In Section 3, we describe an efficient implementation based on adding a micro-payment option to TCP. This option augments the TCP three-way handshake with an initial request and response, in a fashion similar to **[Tsu95]**. The client can agree to use micro-payments and pay an initial price for a given number of bytes transmitted or received, peak send rate, and duration. Additionally on expiration of the bandwidth or of the duration initially paid for, additional request for payments can be generated and sent via the TCP option. For other approaches see work done by the W3 Consortium [**MPAPI**][**MPTP**].C. Pushing Costs to The Clients

The first step in our approach consists of pushing a part of the cost to the client. We achieve this in the form of micro-payments that are exacted per service unit as per a given subscription model. As described earlier, we consider two types of payments: proof-of-work schemes and scrip-based schemes. Use of the service can now be limited by making the cost of launching an attack non-negligible to the client. With payments, the adversary will have to subvert more clients to increase the intensity of the attack.

2.3. Dynamic Resource Pricing as Discriminants

As the logical next step, we implement a dynamic pricing strategy that can favor good user behavior and discriminate against aggressive adversarial behavior.

In our model, we have a time-varying price function for each service. The price function relates the price of the service to supply, demand and other factors.

Each user has a utility function that determines how much they are willing to pay for a unit of a given service as well as how many units they will consume at a given price at any given time. The cumulative effect of the utility functions drives the overall demand for the service. Furthermore, the spending behavior can be monitored in a distributed fashion for anomalies. For example, a typical user may not pay more than $3 for a pay-per-view and a unit price of $100 being paid is an indicator of a likely subverted machine and an attempted denial-of-service in progress.

We can define a price $P(i,j,t)$ of a unit of service i, for subscriber j, at time t, that depends on a supply vector S, demand vector D, and an auxiliary vector A. The supply vector consists of supply parameters, for example, in the case of a web server it can be the maximum number of connections that can be served, pending connection queue size, and server throughput in terms of given request sizes. The demand vector is constructed by monitoring the offered load, for example, connection arrivals per second, bandwidth consumed per connection, and duration statistics for connections. The auxiliary vector accounts for factors such as vulnerabilities of software being used, number of incidents being reported such as signatures of hostile packets or reported mismatch in spending profiles. The *S*, *D*, and *A* vectors can include both per-flow and aggregate statistics.

Selection of one particular user behavior over another occurs due to interplay of the price and utility curves. While the idea of using pricing to mold user behavior is known, our approach extends this idea to discriminate against adversarial behavior.

Consider the following scenario of a web server. A number of different kinds of attacks can be launched against a web server. These include exploiting OS and protocol stack vulnerabilities such as SYN floods and buffer overflows, connection depletion attacks using idle connections, depleting server resources using requests that are expensive to process, inundating the request queue with bogus connections, and depleting network bandwidth by requesting large volumes of data. Different pricing strategies are required to protect against each of these attacks.

A robust pricing function must monitor a number of indicators, including the frequency with which users generate requests. Even if individual users remain anonymous, control can be exercised at the granularity at which the flows can be isolated and monitored. For example, pricing controls can still be exerted at the level of the originating ISP. Resources can be partitioned between anonymous users and known subscribers with long-term relationships. ISPs can protect themselves by using similar price controls and other monitoring within their administrative domains. Adoption of egress filtering and IP trace back can further aid in enforcing such controls.

Pricing functions must be robust against any potential new attacks enabled due to the pricing strategy itself. For example, care must be taken that an adversary is not able to populate the system with fake requests when the price is low and increase the price for legitimate users. This means that a pricing function that is robust against connection depletion attacks must necessarily limit the connection duration and require that each connection be refreshed periodically to protect against zombies.

2.4. Price-based Service Quality Differentiation and Survivability

We have discussed how different pricing functions can be used to select different kinds of user behavior, thereby protecting against some classes of attacks. A single price function is unlikely to provide sufficient service quality differentiation necessary to satisfy a wide range of user requirements. Furthermore, a single pricing function is unlikely to protect the system against all forms of attack. Given enough resources, a wealthy adversary can easily thwart a system that uses a single pricing strategy.

Therefore we propose the following strategy. We partition and isolate the available resources among various service classes. For example, in the case of a network router, weighted fair queuing can be used to partition resources. Furthermore, resources can be isolated using VPNs. Server resources can similarly be partitioned and isolated among service classes by using OS prioritized scheduling techniques and by virtual OS techniques respectively.

We hypothesize that the survivability of the system can be further enhanced by associating different partitions with different discriminants (pricing functions) that are robust against different classes of attacks. With this approach, a successful resource depletion attack will not only require more resources, but also the simultaneous launch of different forms of attack for each service class.

Possible extensions of this strategy include dynamic policy iteration that progressively improves robustness against a larger class of attacks or a randomized policy iteration that makes it harder for the adversary to guess the pricing function and determine efficient attack strategies.

3. System Architecture

Figure 3-1 illustrates the operational architecture for the MbSQD system. The MbSQD system employs a distributed architecture with three distinct features:

1. *Deployment of resource brokers at network boundaries:* MbSQD will use stateful packet filters and/or application proxies to control resource utilization at the logical boundaries of user subnets (on either the provider's or the client's sites). This architecture has the following advantages:
    a. The operation of client and server applications will not be affected by the

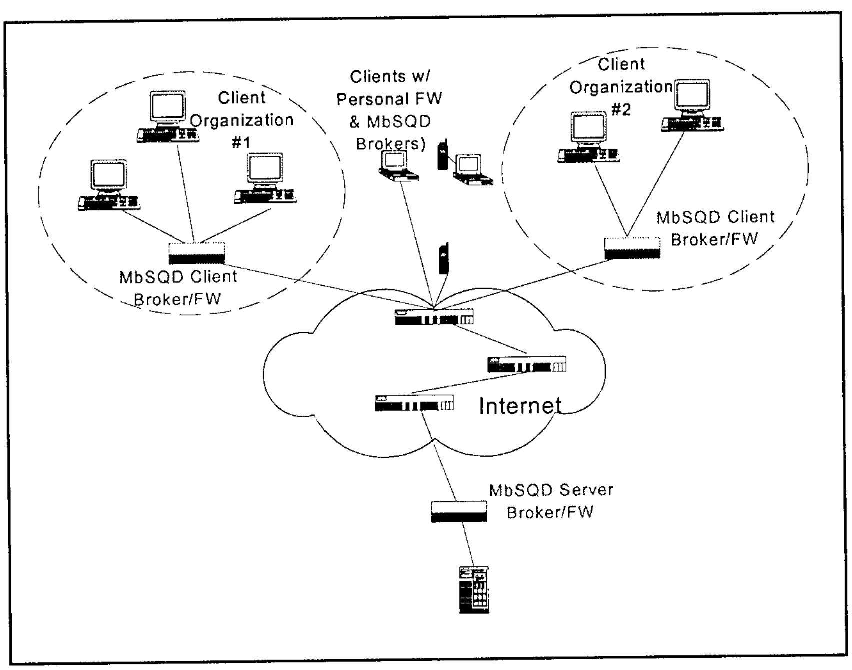


Figure 3-1. MbSQD Operational Architecture

deployment of the traffic control system; in fact, both clients and servers may not be aware of its presence except due to apparent changes in network throughput and device performance. No modification of end-node protocols and applications is necessary.

b. The architecture may be used to control the utilization of both network and information resources including network throughput, server capacity, information access and device usage. The brokers may be installed at the border gateways of autonomous systems if they are intended to be used for inter-domain traffic control, or they can be placed at the "choke points" of server access if they are used to control information access and/or device usage.

c. Price-based resource management can be made *mandatory* in order to obtain the highest priority access privileges (instead of discretionary as it is imposed by devices under the control of network management).

2. *Employment of client-side defined price and purchase decision functions:* MbSQD achieves rapid control of resource utilization by relying on the interactions between the dynamic pricing of resources and the autonomic purchase decisions made by individual clients. By employing different pricing functions, MbSQD can favor the clients that exhibit desired behaviors or use certain forms of purchase decision functions. This behavioral discrimination is a unique feature of the dynamic pricing scheme.
3. *Operated with payment protocols integrated with TCP:* MbSQD uses a three-message handshake protocol to initiate service request and conduct payment transaction; it also uses two-message handshakes to pay for continuous resource use. The initial payment protocol can be readily integrated with the TCP connection establishment, and the renewal handshakes can be "piggy-bagged" onto TCP data segments as options. The protocols are also designed to support different forms of payments including scrip and proof-of-work. A micro-payment infrastructure will be needed in order to use scrip.

In the remaining parts of this section, we will examine the three essential components of MbSQD: the resource brokers that are installed at the boundary gateways, the business logic that implements the price and the purchase decision functions, and the payment protocol that conduct the business transactions.

3.1. Resource Brokers

The MbSQD resource brokers are to be installed in gateways at the boundary of Internet sub-networks, where they can function as application proxies or packet filters. The brokers determine whether datagrams going to and from certain *IP addresses* using specific *transport protocols* and *port numbers* should be passed or discarded. Functionally, the resource brokers may be the proxies of application clients or servers. The server and client brokers enable passage of datagrams based resource prices and budgetary considerations. Operationally, each broker consists of two sets of components that either operate on *data* or *control flows.*

Server vs. Client Brokers

The client brokers function as proxies of the client applications that run on one or more end hosts. The client brokers submit the requests for services — specified in terms of server IP addresses, transport protocols and port numbers — to the server broker on behalf of the client applications, make the purchase decision when the server brokers reveal the current prices of the services and conduct the transaction in order to establish passages for the traffic.

The server brokers, on the other hand, function as proxies of the server applications that provide specific services. The brokers determine the dynamic prices of the services based on several traffic parameters that are monitored continuously. They also work with the client brokers to conduct the payment transactions and control the client-server traffic flows.

Control vs. Data Flows

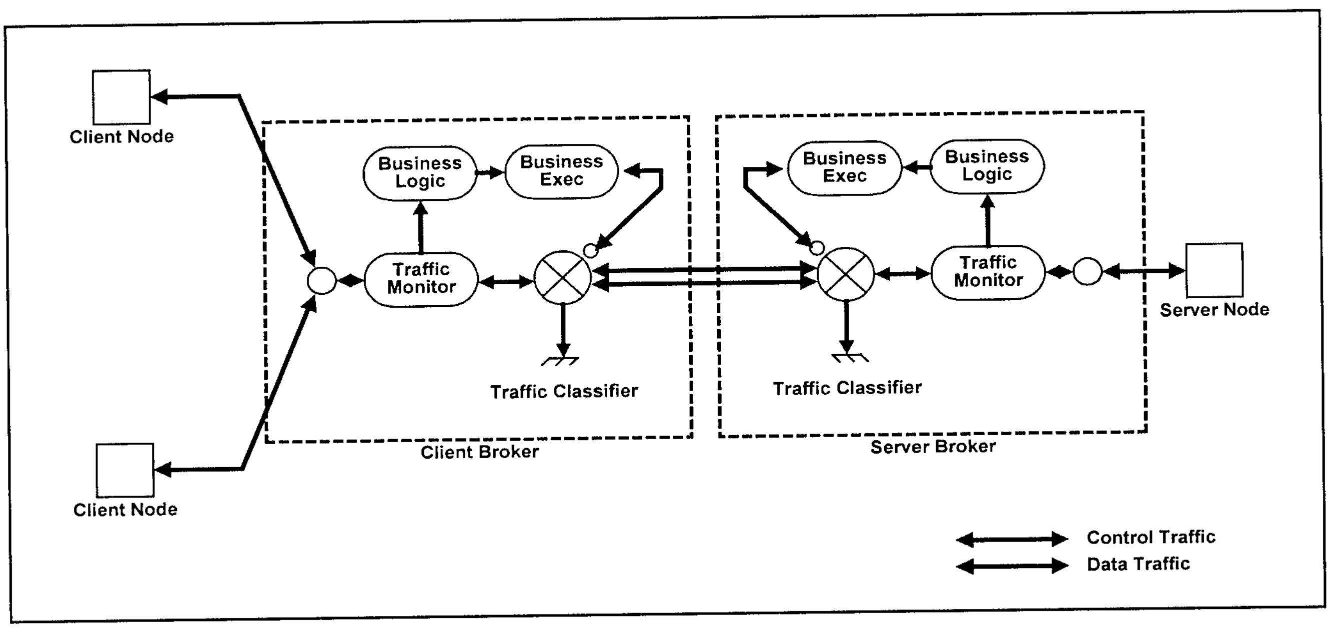


Figure 3-2. Functional Architecture of Resource Brokers

Each client or server broker consists of four components: *traffic classifier, traffic monitor, business logic* and *business executive* as shown in Figure 3-2. The business logic and executive establish the "control flows" of price/purchase exchanges and carry out the payment transactions. The traffic classifier and monitor work to control the "data flows" between clients and servers.

The functions of the four components are obvious from their names.

- *Traffic classifier* redirects IP datagrams according to the nature of their payloads so as to separate control and data flows. It also determines whether to pass or discard datagrams in the data flows based on the outcome of payment transactions.

- *Traffic monitor* provides the values of traffic parameters that are used to establish the current prices of specific services and/or serve as the indicators of anomalous traffic behaviors.

- *Business logic* is the decision-making component that computes the service prices in a server broker and makes the purchase decisions in a client broker.

- *Business executive* is the module that conducts the payment transactions using micro-payment protocols and controls the traffic classifier.

3.2. Business Logic

A business logic is a collection of rules and associated parameters that are used to control the traffic flow related to a service. It is designed to protect the resources such as network bandwidth or server capacity that a service deems important. Service providers must decide how to price and sell the services that they offer and then configure the business logics for each service. In MbSQD, services are distinguished by their server IP addresses, transport protocols and port numbers.

A client gains access to a service running on a specific server by purchasing a *subscription* to that service. Service providers are free to decide how to price the service — in packets, seconds or by connections. By carefully choosing the pricing algorithm and the units in which the subscription is sold, a service provider can manage the utilization of its service so that "good" client behavior is rewarded and "bad" client behavior is penalized based on some a priori definition of "good" and "bad". Note that the design of the business logic is independent of that of the payment mechanism. The current design of MbSQD can use scrip and proof-of-work computation as payment tokens.

The rest of this section discusses briefly the *subscription types* — i.e. how the resources are quantified and sold — as well as the *price* and *purchase decision functions.*

Subscription Types

Currently, four general subscription types are implemented in MbSQD.

a. *Subscriptions in Packet:* when subscriptions are offered to customers on a per packet basis, the service provider may define a maximum number of packets allowed for the client to send and/or receive. Once the client has met the quota, the subscription expires and the client must pay for additional service.

b. *Subscriptions in Seconds*: this type of subscriptions is sold in seconds of connection time. When the time interval has elapsed, the subscription expires and customers must purchase a new subscription. This time-based subscription may be used in conjunction with another subscription type to create a hybrid subscription type. For instance the subscriptions may be sold in terms of number of packets, but a client must send or receive the packets within a certain period of time. Such a hybrid type may be useful in the cases that a service provider intends to discourage its clients from "squatting" on a connection.

c. *Subscriptions in Connections:* a client simply pays for a single connection that lasts an indeterminate amount of time. This subscription type may be combined with the time-based subscription to simulate the leasing of the resource.

d. *Subscriptions in Bytes*: a client may also purchase a subscription based on the number of bytes sent to/from the server. The subscription expires once a client has exceeded the maximum number of bytes specified by the service provider.

Pricing Functions

The pricing strategy is the mechanism that the server broker uses to control resource consumption. Whenever a new subscription request of a specific service arrives at the broker, it invokes the business logic of the service to calculate a price for the new subscription based on the current values of the *market observables* [Sect.VI.A.3] defined for the service.

In our experiments, we tested the following four different forms of pricing functions:

Constant Function ($p = k$): the price p of the resource is set to constant k regardless of its level of consumption.

Linear Function ($p = kc$): the resource price p is proportional to the value of a chosen market observable c such as the number of open connections.

Asymptotic Function ($p = kB/(B-c)$): this function raises the resource price p to infinity as the market observable c approaches its limiting value B; such a pricing strategy is useful in safeguarding a resource with a hard limit in capacity.

Exponential Function ($p = k_1e^{k_2c}$): this function produces the fastest increase of resource price with respect to the increasing value of the market observable c; such a pricing strategy is useful in controlling consumption of a critical resource.

Each of these pricing functions produces a floating point number that can be translated into appropriate proof-of-work computation or scrip values.

Purchase Decision Functions

At the client brokers, interacting with the resource prices are the *purchase decision functions*, which determine whether to purchase the subscriptions by making required payments. The decision functions can employ sophisticated strategies based on the market observables and other parameters supplied by the clients. The simplest decision function, nonetheless, may specify a price ceiling for each client: the client broker purchases the subscription only if its price lies below the ceiling.

3.3. *Micro-payment Protocols*

MbSQD employs a three-message handshake to perform the payment transactions. The handshakes provide a framework protocol that can be used to support different forms of micro-payment including scrip and computational proof-of-work. It can also be made compatible with various micro-payment infrastructures.

Figure 3-3 shows the message sequence of the payment protocol. It uses three messages (subscription request, payment request and payment responses lines) to submit the service/subscription request and exchange the payment request and response. These messages can be readily integrated with the three TCP connection establishing messages with the service and the payment requests appear as options in the SYN messages from the initiator and the responder respectively, and the payment response integrated as an option into the ACK message of the initiator.

In order to support proof-of-work, two more messages (POW request and POW response lines) are added between the client broker and the clients. These messages complete a handshake that will be completed if and only if the proof-of-work computation is carried out successfully.

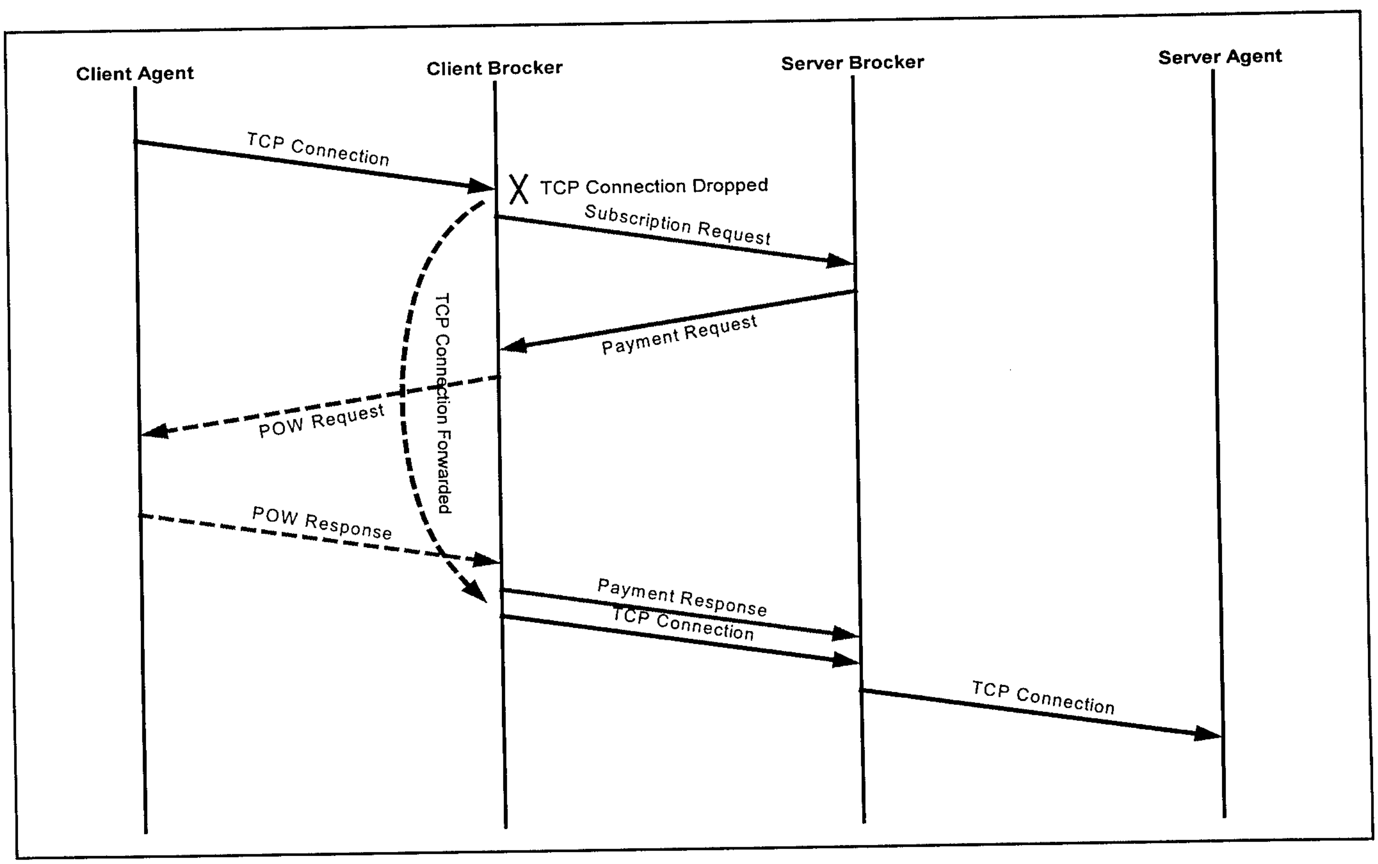


Figure 3-3. Message sequence of payment protocol

4. Experimentation using MbSQD Simulation

We investigated the behaviors of MbSQD broker architecture and traffic management mechanisms by conducting a series of simulation experiments using public-domain discrete-event network simulator, ns-2. Figure 4-1 shows the experiment configuration. In the experiments, a fixed set of legitimate clients was programmed to request the service offered by a single server. Their requests were mingled with much larger number of requests initiated by the rogue clients that were subverted to instigate DDoS attacks. The service requests of both "good" and "bad" clients were relayed by the client and server brokers deployed at the boundaries of the sub-networks that contain the clients and the server respectively. Both client and server brokers could operate in active or inactive mode. In the inactive mode, the brokers behaved like ordinary firewalls or routers. When they were activated, client and server brokers could control the traffic flowing through

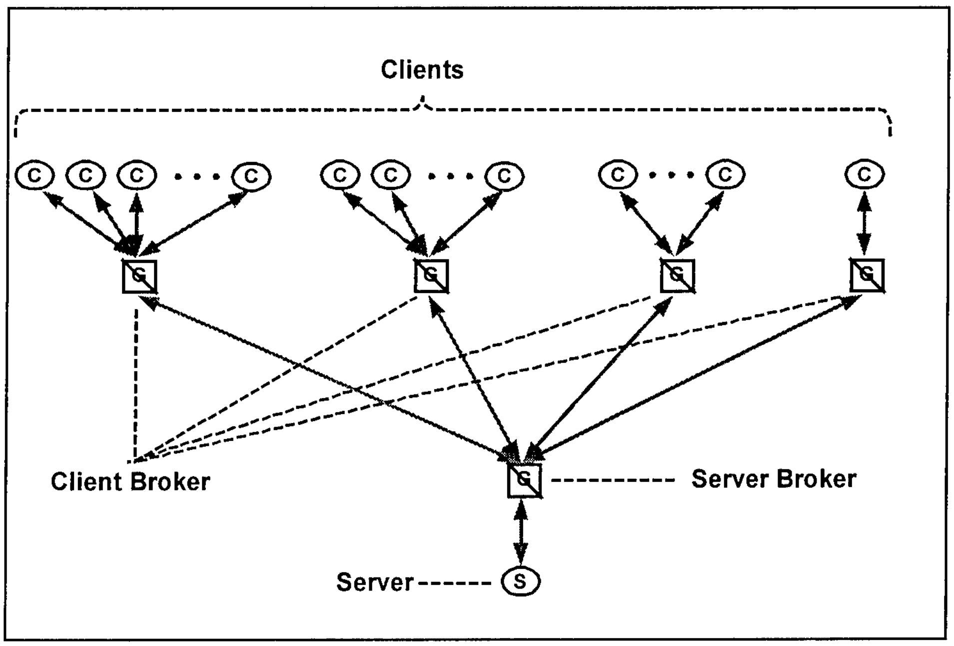


Figure 4-1. Configuration of Simulation

them by matching the IP datagrams with the connections established between the brokers. The datagrams were passed if they could be matched with one or more of the established connections and discarded otherwise. In each experiment run, we observed the progress of two DDoS attacks one with and the other without the use of MbSQD. Control parameters such as rogue client numbers and traffic characteristics were changed between different experiment runs.

4.1. Objectives

The primary goal of the experiments is to investigate the *effectiveness* of MbSQD architecture in mitigating the DDoS attacks. The *degree of effectiveness* was inferred from the following two sets of observations:

A comparison between the number of subverted clients required to launch two similar DDoS attacks that cause compatible levels of performance degradation with and without the activation of MbSQD brokers;

A comparison of the residual level of services available to the fixed set of legitimate clients in the two similar DDoS attacks with and without using MbSQD brokers.

A secondary goal of the experiment is to investigate the *usefulness* of price and/or other market parameters employed in MbSQD as indicators of presence/absence of possible attacks and/or measurement of the level of severity of the attacks.

4.2. Metrics

In order to establish a quantitative description of DDoS attacks, and an economic model for client-server traffic, we introduced three sets of parameters into MbSQD, known as *attack quantifiers*, *market observables* and *market controllables.*

Attack Quantifiers

The effect of a DDoS attack may be quantified by two sets of measurements:

Service Quality: as its name indicated, this measurement reflects the quality of a specific *service* received by a specific *client* or *group of clients.* Assuming web browsing is the service of interest and the DDoS attack aims at exhausting the number of active HTTP sessions maintained by the server, a suitable measurement for the service quality would be a pair of values consisting of the average latency for establishing a new HTTP session and the average data rate sustainable by a legitimate client under no-load condition.

Attack Duration: currently, a DDoS attack may last as long as the attackers intended. By introducing a payment scheme and providing each client with limited amount of credits, MbSQD established an inherent limit to the amount of service any one client may obtain before spending all its credits, and thus created a natural end to an attack. In the experiments, we monitored the progress of an attack by collecting measurements periodically throughout its course.

Market Controllables

In MbSQD, the trading of resources is controlled by the server broker through the setting of three control parameters:

Connection Granularity: the connections established by the server brokers may cover a group of clients and a range of protocols. Selectors such as source/destination IP addresses, transport protocol identifiers and TCP/UDP port numbers serve to specify the granularity of the connections.

Connection Duration: the connections established between client and server brokers could only last for a finite duration. This parameter specifies the duration of the connections established at the current moment. Note that the duration may be specified in real time or data size (either in number of bytes or packets).

Connection Price: this parameter refers to the time-varying price for a connection that is set by the pricing function maintained in the server broker.

Market Observables

The price of a resource was computed based on the current values of a selected set of parameters that reflect resource consumption. The choices of these parameters are determined largely by the nature of the resources and the strategy employed to manage those resources. The following three parameters were used in our experiments to determine the price of HTTP connections.

Connection Request Count, which records the total number of HTTP requests received by the server broker within a measurement period.

Connection Establishment Count, which records the total number of HTTP connections established by the brokers within a measurement period.

Data Throughput, which records the total number of data bytes passed to the server within the past second; the measurement is also computed as a percentage of the maximum data rate sustainable by the server and the data link between it and the server broker.

4.3. Results

We performed three sets of experiments that were designed to study the behaviors of MbSQD system in response to three different kinds of DDoS attacks:

1. *TCP-SYN Attacks:* in these experiments, the rogue clients flood the server with SYN packets with forged source IP addresses in order to overwhelm the server with half-opened TCP connections;

2. *Server Flooding Attacks:* in these experiments, the rogue clients flood the server with frequent and long TCP connections uploading large amount of data to the server; this set of experiments were also designed to examine the effects of using computational proof-of-work as a method of payments offered by the clients;

3. *Server Draining Attack:* in these experiments, the rogue clients initiate frequent TCP connections downloading large amount of data from the server (e.g. an HTTP server). This set of experiments also examines the effects of using fungible payments as a means of payments.

Experiments of type (1) and (2) were run with 25 legitimate clients requesting TCP connections of random exponentially distributed durations (average 0.5 seconds) separated by random exponentially distributed intervals (average 0.1 seconds). Each of the legitimate clients would establish a connection with the server and upload a relatively small amount of data. Experiments were run with from 0 to 4000 rogue clients. The rogue clients were identical to the legitimate clients except that they were more aggressive: their average interval was shorter (0.1 seconds) and the average duration of their data uploads was also larger (0.7 seconds).

Experiments of type (3) were run with 128 legitimate clients opening TCP connections and sending requests requiring a response of a random, exponentially distributed size (average 17,000 bytes) at exponentially distributed intervals (average 0.1 seconds) Experiments were run with from 0 to 512 rogue clients. The rogue clients loop continuously sending requests requiring random, exponentially distributed sizes of a larger average (average 100,000 bytes).

Mitigation against TCP-SYN Attacks

In these experiments, the rogue clients were programmed as constant bit sources, generating SYN packets as fast as they can without completing the connection establishment. The attack did not start until 0.5 seconds into the experiment, and stopped 1.5 seconds before the end of the test. Figure 4-2 displays the result of the experiment. The graph plots two traces: the number of packets from the legitimate clients that were delivered to the server agent while MbSQD was running verses the same measurement without MbSQD. The vertical axis shows the total packet count (over the course of the entire experiment). The horizontal axis is the ratio of Good Clients to DDoS attacking agents. As the graph shows, the intensity of the attack has little impact on the throughput of the legitimate clients when MbSQD is active; on the other hand, the service is effectively denied in the absence of MbSQD.

These drastically different results have a simple explanation. By requiring a proof-of-work response from the clients before the brokers would forward packets, the impact of the SYN attacks is shifted from the server to the gateways that host the brokers. A naive SYN packet flood has no impact on the server because the server broker discards all the attacker packets without burdening the server with them.

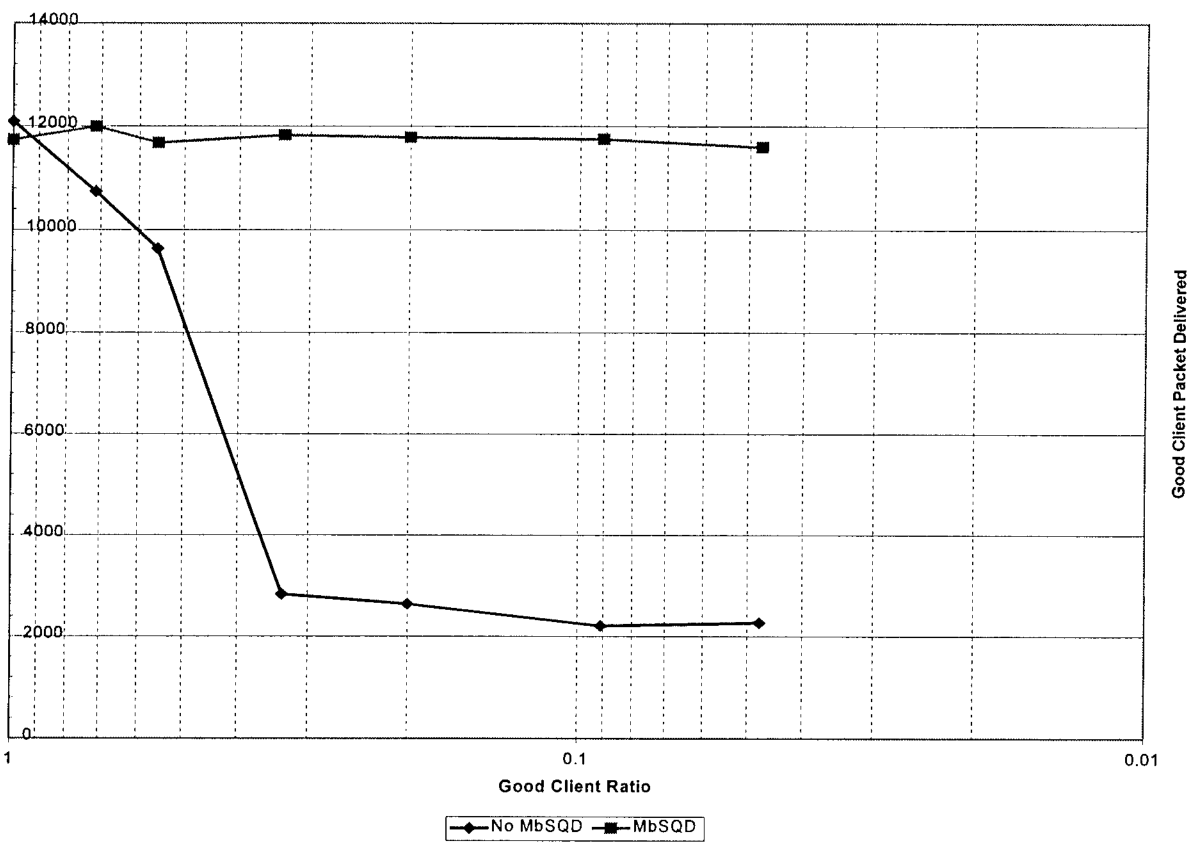


Figure 4-2. MbSQD mitigation against TCP-SYN attacks

Mitigation against Server Flooding

In these experiments, the rogue clients had similar but more aggressive behaviors than the legitimate clients: their average connection interval was set to be 0.01 second and their average duration was 0.7 second; in other words, the rogue clients requested connections much more frequently than the legitimate clients, and held onto them for a little longer. The resource protected by the server broker includes the server node, the server broker, and the link between them. In these experiments, the price was linearly proportional to the number of open connections at the server broker (and at the server). The token of payment was proof-of-work computation that the clients must perform in response to the challenges from the client broker.

We ran three sets of simulations: a control case without MbSQD action, one with the server broker charging a price for the establishment of each connection and the other one with the server broker charging a price for every pass of 32 packets. As a measure of service quality, we counted the number of legitimate-client requests delivered to the server during ten seconds of simulation time. (We also verified that the total number of requests delivered to the server remained at 10000, no matter how many rogue clients were present.) Figure 4-3 displays the results of the experiment.

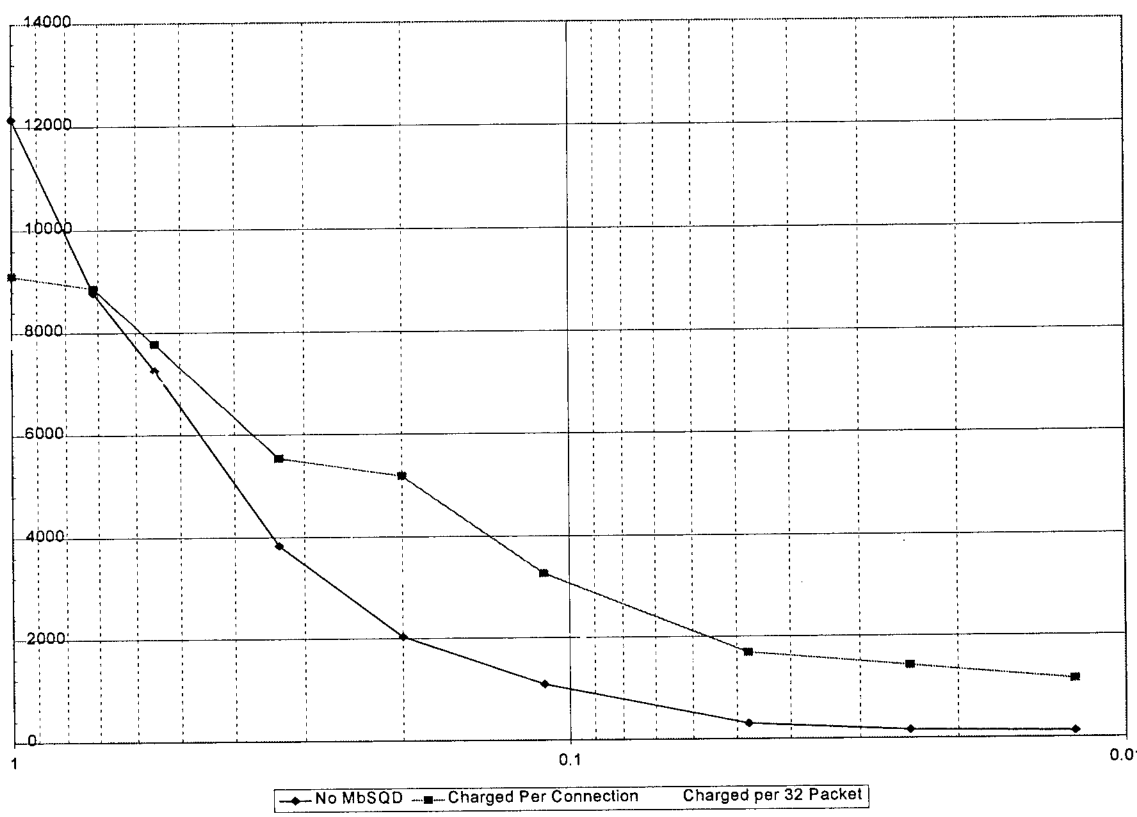


Figure 4-3. MbSQD mitigation against server flooding

The graph shows that all attacks were ultimately effective even with the use of MbSQD. However, it required eight times as many rogue clients in order to achieve the same level of service degradation when the brokers were active. Since our pricing strategy throttled both legitimate and rogue clients, the effects were not quite as dramatic as those of the first set of experiments, but it remains the case that an attacker must infect two to three times as many hosts in order to achieve and sustain a significant level of service degradation.

Mitigation against Server Draining

In this experiment, the rogue clients are attempting to act as a drain on a server's bandwidth. They attempt to repeatedly download large files, exhausting the number of connections available for legitimate requests (the size of their download is large simply to let them sit on the connection once they have it).
In this experiment, users don't buy connections, they lease them. The lease period was set so that, in an unloaded system, 90% of all legitimate tasks would complete requiring only one payment.

The fact that the effectiveness of this price function diminishes under heavy load comes as no surprise. As the load increases, a constant time lease covers less and less data. An experiment based on the amount of data transmitted (as opposed to connection duration)

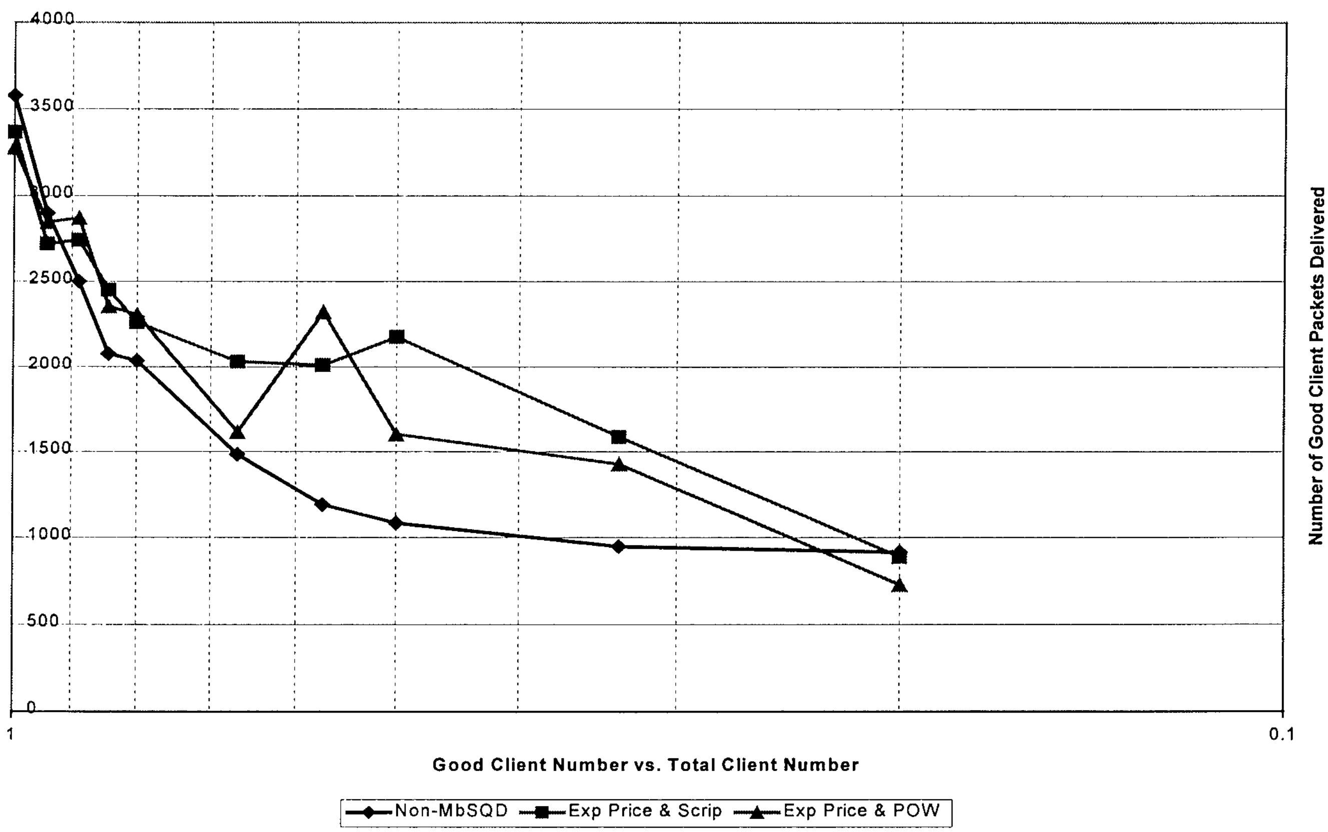


Figure 4-4. MbSQD mitigation against server draining

should prove to be more robust at the higher attack rates.

5. Conclusions

In this report, we have explored the application of dynamic pricing mechanisms in mitigating DDoS attacks. We have presented the MbSQD architecture and protocol that supports both proof-of-work and monetary-like micropayment schemes. We have prototyped the MbSQD system using the ns-2 simulator. We also presented simulation results on the effectiveness of different pricing strategies for some DDoS scenarios based on a monopolistic service model.

5.1. Lessons learned

We can make the following observations from the simulation experiments we have conducted:

- Pushing costs back onto clients appears to be effective for mitigating server-based DDoS attacks. Specifically, MbSQD does show promise for maintaining control of client-server traffic flows over the Internet. Traffic parameters can be implicitly exploited to maintain control even under changing conditions
- Proof-of-work methods are effective for elimination of spoofed requests or flooding via a limited number of machines. Scrip based payment methods can be effective if integrity can be maintained over the money supply.
- Different client behaviors can be discriminated by different server pricing strategies. Service Brokers can work to favor certain traffic behaviors in the range of scenarios we have studied.
- As to be expected, the choice of a pricing function will have a very strong effect on the effectiveness of MbSQD. Pricing functions can favor either the defender or the attacker and care must be taken to choose pricing functions that elicit behavior conducive to the accomplishment of the mission of the server(s) being controlled.

5.2. Suggested Directions for Future Research

The investigation presented in this report is preliminary in nature. We suggest at least the following two possible extensions of the current system.

Dynamic Subscription Parameters

Currently, subscription parameters types and limiting values are fixed throughout the process. However, a service provider might define a business logic in which lengthy subscriptions are offered when the service is not busy, but allowed length may shrink in response to service load. Alternately, a service might offer free service until its resources are depleted to a certain level, at which point the service becomes fee-for-use until utilization drops again. By the same token, a service might use a linear or constant pricing algorithm until some threshold of resource use was reached, using an exponential strategy thereafter.

Service Differentiation

Service providers may want clients to receive different levels of access to resources or pricing based on the service category to which they belong. Service category in this case is an abstract group defined by the service provider that might relate to factors such as the client's service use profile e.g., "well-behaved" clients vs. "poorly-behaved" clients or some out-of-band, cached relationship e.g., "AOL customers" vs. education domain clients. Below are a few possible ways that a service might employ to differentiate between clients.

Buyer's Clubs: the price that a client pays depends upon the last price category it was in. Hence, people who paid for premium service might get a price break the next time they renew their subscription, or might receive a price break if they purchase a large enough unit of service.

Threshold-based Packet-Dropping: service categories such as gold, silver, and bronze correspond to certain price thresholds and packets for a particular subscription are dropped whenever the price rises above the level of service that the client paid. For example, a client purchases silver service; whenever the price is 16 or greater?, packets are dropped at some known rate for this particular service category.

Queuing Manipulation: level of service corresponds to the delay that the client experiences in having its packets processed.

Subscription Parameter Modification: the level of service corresponds to a certain set of values for subscription parameters. Example: Subscriptions are sold by packet. If a client purchases gold service, it is are able to purchase 50 packet blocks; silver = 30 packet blocks; bronze = 10 packet blocks.
Further research is necessary to determine how to combine different dynamic pricing mechanisms in order to enhance overall system survivability by discriminating against different kinds of adversarial behavior. However, our preliminary results indicate that dynamic pricing strategies offer a promising new direction in countering server-based DDoS attacks on the Internet.

6. Bibliography

[Jue99] A. Juels and J. Brainard, "Client Puzzles: A Cryptographic Defense Against Connection Depletion Attacks," Network and Distributed System Security Symposium '99, San Diego, CA, USA, February 1999.

[Ful00] E. Fulp and D. Reeves, "A Multi-Market Approach to Resource Allocation," Proc. of Networking 2000, Lecture Notes in Computer Science, G. Pujolle, ed., No. 1815, pp. 945-956, May 2000.
[Bla98] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "RFC 2475: An Architecture for Differentiated Services," December 1998.
[Pos81] J. Postel, "RFC 793: Transmission Control Protocol," September 1981.
[Gib01] S. Gibson, "The Strange Tale of the Denial of Service Attacks against GRC.com", www.grc.com/grcdos.html", June 2001.
[Fis99] P. C. Fishburn and A. M. Odlyzko, "Competitive Pricing of Information Goods: Subscription Pricing Versus Pay-Per-Use," Economic Theory, Vol. 13, pp. 447-470, 1999.
[Niv73] L. Niven, "Flash Crowd," The flight of the horse, Ballantine Books, 1973.
[Bak99] Y. Bakos and E. Brynjolfsson, "Bundling Information Goods: Pricing, Profits, and Efficiency," Management Science, December 1999.
[Dwo92] C. Dwork and M. Naor, "Pricing via Processing or Combating Junk Mail," in Ernest F. Brickell, ed., Advances in Cryptology, Crypto '92, Vol. 740, Lecture Notes in Computer Science, pp. 139-147. Springer-Verlag, 16-20 August 1992.
[Riv97] R. Rivest and A. Shamir, "PayWord and MicroMint: Two Simple Micro-payment Schemes, " Lecture Notes in Computer Science, vol. 1189, Proc. Security Protocols Workshop, Springer-Verlag, pp. 69-87, 1997.
[Jak99] M. Jakobsson and A. Juels, "Proofs of Work and Bread Pudding Protocols," In B. Preneel, ed., Communications and Multimedia Security. Kluwer Academic Publishers, pp. 258-272, 1999.
[Cha83] D. Chaum, "Blind Signatures for Untraceable Payments," Advances in Cryptology--Crypto '82, Springer-Verlag, pp. 199-203, 1983.
[Tsu95] T. Tsutsumi and S. Yamaguchi, "Secure TCP -- providing security functions in TCP Layer," Proc. INET '95, pp. 905-913, June 1995.
[BBC01] http://news.bbc.co.uk/low/english/sci/tech/newsid 1348000/1348820.stm
[CAIDA01] http://www.caida.org/outreach/papers/backscatter/
[Sno01] A.C. Snoeren, C. Partridge, L.A. Sanchez, C.E. Jones, F. Tchakountio, S. T. Kent, and W. T. Strayer, To appear in Proc. ACM SIGCOMM 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, August 2001.
[MPAPI] W3C Micro-payments API and Markup Working Groups, http://www.w3.org/ECommerce/Micro-payments/
[MPTP] Micro-payments Transfer Protocol, http://www.w3.org/TR/WD-mptp-951122
[Milli] Compaq Millicent, http://www.millicent.digital.com/
[Pay] PayPal, http://www.paypal.com
[NS2] NS-2 simulator, http://www.isi.edu/nsnam/ns/index.html
[Arq] The ARQoS Project, http://arqos.csc.ncsu.edu/
[Free] The FreeHaven Project, http://www.freehaven.net/
[Digi] DigiCash, http://www.digicash.com
[Hash] A. Back, "Hash Cash: A Partial Hash Collision Based Postage Scheme," http://www.cypherspace.org/~adam/hashcash.
[**Mojo**] Mojo Nation, http://www.mojonation.net/

Appendix A – MbSQD Simulation Model Documentation

Introduction

The MBSQD experiment software consists of:

1. A series of TCL scripts to be loaded into **ns**, defining MBSQD node-types and other objects.

2. A series of TCL scripts that actually define the experiments (and which make use of the new **ns** objects defined by the scripts in (1).

3. A series of shell-scripts to run the experiments and process the resulting data.

This software is not really ``user software"– it is a set of research utilities that can be used to analyze the behavior of the MBSQD architecture. As such, it's "tinkerer software".

Other software used

The scripts used by MBSQD use the following software, which may not already be installed on your UNIX system:

ns, the NS-2 network simulator, available from http://www.isi.edu/nsnam/ns/.

gnuplot, the GNU plotting program, available from http://www.gnuplot.org/.

gzip and **gnuzip**, the GNU data-compression (and de-compression) programs, available from http://www.gzip.org/.

Running an experiment (overview)

The easiest thing to do is to run an experiment using the command **run-connect** in the *mbsqd/scripts* directory.

run-connect expects you to be in the *mbsqd/src/scripts* directory, since it expects to find a connect subdirectory containing some configuration and data files used to direct the experiment's actions.

run-connect also expects an **NS_DIR** environment variable to be set to the directory where the **ns** executable is located.

run-connect runs several iterations of the **connect.tcl** script, each iteration with different configuration parameters, and it stores the output of each iteration in a clearly-defined, uniquely-named file.

The output of the experiment is then processed using the script **process-connect**, which uses the same configuration files that drive **run-connect** to digest the output logs, producing **gnuplot** graph directives in the subdirectory *connect/plots*.

process-connect takes as its single argument the date-prefix of the data generated by the run of **run-connect** that you are interested in. (Once you're familiar with how the date-prefix is formed, it is easy to figure out the date-prefix from a given run, otherwise you'll have to look in connect/data to see what prefix is used on the files you are interested in.

The date-prefix has the form:

<yy>-<mon>-<dd>-<hh>   e.g.,

01-Jun-20-14
01-Jun-21-19

These date prefixes are used to identify the output of a given run, allowing one to retain and examine historical data, and to associate the data with the resulting plots.

The output of **process-connect** is a set of files in *connect/plots* subdirectory. These files also all share the same date-prefix as the data used to generate them. The -process file conta'ns some log output; the -plot file contains directives to be invoked from within **gnuplot**, as is done with this script from 13 June 2001:

```
[dm@a plots]$ gnuplot

        G N U P L O T
        Linux version 3.7
        patchlevel 1
        last modified Fri Oct 22 18:00:00 BST 1999

        Copyright(C) 1986 - 1993, 1998, 1999
        Thomas Williams, Colin Kelley and many others

        Type `help` to access the on-line reference manual
        The gnuplot FAQ is available from


        Send comments and requests for help to
        Send bugs, suggestions and mods to
```

```
Terminal type set to 'x11'
gnuplot> load '01-Jun-13-10-plot'
gnuplot>
```

process-connect works by running **grep** over the log files (written by the *mbsqd-log* TCL routine) to pick out significant events.

run-connect produces a lot of status output as it runs:

```
[dm@a scripts]$ ./run-connect
-------------------------- mbsqd-packets 0 exponential  Fri Jun 22 14:39:48 EDT 2001
making connect/data/01-Jun-22-14-mbsqd-packets-exponential
Fri Jun 22 14:39:48 EDT 2001
Using exponential price strategy
NO restrictions on the maximum number of active connections
Creating 32 client nodes, each with 4 agents
Creating 0 DOS nodes, each with 0 agents
0.196848: FullTcpAgent::recv(_o1165): no SYN for our SYN(1)
0.205077: FullTcpAgent::recv(_o1234): no SYN for our SYN(1)
0.344800: FullTcpAgent::recv(_o1403): no SYN for our SYN(1)
0.347039: FullTcpAgent::recv(_o1394): no SYN for our SYN(1)
0.495879: FullTcpAgent::recv(_o1516): no SYN for our SYN(1)
0.598456: FullTcpAgent::recv(_o1608): no SYN for our SYN(1)
...
```

This is largely ignorable. The real output from **run-connect** will be found in the *connect/data* sub-directory, and will consist of files with names like:

01-Jun-22-14-mbsqd-packets-exponential
01-Jun-22-14-mbsqd-seconds-exponential

The structure of these names is as follows:

<date-prefix>-<strategy>-<pricing scheme>

Where <date-prefix> was described above.

Experiment configuration files

run-connect is driven by several configuration files, all located in the connect subdirectory.

dos-populations

Consists of a list of denial-of-service agent populations (e.g., 8 DOS agents, 16 dos agents, etc.). Each agent population should be on its own line (though this is not necessary), and a # (hash-mark) in the first column indicates that the line is a comment.

dos-populations

This is the *dos-populations* file used by **process-connect**. Having a separate file permits one to process the output of one run while another run is taking place.

N-config

It is perhaps a short-coming that each DOS-population, N needs a corresponding *N-config* file in *connect/config*. This file contains a lot of legacy boilerplate and also two lines specifying the number of *DOS_NODES* and *DOS_AGENTS* on each node.

DOS_NODES times *DOS_AGENTS* should be N.

strategies

Contains a list of ``strategies" to employ in the experiments, one strategy per line. A line beginning with a # (hash-mark) is a comment.

Strategies are basically just strings that are passed in as arguments to the **connect.tcl** script. To figure out what each strategy means, or to define new strategies, you'll need to refer to that script.

prices

Contains a list of price ``strategies" to employ in the experiments, one strategy per line. A line beginning with a # (hash-mark) is a comment.

Like the names in *strategies*, the names in *prices* are given meaning by the *connect.tcl* script.

Writing your own experiment

The file **connect.tcl** is an example application that uses the modules defined by the MBSQD project.

First and foremost, it sources the MBSQD tcl package by looking in one of two ``standard" places:

```
# Load the mbsqd software
if [ file exists ../tcl/ns-mbsqd.tcl ] {
   set MBSQDDIR ../tcl
} else {
   set MBSQDDIR /usr/mbsqd/tcl/mbsqd
}
source ${MBSQDDIR}/ns-mbsqd.tcl
```

This loads all the necessary definitions.

Later, some gateways are defined with their brokers:

```
set server_gw [$ns node]
[$server_gw broker] setFlavor "server"

set client_gw [$ns node]
[$client_gw broker] setFlavor "client"
```

ns-mbsqd.tcl has redefined the node initialization so that creating a new node actually creates a MBSQD node, which can be equipped with a broker. Above we have created server and client brokers.

A server broker is not complete without a business-logic (though there is a default business logic:

```
set bl [createNewBusinessLogic [$server_gw broker]]
[$server_gw broker] setBusinessLogic $bl
```

We also create a server host node on which the server will reside:

```
set host [$ns node]
[$host broker] setFlavor "null-log"

[$server_gw broker] setServerHost $host
[$client_gw broker] setServerHost $host
```

The ``null-log" flavor of broker is just a pass-through broker, that does not do anything to packets. null-log brokers are good for server hosts and client hosts.

And the necessary links to join them:

```
set link [$ns duplex-link $server_gw $host \
    $HOST_LINK_SPEED $HOST_LINK_DELAY RED]

set link [$ns duplex-link $client_gw $server_gw \
    $LINK_SPEED $LINK_DELAY RED]
```

connect.tcl then uses its *create_node* routine to create a number of agents (either legitimate agents or DOS agents). These agents are created using the *create_connection* routine (defined in **ns-mbsqd.tcl**.

The critical line is

```
set bl [createNewBusinessLogic [$server_gw broker]]
```

createNewBusinessLogic is defined in **config-sbbl** in the same directory as **connect.tcl**. The commands in **config-sbbl** build an array *SINFO* containing configuration information for the business logic. The array is initialized from global variables defined in experiment configuration files, but one could populate the array in other ways.

Appendix B – References to the "ns-2" Network Simulation Model

NS-2 is an extensive network simulation environment originally funded by DARPA and developed by researchers from:

AT&T Research
Lawrence Berkeley National Laboratory
UC Berkeley
USC/ISI
Xerox PARC

It is a well-written, powerful collaborative simulation environment with an active research community.

Source code, manuals, and various tutorials are available at:

http://www.isi.edu/nsnam/ns/

An interactive workshop on ns is available at:

http://www-aml.cs.umass.edu/~ns/index.html

NS-2 simulations are written in both C++, which is used to develop new modules, and Object TCL (O-TCL), which is used to script system operations. A good tutorial on O-TCL is available at:

ftp://ftp.tns.lcs.mit.edu/pub/otcl/doc/tutorial.html

Part II – Cartography of Cyberspace

1. Introduction

As noted in the preface to this report, the work on the prime charter of the Cartography of Cyberspace task was terminated after only six months (out of the original 27 month effort) due to changes in the focus of the overlying IASET program. For better or worse, we had taken to heart the admonition given during the IASET PI Kickoff meeting in April 2000: "Be daring, attempt to push the limits of what we know, and have fun". We realized in setting out that our goal was very ambitious, but we had approached the effort with some pluck and enthusiasm and (if we can continue the "cartography" analogy for a bit), we were not unlike team of adventurers going deeply into uncharted territory – only to find out en route that funding was eliminated back home and that the logistical supply drops we were counting on weren't going to occur after all. This left us with the dilemma of either pushing forward towards the original destination with the remaining funding, hoping to make it there successfully (and that somebody would care if we did); or taking the more prudent course of regrouping, choosing a new destination in line with current sponsors, and attempting to make progress towards that new destination. After some reflection, we chose the latter course. Market-based Service Quality Differentiation (MbSQD) was that revised destination, and our progress has been documented in Part I.

What we have done here in Part II, is to attempt to preserve some of the "diaries", "sketches", and "crude maps" that we had been constructing along the initial journey. The research results presented here must be considered preliminary in each of the thrusts – however we'd like to think that some useful progress was made. The intent of this part is to document the effort that was spent to allow others that may eventually hope to pursue a similar destination the opportunity to glean some useful results from the work as it stands.

Overview

The premise of the Cartography of Cyberspace task was that structures and interactions in cyberspace could be modeled very much like other complex man-made and natural systems. By codifying the interactions and structures one can provide a common framework and increase the measurability of some of the currently vague notions of Information Security. The Object-Oriented (OO) design language UML (Unified Modeling Language) was proposed as a tool for modeling these structures and interactions.

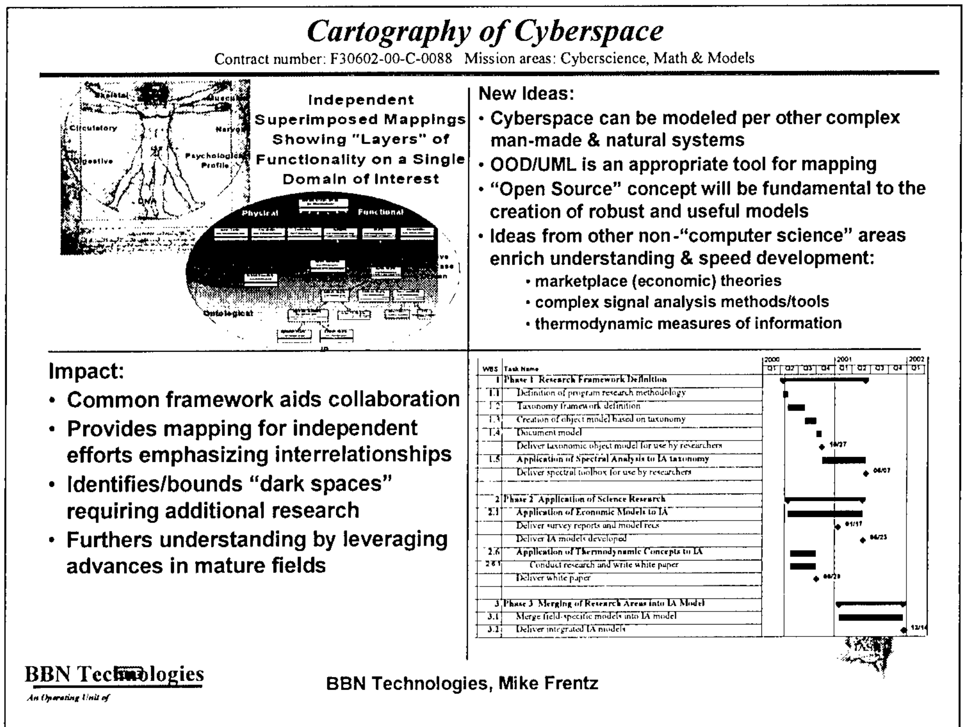


Figure C-1 Cartography of Cyberspace Quad Chart

One of the underlying assumptions in this approach is that there exist many analogies between phenomena occurring in "cyberspace" and those that occur in the real world. Several parallels were proposed between observed phenomena occurring in cyberspace and concepts from existing sciences. Most of these sciences for comparison were proposed as options to the effort. The base selection chosen for pursuit was the economics area. A comparison with the principles of thermodynamics was also selected as an option by DARPA.

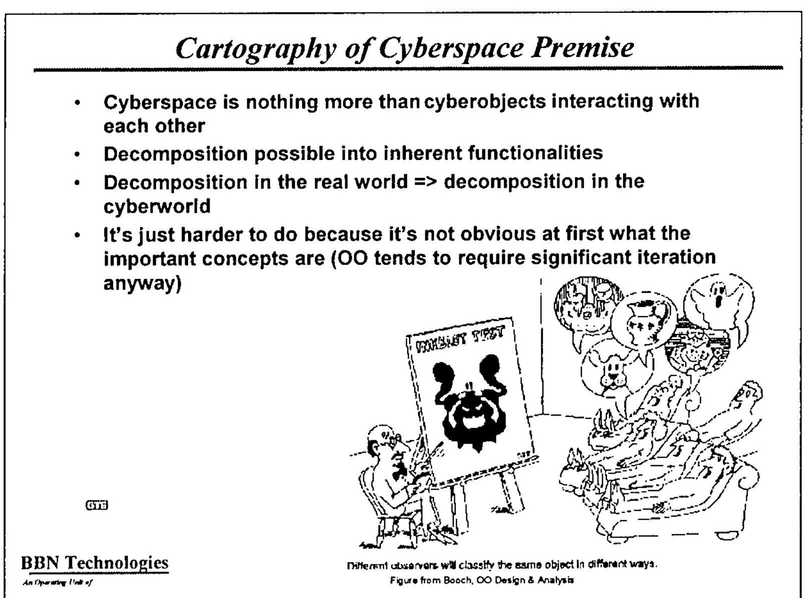


Figure C-2 Premise of Cartography Task

We made four public (program-wide or working group-specific) presentations during the course of this effort:
On April 25, 2000 at the IASET Kickoff meeting we presented our initial approach
On July 18, 2000 at the Cyberscience workshop of the Joint PI meeting, we gave a position paper on cyberscience, including some observations on the perspectives and prejudices that were predominant during the evolution of what are now regarded as traditional sciences (specifically physics, chemistry, and biology).
On July 20, 2000 at the UML workshop of the Joint PI meeting we participated on a panel on the use of UML and presented a demonstration of an interactive HTML-based UML model accessible by a large user base without special software.
On August 31, 2000 at the Workshop on Macroscopic Phenomena we presented a presentation on the layered worldview (that was well-received).

A summary of the work performed in each of the dominant thrusts of the Cartography of Cyberspace task follows.

2. Taxonomy

2.1. Technical Approach

The technical approach for the taxonomy effort was to:

- Develop a visually based extensible research framework – a visual taxonomy – to enable the mapping of security concepts to network implementation
- Classify pre-existing sets of vulnerabilities from community-wide data bases (e.g. CERT, CVE, BugTraq) using the developed taxonomy
- Make this framework available to the IA&S community via the web for extension, collaboration, and criticism

One of the goals of this task was to develop a visually-based extensible research framework to enable collaborative investigation into the security aspects of networks and the mapping of security concepts to network implementation. A taxonomy is not a neutral structure – its organization implicitly creates a theory for the problem space and will determine what data gets collected and what questions can reasonably get answered. Therefore by structuring a taxonomy appropriately, one could conceivably encourage or discourage the codification and measurement of security-relevant notions (such as authentication, confidentiality, integrity) that, at present, are not very well-conceived in the minds of most non-security professionals.

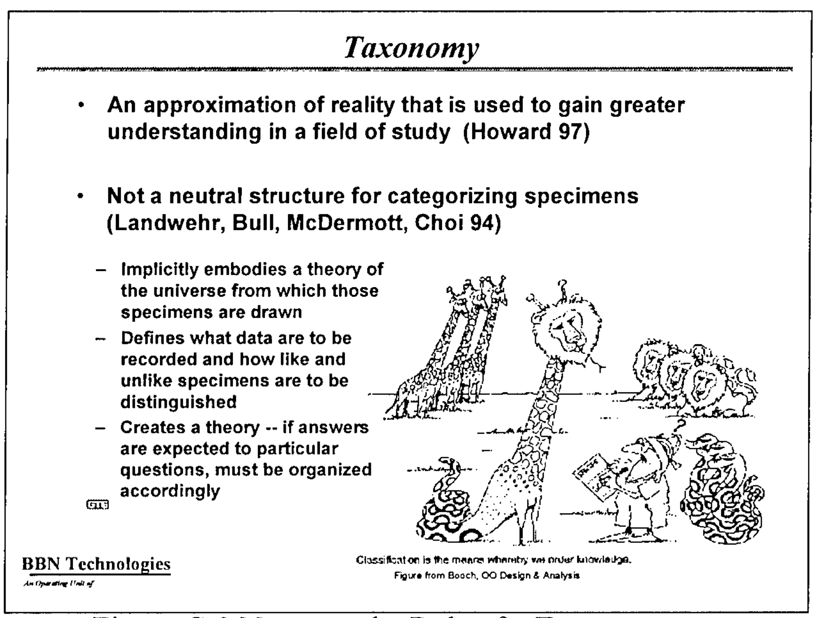


Figure C-3 Notes on the Role of a Taxonomy

It is possible to model networked information systems from multiple abstraction perspectives (layers), e.g.:

- Physical layer (Structural component interactions)
- Functional layer (Abstract Functional decomposition)

- Operations related
- Security related
- Transactional layer (actors: accessors, accessees, cooperators, values)
- Ontological layer (information dimensionality)
- Development of Use Case Diagrams (interaction motive/course of action).

UML, the Unified Modeling Language, has emerged in the last couple of years as the predominant OO "blueprint" language in order to map relationships between abstract programming concepts. We began by formulating the concepts that composed security in a system under a UML framework. We developed a simple model of a specific security instance (e.g. availability) and began to parse it into the specific required functionality and the corresponding constituent components. This was developed further as an interactive UML model that could be interactively explored via a web browser.

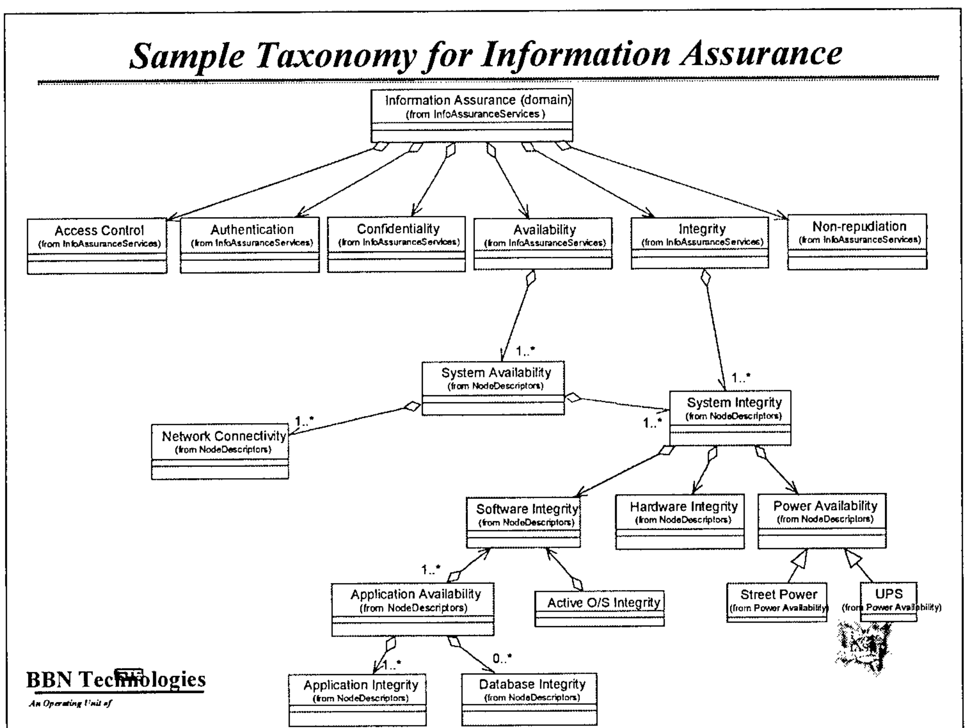


Figure C-4 Sample Top-level UML Taxonomy for IA

The idea with an interactive taxonomy is that an analyst can start with a very high level description of a complex system, and tunnel down through various layers and perspectives of system composition to understand the underlying structure and interactions and thus the vulnerability. Just as a contractor about to begin modification to an existing complex structure may require access to structural, electrical, plumbing, and ventilation diagrams (each available in various levels of detail or component diagrams) in order to facilitate his job, the linking of HTML-based UML models shows promise in allowing a security analyst, systems administrator, or even an end user to examine the design characteristics of a system in detail. We used both Rational Rose and MS Visio to publish these diagrams – though neither ultimately turned out to be the ideal tool (the diagrams could be better organized using Rational Rose, however the creation of embedded HTML links in the diagram was greatly facilitated with Visio).

The interactive version of this preliminary taxonomic model that permits "drill-down" into security conceptual, hardware, and user case functions can be accessed at https://archive.ia.isotic.org/dscgi/ds.py/View/Collection-572. Printouts of the various screens for the sample model are shown below. Yellow highlighting on a box signifies that it is *active* – clicking on it will give more detail (in a "fully populated" version of the model, this wouldn't be necessary since virtually every box would be active). Red highlighting indicates that this box is the "home state" for the particular view shown.

With a map such as this, an analyst or user can interactively and visually probe a complex network of systems, viewing the networked infrastructure from various perspectives and viewing potential attacks on the infrastructure as well as causal relationships between operating characteristics (e.g. availability and the respective conceptual infrastructure and the relations to specific components comprising the infrastructure. These maps, individually supplied from each vendor/facility/etc. can be aggregated, with reuse of common infrastructure components and configuration options, to allow the development of vulnerability libraries that can be reapplied to specific networked information systems.

At the lowest level of the availability model, specific functional tests can be conducted (e.g. validating application integrity) to verify that the constituent functional components of "availability" are intact.

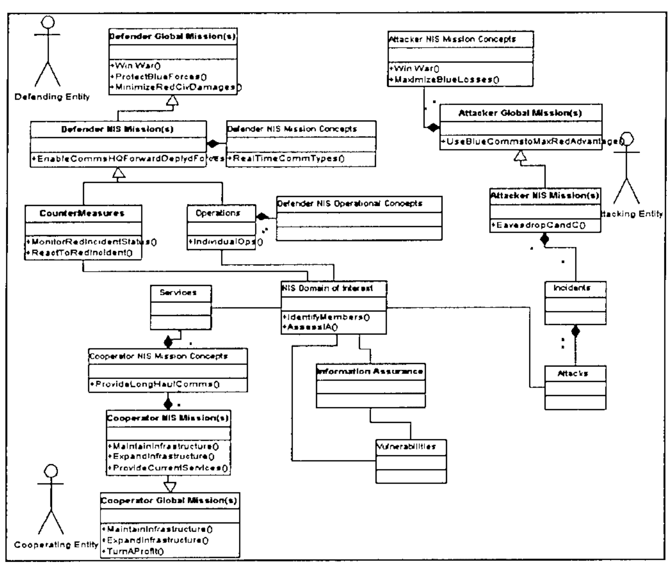


Figure C-5 Global View of Interactive Taxonomy

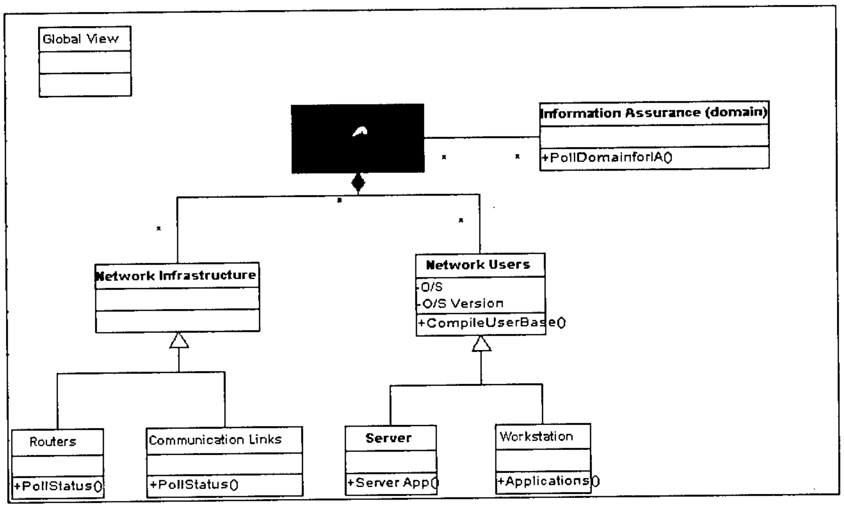


Figure C-6 Composed Hardware Domain View of Interactive Taxonomy

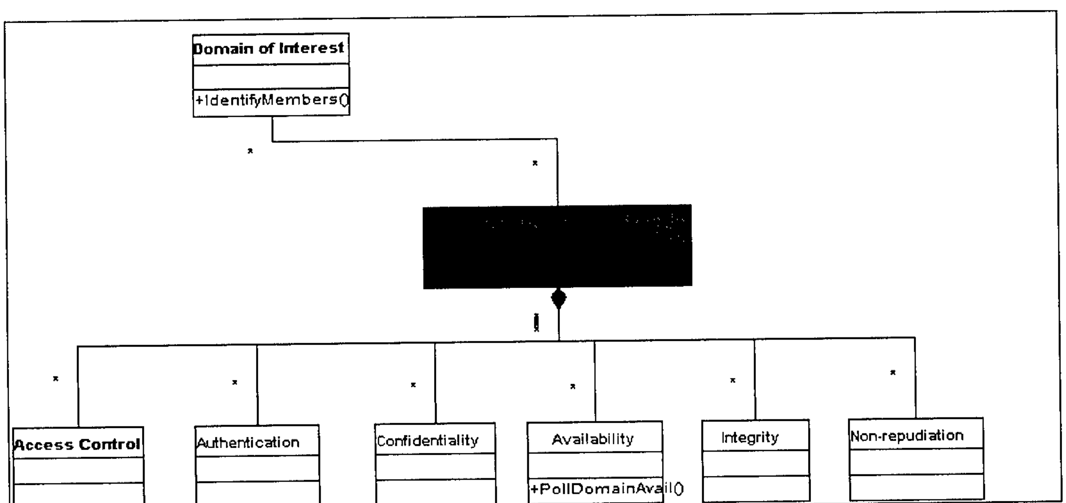


Figure C-7 Information Assurance Functional View of Interactive Taxonomy

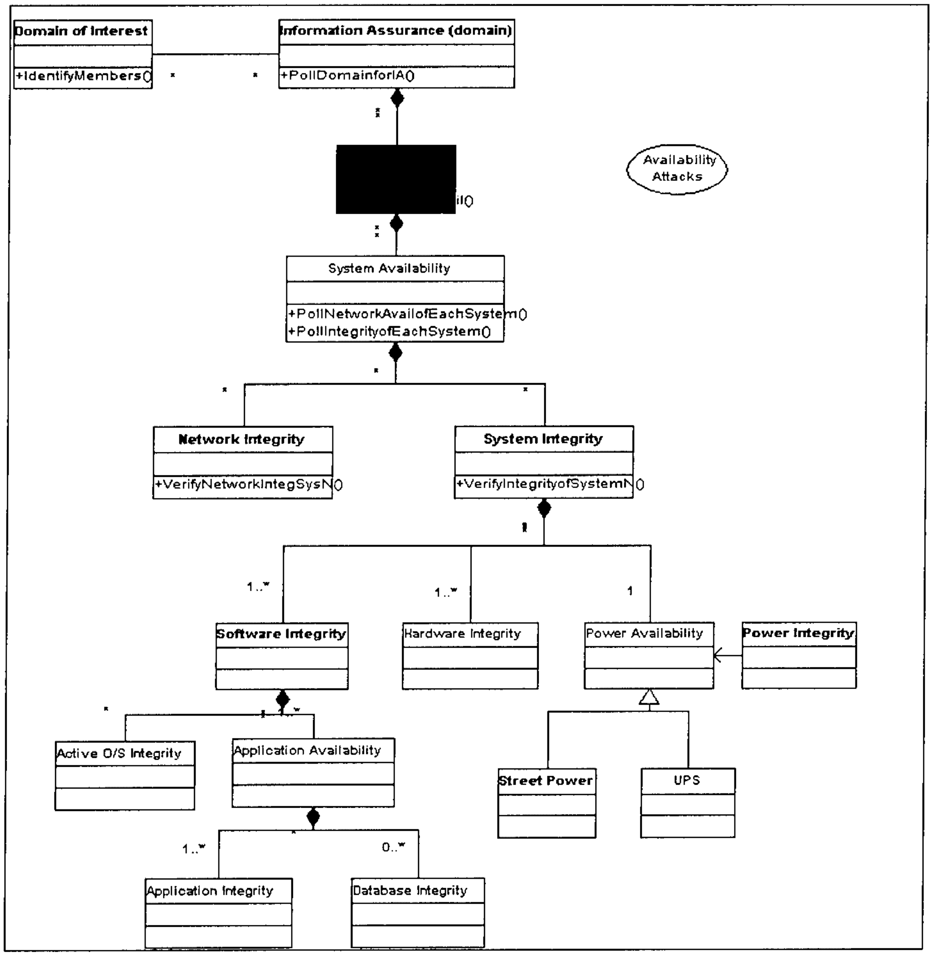


Figure C-8 Availability Functional Breakout View of Interactive Taxonomy

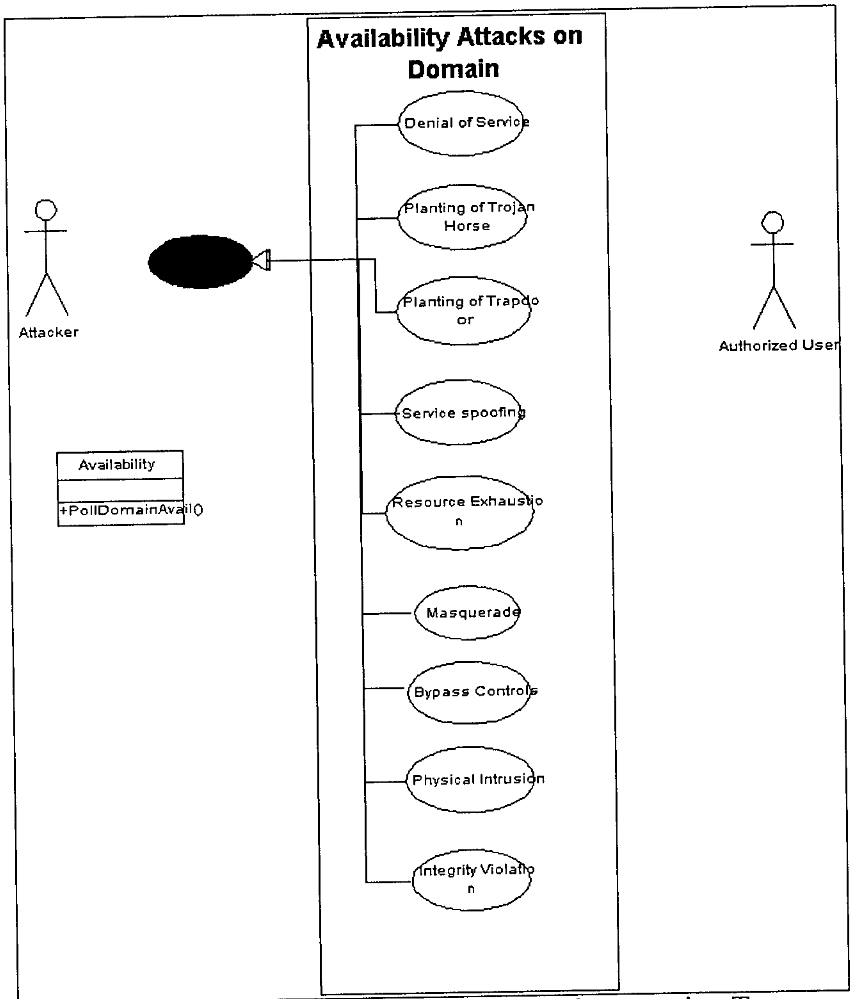


Figure C-9 Availability Attack Synopsis View of Interactive Taxonomy

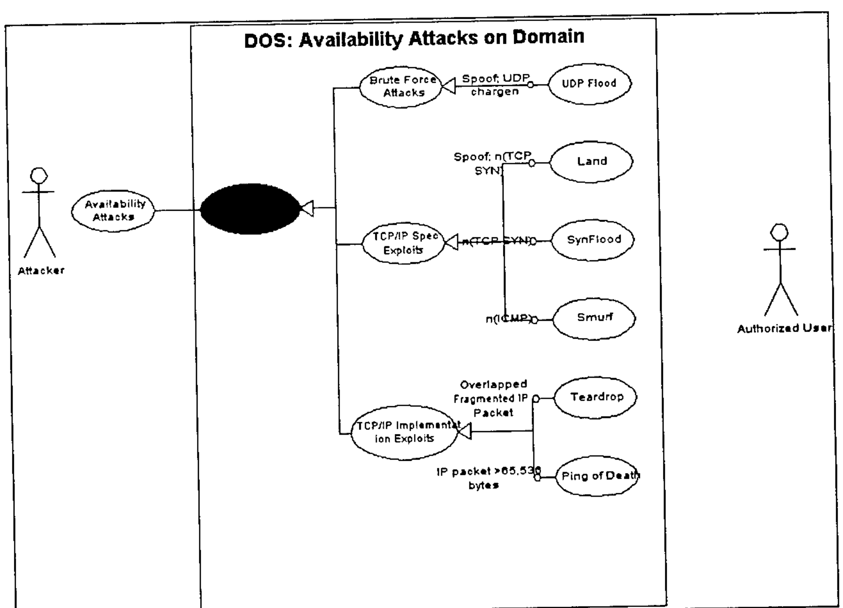


Figure C-10 Denial of Service Availability Attacks View of Interactive Taxonomy

Taxonomy Development

One of our early steps was to research published taxonomic efforts in order to leverage existing work as well as to examine the consistency of the models. There are numerous taxonomies that attempt to classify security-oriented phenomena such as system vulnerabilities and attacks. We reviewed and assimilated about a dozen prominent taxonomic papers that were relevant to security. These references are included in the bibliography section.

While each of the taxonomies presents a valid specific decomposition for a particular sub-area, there also tends to be disjointedness in the nature of the concepts categorized. There tended to occur a generalized mixing of terms, concepts, and goals from different logical and physical system layers and from different problem domains. Instead of describing a system in any single domain – such as the steps to be carried out for the successful accomplishment of a mission – the terms used would often slice through multiple dimensions: mixing intent, specific system characteristics, and security concepts. These analyses often resulted in a feeling that the taxonomy being provided was incomplete – no single dimension seemed adequately covered.

Security Premise

During the course of our analysis, one point was driven home to us repeatedly:

Security itself is never a *mission*, but rather is *always an adjunct* to accomplishing a mission.

The primary user in the field *does not care about security per se* – they only care about the mission. Security, itself, does not add *any* value to a mission. In the absence of an adversary (i.e. a credible risk) and, given a well-engineered system, security is rather always a *cost burden* to the end user. A corollary that results from this is that the prime user will always prefer to do without security it if they don't perceive the benefit to outweigh the cost.

This concept is essential to security and it is one of the key issues with respect to security not being applied properly – or at all at times – despite the fact that most everyone involved in the design or usage of a system may technically "know" better.

One of the difficulties in developing a taxonomy is that security is fundamentally a "negative" discipline. Its goal is to prevent something occurring rather than to accomplish something on its own, and therefore a taxonomy is difficult to describe in terms of a checklist of sufficient terms. In many ways, security is more akin to a safety discipline (e.g. automotive safety, nuclear safety) or the insurance industry than a typical "positive" engineering discipline. Security only adds value in the event that something goes wrong that will prevent a mission from being accomplished effectively (or would similarly contribute to the failure of some other mission). Security is always peripheral to the primary mission for which a system exists and it always adds cost in one or more dimensions, such as: speed, weight, usability, or actual cost. A user base – especially one under performance pressure – will prefer to do without security if they do not perceive a favorable cost/benefit tradeoff in the accomplishment of their mission. The only valid cost model for security is one similar to that of the insurance industry – *only enough security to adequately compensate for an unacceptable mission risk is appropriate* (and will therefore have a chance of consistently being used under operational conditions). Continuing the analogy further, the automotive safety and insurance industries only evolved significantly after the assets they were protecting were:

- Mature (in common use)
- Involved significant losses in terms of either lives or property
- Became financially viable, either through the development of a business model that made it attractive to third-party suppliers or were mandated through a combination of market and legal forces.

What makes security a prime issue today is that the risk involved by the predominant use of commercial ("COTS") hardware and software comprising our information systems is not commensurate with the military application of these systems. Most commercial applications have not traditionally involved both the potential for grave damage and the motivation for causing that damage, as is commonplace for a military application.

The reason for emphasizing this idea will be apparent in the next section.

Security Composition

By characterizing a system independently in several domains, it may be possible to develop a "complete" description of a system in each domain, and then (hopefully) be able to map across these domains to yield the linkages between mission and vulnerabilities. This would appear to be an antidote for the unsatisfying incompleteness of the published taxonomies - where items from different genres are combined without any completeness within any of the domains. We proposed that it is necessary to view a system in terms of three distinct but interrelated worldviews – mission oriented, functionally oriented, and implementation based.

The mission layer of the system conceptualization worldview expresses the system functionality totally in terms of the end user language and concepts. Again, stressing the point above, the mission layer worldview is the sole motivator for "security". If a mission can be accomplished with a very high probability, or conversely if the mission is not critical in some context, the security was by definition adequate.

Figure C-11 illustrates a parsing of a system at the very highest level into three discrete *worldviews*. Each proposed worldview expresses the system concepts in the needs and language of a specific audience

Mission ==> End User
Functional ==> Software/System Engineer

Implementation ==> Systems Admin/Operations Personnel

In order to actually build a system, it is necessary to provide a translation from concepts in the end user domain to generic functional descriptions of actions to be taken in a domain relevant to the system. In the formal system development world (e.g. Mil Spec 2167A or Mil Spec 498), these mappings take the form of system and software requirements specifications that provide for explicit trace-through from requirements in the user's external domain to implementation in code.

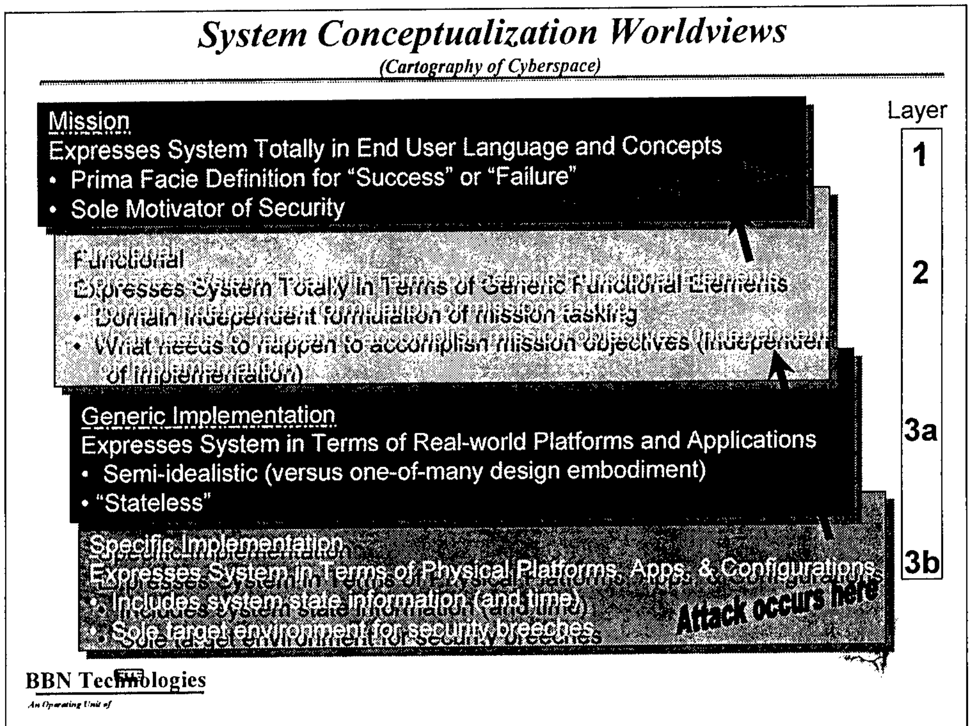


Figure C-11 System Conceptualization Worldview Summary

The worldviews provide an interesting perspective on security flaws – while attacks always occur in the third worldview, they are only felt and are measurable (i.e. *are significant*) in the first worldview. If the user base can consistently accomplish its mission successfully, the attack is by definition benign. The lack of traceability across worldviews is a primary contributor of system insecurities. The mission layer provides the only authentic and useful definition for security.

The key point is that the concept of *security* (whose relevance is entirely mission-specific) is separate from the *attack mechanisms* (which are purely a function of system architecture and implementation and therefore largely driven by commercially available products) at a taxonomic level.

Re-expressing system concepts in a layered worldview may allow "locally complete" descriptions of systems to be generated for the multiple quasi-independent domains. Standardization of a nomenclature for layer 2 (functional view) would appear to be a necessary prerequisite for the development of secure system standards for COTS software. A long-term goal to financially incentivize COTS vendors to produce that level of documentation in order to effectively sell to the government or "critical private" systems market may be feasible and would be beneficial to increasing the security accountability of widely used systems. With the advent of efforts such as DMTF's Common Information Model [**DMTF01**], the development of a more methodical development strategy could eventually be commonplace in the commercial software arena that could make accurate mapping of deployed code feasible.

We attempted to model a realistic attack scenario using one of the Red Team experiments conducted during the DARPA ISO IA Program (RT00-01) in the new worldview taxonomic framework but quickly ran into difficulties due to the lack of a standardized terminology.

We realized that a more tenable approach might be to begin to develop a taxonomy for information and that this could lead to a more focused terminology for the parsing of applications into a worldview model. This led us to review published literature in the semiotics, emergent systems/complexity science, transaction processing, and axiology ("science of values") areas [HAR01]. We then focused on the development of a Layer 1 compatible information framework (a trio of descriptors based on content, value, and quantity/complexity), a "typological" framework for information (based on its application, information operations and artifacts), and the development of a composition algebra to measure information complexity.

Toward a Taxonomy for Information

In order to develop workable instantiations of the three-layer worldview, it became increasingly apparent that one of the difficulties was going to be the lack of a taxonomy of *information* itself. We therefore began an investigation into what are the possible features of information. The following slides are a collection of ideas and thoughts with respect to the salient characteristics of information. These ideas do not present a comprehensive, coherent view of information (especially at this stopping point) but, given the necessary change of course, we thought that it was appropriate to document our research and our thinking to this point.

One feature that became quickly apparent is that there are two different aspects to information, which plays out very differently depending on whether the question is approached from a physics or biological viewpoint. A synopsis of one of the sources that we came across that presented this bifurcation is given in Figure C-12 [BAT79]. One of the most troublesome aspects of representing information in the networked communications realm is that both views apply. We are attempting to solve a problem that involves a society of intelligent unpredictable entities (a.k.a. people) interacting via semi-stable, reasonably well-known interaction protocols via unintelligent, but complex and obtuse tools (a.k.a. personal workstations). While information, in the context of

"IA" refers to the *biological definition* of information, the artifacts that we work with are characterized by the *physics-based definition* of information.

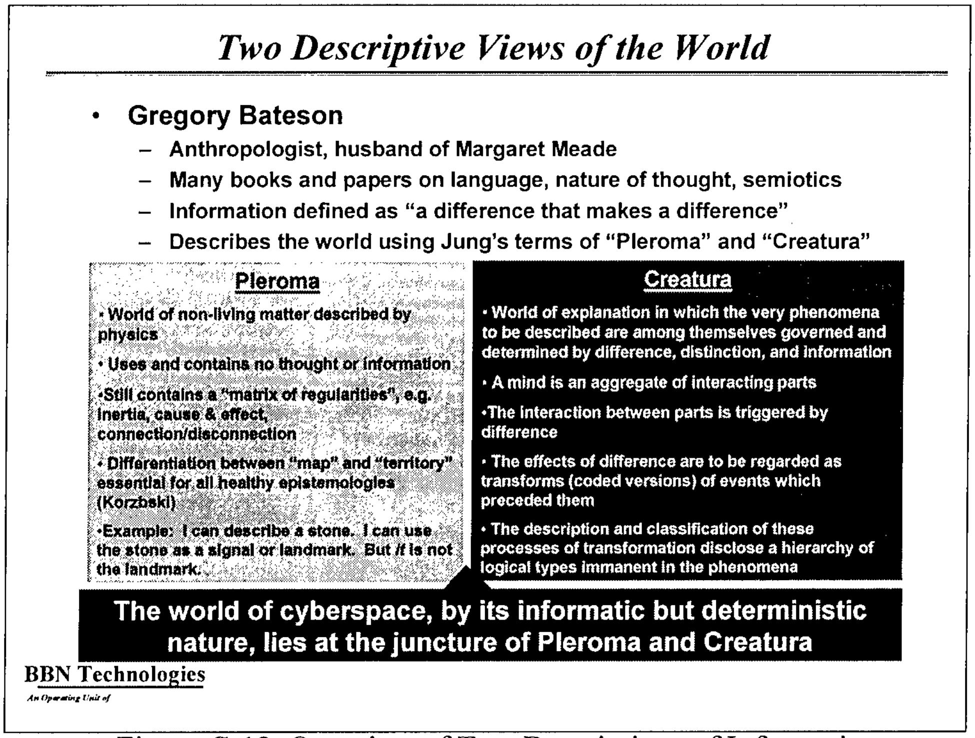


Figure C-12 Overview of Two Descriptions of Information

Definition of Information

One obvious question (with a less than obvious answer) is simply "What is information". It turns out that different communities are satisfied with very different answers. Bateson's view of information – of a "difference that makes a difference" – is a bit abstract, but at its essence it does indeed refer to the type of information referred to by "Information Assurance" (versus the quantitative "Shannonian" view that is content and context free and is not relevant in the IA context). An alternate definition even more closely suited to IA is: "an encoding that in some context is valued by an entity". A third definition, and one that lends itself to a quantization of information is "Information is a useful linking of concepts".

These three definitions of information each tend to emphasize a slightly different aspect of the *features* of information, that we now propose: *content, value, and complexity*. The partitioning of a particular particle of information into each of these domains is very much dependent on the contextual background of the receiver of the information rather than being solely properties of the information itself. Thus the characterization of a specific information particle into its constituent entities must be described both by the information itself and the contextual background of its consumer. Also, note that information is *digital* by the third definition – its being the documentation of a link between multiple concepts in finite dictionaries of discrete entries.

What is Information?

- There are two distinct characterizations of information:
  - one deriving from its meaning and purpose (predominant view in biology),
  - one deriving from its actual representation (predominant physical science view)
- Physics-based definition of information
  - Traditionally a "Shannonian" measure of complexity
  - Capacity-centric
  - Linking of concepts
- Biological (Semiotic) -based definition of information
  - "a difference that makes a difference", Bateson 1972
  - "an encoding that in some context is valued by an entity"
- Information Assurance lies on the cusp
  - Most definitely value-driven
  - Relies on deterministic processes
  - At the juncture of Pleroma and Creatura

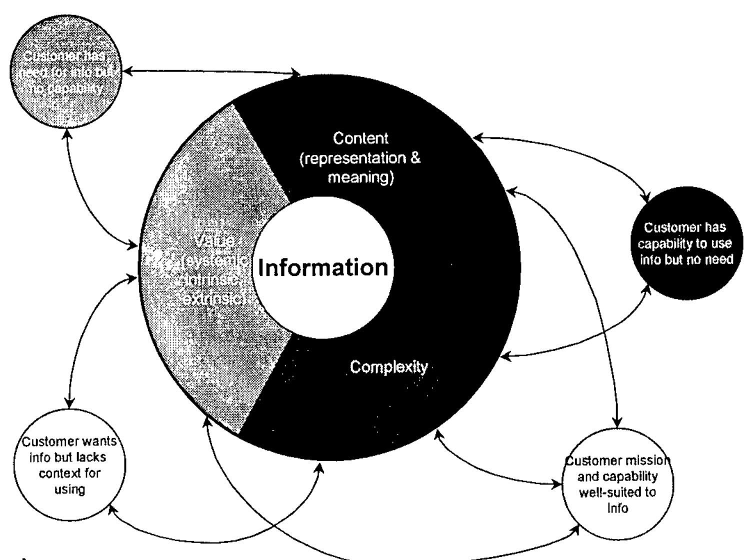
A "composite" definition:

Information is a useful linking of concepts

BBN Technologies

Figure C-13 Information Definitions

Information Uncertainty Triad



BBN Technologies

Figure C-14 A Triad of Information Particle Descriptors

Information Value

Information has a value that is dependent on both its intrinsic content's value to a potential consumer and a value based on its usability in terms of format. If information is encrypted such that the intrinsic information is not accessible within a timeframe of usefulness to the potential consumer, then its extrinsic value is greatly reduced (probably to zero). Intrinsic and extrinsic values are key descriptors in the development of the Layer 1 (Mission worldview) for both the defender and the attacker(s).

Taxonomy of Information ("the coin of the realm")

Layer 1 Value Descriptors

Information

Information == A finite (digital) encoding that is valued by another entity

Intrinsic Value , V_I(entity) == value that would be placed on information by an entity were its information in an accessible form

Extrinsic Value, V_E(entity) == value (that should be) placed on information - in its current form - by an entity

Example: "The book is on the table"
"Das Buch ist auf den Tisch"
"Quên sách trên bãn
"Uifacpplajtapoauifaubcmf"

V_I is identical for each of these phrases for a given entity (the value of the actual information is independent of its expression)

V_E differs vastly depending on what other knowledge or resources are available to a given entity

BBN Technologies

Figure C-15 Intrinsic and Extrinsic Value Descriptors

Value descriptors, are dependent on both the information content as well as on the context of the potential consumers of the information. An information particle with an identical artifact in the physical world may have very different value measures dependent upon its basis of origin as illustrated in Figure C-16. The notion of a systemic value is drawn from axiology – a science of "values" [HAR01].

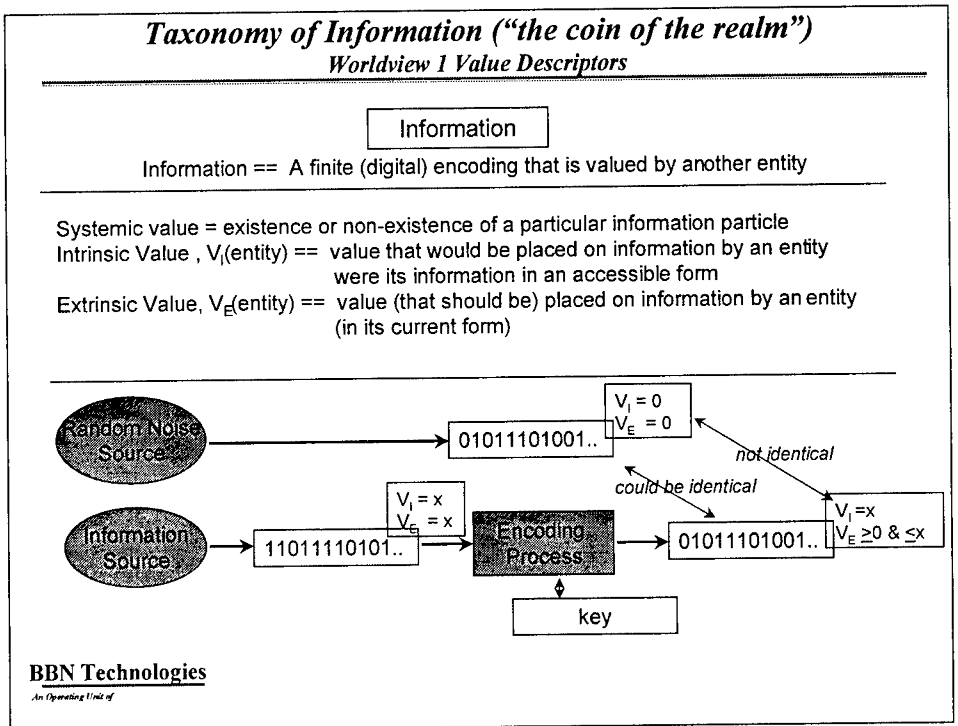


Figure C-16 Value Descriptors for Information

Knowledge may have very different values based on its application. Figure C-17 shows some initial thoughts on types of knowledge. The fourth category, *evological* knowledge, is the "eureka" type of knowledge that has great intrinsic value in terms of its newness and its resultant potential for bringing about significant changes in its environment.

Four Types of Descriptive Knowledge

- Instinctual
  - Knowing "what" will typically happen (not how or why)
  - Example: a goose has instinctual knowledge to fly South for thewinter
- Epistemological
  - Knowing "how" it works
  - Describing our perception of a system (what we know and possiblyhow we know what we know)
  - Examples:
    - 200 years ago people had an epistemological knowledge that a goose will fly South for the winter ;
    - my daughter has an epistemological knowledge of how my car opertes
- Ontological
  - Knowing "why" it works
  - Describing what actually exists (not just what we know)
  - Examples:
    - scientists are gaining ontological knowledge on the layers of processes that a goose uses in migrating South for the winter
    - my mechanic has an ontological knowledge of my car (hopefully)
- "Evological"
  - Knowing what something is capable of becoming
  - Describing what could be possible
  - Examples:
    - In the event of a catastrophic change in climate at some time lthe distant future, we could conceivably reprogram a goose to migrate somewhere different
    - an engineer may design subsystems (e.g. adaptive independent shock absorbers) to improve the performance of a state-of-the-art racing car

The value element of progress is gained in moving from the ontological stage to the evological stage – recombining concepts in new and useful ways

Our knowledge of cyberspace is mostly at the epistemological stage. Our knowledge of the computer technology underlying cyberspace is more ontologica l.

BBN Technologies

An Operating Unit of

Figure C-17 Some Categories of Descriptive Knowledge

Information Complexity

Information, being digital, can be quantized on two different levels. We can define the concepts of an information bit (infit) and an information byte (infyte?). An *infit* is a measurable quantity that maps concepts back to the expressed language of the information. An *infyte* is an atomic level of information – a "concept" – whose information complexity and content will be dependent on the conceptual dictionary employed. These concepts are defined in Figures C-18 through C-20.

Dictionaries, as commonly understood, actually can serve two somewhat distinct purposes. One purpose is to relate the expression of a concept in a particular language. A second, more subtle, role is to serve as a library of concepts that are commonly understood. Each entry in a dictionary consists of one or more concepts that have "congealed" in some community of entities, such that their expression is a very concise representation of a more complex structure. A medical dictionary will be very different from a general dictionary, which will be very different from a networking glossary – each is compiled for a different target audience that can have a very different knowledge base. This notion of the possession of multiple distinct typological dictionaries by an entity strongly influences the complexity of the expression of a concept for that individual.

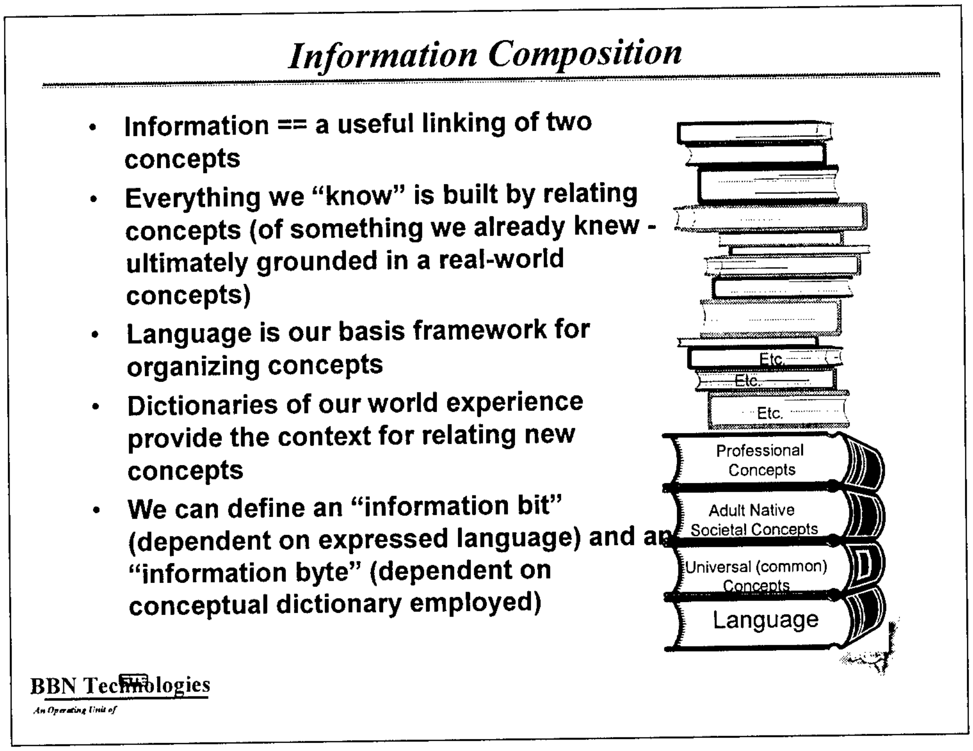


Figure C-18 Composing Information

Two fundamental units of information

- **Information bit ("infit")**
  - **atomic unit of information expressible in a given language**
- **Information byte ("infyte")**
  - **atomic unit of conceptual information**
- **Complexity is entirely dependent on the user's existing knowledge bases - complexity should be measurable by the required combinations of those existing (primitive) concepts**
- **Dictionaries will have different complexity measures depending on the number of more primitive dictionaries that they imply.**
- **Concepts that draw on an existing concept dictionary will have low complexity**
- **Concepts that combine multiple existing concepts into one or more layers of new concepts will have a higher complexity**

BBN Technologies

Figure C-19 Fundamental Units of Information

Information Composition (continued)

Information, as defined, can express any complicated set of conceptual relationships as a composite of infobytes

Information == a useful linking of concepts

$I(m) = (C_1, \ldots, C_n)$

A continually growing (but ideal) dictionary of concepts, **C**, may be defined as:

$\mathbf{C} = \{C_1, \ldots, C_z\}$

An expression of a linking of concepts is necessarily language-specific and can be expressed as

$I^{language}(m) = (C_1, \ldots, C_n)$

infyte == atomic unit of conceptual expression

$= C(m) = I_{(0)}^{language}(m)$

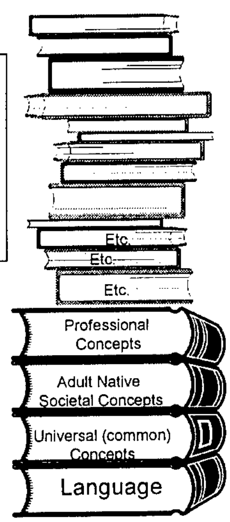
$I_{(0)}^{language}(m) = (i(j), i(k), \ldots, i(z))$

An ideal dictionary, **D**, for a language is defined as:

$\mathbf{D} = \{I_0^{language}(1) \ldots, I_0^{language}(n) \}$

infit = i(j) == atomic unit of information for a given language (aka a character)

$I^{English}$ = {a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z, (end of word character), (end of sentence character), (other standard punctuation) }



BBN Technologies

Figure C-20 Information Composition

Interdisciplinary Pitfalls

The practicality of the eventual development of a cyberscience is still an open question. One of the difficulties is that the relevant published literature spans a myriad number of distinct scientific areas – and some of the areas are still actively growing and are even known by multiple names. Many of the linguistic and anthropological fields in particular are intimately connected with the organization and conceptualization of information and are very relevant as we move from the bits and bytes of the machine to the "infits" and "infytes" of information particles. Some of the headings under which we found connections that were relevant are:

- Complexity science / Evolutionary Theory / General Biology / Memetics (evolutionary models of information transmission)
- Information Theory
- Quantum Computing & Quantum Information
- Cybernetics
- General Systems Theory
- Semiotics
- Artificial Intelligence
- Anthropology
- Philosophy
- Linguistics/Computational Linguistics
- Machine Translation
- Economics
- Ethics
- Axiology
- Game theory
- Psychology / Quantum relations to consciousness

A book on complexity science (a.k.a. "general biology") was published in the mid-October 2000 [KAU00]. It introduces several of the concepts that we have independently developed on this effort, namely: the layered worldview, an analogous typological framework for knowledge, and analogous thoughts on information composition and asymmetric information transfer mechanisms across layers. (The three-layer worldview (mission, function, and implementation) also appeared in an article in the September/October 2000 IEEE Internet Computing issue on Knowledge Networking concerning the construction of semantic networks[DEC00])

Kauffman's emphasis has been on the emergence of complex systems in evolutionary biology and its generalization to complex economic systems. This apparent convergence between Kauffman's "protoscience" work and ours provides some reassurance that we are on the right track with respect to the development of an underlying science for cyberspace. A key notion of Kauffman's work is that the configuration space of a biosphere is not *pre-statable*, which leads to a fundamentally different formulation for science in biology than in the world of physics (where the configuration space is pre-statable). This tends to explain the theory-primary/experiment-secondary formulation of physics vs. the artifact-dominant formulation of biology. We now believe that cyberscience, due to its essential nature of both cooperative & competitive self-interested agents interacting in defined semantic interactions using information vs. chemical interchanges exhibits the same open-endedness of configuration space. The development of a cyberscience will ultimately draw from and potentially contribute to ongoing work in complexity science.

We were in the process of attempting to incorporate the above notions when the effort was aborted.

2.2. Conclusions

• IA issues are important to all types of communication/information systems, not just those that involve computers and networks.

• Information security has traditionally been viewed with too little hierarchical structure - *a potpourri of worldviews.*

• A three-layer worldview approach to modeling security appears to be desirable in terms of providing descriptive language for each of the "users" (end-user, designer, implementer/maintainer) of a "system"

• Re-expressing system concepts in a layered "worldview" may allow "locally complete" descriptions of systems to be generated for the multiple quasi-independent domains

• Security is a multi-layer problem: a system consists of multiple logical and physical layers – system goals are always defined and satisfied at the 'mission' level (above the level of implementation); violations occur at the level of implementation. Attacks occur in the layer 3 (implementation) worldviews, while their effects are felt only in the layer 1 (mission) worldview - the lack of traceability across these domains (due to implicit rather than explicit mapping in most software) is a primary source of system insecurities

• The standardization of a nomenclature for layer 2 (functional view) is one path that could be followed for the development of secure system standards for COTS software. Formulation of systems under this architecture is a huge task. Efforts such as DMTF's Common Information Model [DMTF01] may lead to standardized languages for system layers that could prove suitable for the evolution of such a language. The long-term goal would be to financially encourage COTS vendors to produce that level of documentation in order to effectively sell to the government or "critical private" systems market.

• This model however is fundamentally asymmetric – one can traverse down through the layers but cannot unambiguously go up. In simplest terms, this means: one can build a system to accomplish a specific task; however one cannot in general take a specific implementation of a system and pre-determine its capabilities and limitations in a general sense.

• It is important to distinguish information in a semiotic viewpoint vs. physical viewpoint. Most engineering disciplines have viewed the term information with the physical (Shannonian) meaning-free view. The semantic and biological sciences are developing a much richer definition of information relevant to living systems (and computer-facilitated human networks). Exactly what constitutes information depends upon its context – the logical and physical layer of the communication system and the role that the information plays. The semiotic approach will require a triadic model (e.g. sign, symbol, meaning) for meaningful modeling of information systems, rather than the dyadic (stimulus, response) model applicable to modeling systems throughout the rest of the natural (non-human) world [Per84].

• IA issues become most relevant in a system when someone or something attempts to perform an operation upon a unit of information – transactions on information and exchanges of information.

• Security, at its essence, is strongly interdisciplinary due to the possibility of its being viewed as an extension of human interaction and motivations (psychology, economics, semantics, linguistics) rather than an engineering science. *Information,* in the IA context, is only information when there can be assigned a "value" to it by some entity.

• The practicality of developing a "cyberscience" remains an open question; however the predisposition of the authors based on work conducted during this effort would indicate that the "sciences" most applicable to this effort will be very similar to those currently in active research under the complexity studies of modern *general biology* (general biology links traditional biology and physics at a fundamental level) or *complexity science*. The analogies between the concepts in evolutionary biology (self/non-self, food/poison, "good/evil", self-interested agents interacting in defined semantical interactions) are striking. The science is very akin to a science of war (a.k.a. *survival* in biological terms).

• Technology is continually adding services, which the customer base doesn't even perceive the need for, but which quickly become entrenched such that it would be very difficult to go back to the time before the service was available. It'd be interesting to be able to partition the "needs" for these new communications technologies in terms of the missions to be accomplished and what effect there'd be if they were no longer there. When the customer base becomes used to a new service, they tend to get "spoiled" (probably driven by as much familiarity as convenience), but in the long run the user base doesn't really *need* the newer services (as long as the older method or something comparable of accomplishing the tasks are maintained and are usable - *this is very important*). Our attachment to the new services is more psychological than actual, in line with most people's basic evaluation of satisfaction being driven by "loss aversion" rather than gain [Sun00]. Maybe one of the best approaches to information survivability is studying the threats to a process and consciously maintaining redundant services that aren't vulnerable to those particular threats as a cost of doing business. There is a tremendous amount of redundancy present in most critical operational processes. Maintaining a redundancy is key, but the trick appears to be in maintaining that redundancy in the face of trying to operate with a minimal budget and some phantom (as opposed to existing) threat.

• To understand the nuances of IA – for all types of systems and across system layers, it is helpful to:

– Gain a better understanding of what information "really" is (What properties hold true for what layers? What types of transactions are valid and when?)

– Examine how information flows and is transformed within a system and between various system layers

– Explore systems and IA from a multi-disciplinary perspective.

2.3 Bibliography

[AHI98] Alhir, S.S., UML in a Nutshell. Cambridge, MA: O'Reilly & Associates.
[AHI87] Ahituv, N., A Metamodel of Information Flow: A Tool to Support Information Systems Theory. Communications of the ACM. 30(9):187-791.
[ASL95]Aslam, T. (1995): A Taxonomy of Security Faults in the Unix Operating System. Master's Thesis, Purdue University, Department of Computer Sciences.
[BAT79] Bateson, G., Mind and Nature, A Necessary Unity, 1979
[BIOS00] Biosemiotics web page http://everest.ento.vt.edu/~sharov/biosem (e.g. Sharov, "Signs and Values", 1997, Ogryzko, "Physics and Biosemiotics")
[BIS95] Bishop. M. (1995): Taxonomy of UNIX security flaws and survey
[BLA99] Blakley, G. R. Twenty Years of Cryptography in the Open Literature. IEEE. 1081-6011:106-107.
[BOH90] Bohm, D., A New Theory of the Relationship of Mind and Matter. Philosophical Psychology. 3(2):271-286.
[BOO94] Booch, G., Object-oriented Analysis and Design, Addision-Wesley, 1994
[BOO99] Booch, Rumbaugh, Jacobson, The Unified Modeling Language User Guide, Addision-Wesley, 1999
[CAR00] C. Carver, U. Pooch, "An Intrusion Response Taxonomy and its Role in Automatic Intrusion Response", Proceedings of 2000 IEEE Workshop on Information Assurance and Security, USMA, West Point, 6-7 June 2000
[CISL00] Feiertag, Kahn, Porras, Schnackenberg, et al, A Common Intrusion Specification Language (CISL), http://www.gidos.org/drafts/language.txt
[CUP93] Cuppens, F. , A Logical Analysis of Authorized and Prohibited Information Flows. IEEE. 1063-7109:100-108.
[DEC00] Decker, S. et al, The Semantic Web: The roles of XML and RDF, IEEE Internet Computing Magazine, Sept/Oct 2000
[DEN96] Denning, D., Branstad, D. (1996): Taxonomy of key escrow systems
[DMTF01] DMTF's Common Information Model (http://www.dmtf.org/)

[EMM91], Emmeche, C., Hoffmeyer, J., From Language to Nature – the semiotic metaphor in biology, Semiotica 84 (1/2): 1-42, 1991 http://www.nbi.dk/~emmeche/cePubl/91a.frolan.html

[GLI99] Gligor, V., Twenty Years of Operating Systems Security. IEEE. 1081-6011:108-110.

[HAL]Halpern, Moses, "Knowledge and Common Knowledge in a Distributed Environment", IBM Almaden

[HAL] Halme, L.R., Bauer, R.K.: Taxonomy of anti-intrusion techniques

[HAR01] Introduction to Axiology, Robert S. Hartman Institute, http://www.hartmaninstitute.org/ (somewhat "fringey" at first glance, but often quoted by various researchers)

[HOFF91] Hoffmeyer, J., Emmeche, C., Code-Duality and the Semiotics of Nature, http://www.nbi.dk/~emmeche/coPubl/91.JHCE/codedual.html

[HOW98] J.D. Howard, T.A. Longstaff, "A Common Language for Computer Security Incidents", Sandia Report, SAND 98-8667, October 1998

[HOW96] J.D. Howard, "An analysis of Security Incidents on the Internet, 1989-95", PhD dissertation, CMU 1996, www.cert.org/research/JHThesis/start.html

[JAC99] 1999 Jackson, D., A Comparison of Object Modeling Notations: Alloy, UML, and Z. MIT Lab for Computer Science.

[JAC9X] Jackson, D., Alloy: A Lightweight Object Modeling Notation. MIT Laboratory for Computer Science.

[JAC00] 2000 Jackson, D., Sullivan, K., COM Revisited: Tool-Assisted Modeling and Analysis of Complex Software Structures. MIT and University of Virginia.

[KAUF96] Kauffman, S., At Home in the Universe, Oxford Press, 1996

[KAUF00] Kauffman, S., Investigations, Oxford Press, 2000

[KEN00] Scott, K., The UML Dictionary.

[LAN94] Landwehr, Bull, McDermott, Choi, "Taxonomy of general computer program security flaws", ACM Computing Surveys 26-3, Sept 1994

[LIP99] Lipner, S., Twenty Years of Evaluation Criteria and Commercial Technology. IEEE. 1081-6011:111-112.

[MCL99] McLean, J., Twenty Years of Formal Methods.IEEE. 1081-6011:115-116.

[MCL9X] McLean, J., Security Models and Information Flow. Source Unknown.

[MCL90] McLean, J. The Specification and Modeling of Computer Security. IEEE. 1018-9162:9-16.

[MEE94] Meek, B.., Taxonomy of Datatypes, ACM Sigplan Notices. 29(9):159-167.

[MEM01] Journal of Memetics – Evolutionary Models of Information Transmission, A Brief Overview and History of Memetics, http://www.cpm.mmu.ac.uk/jom-emit/overview.html

[MIL99] Millen, J., Twenty Years of Covert Channel Modeling and Analysis. IEEE. 1081-6011:113-114.

[NEI97] Neighbors, J.G., Cyber-Geography: A Framework for Deriving Order Out of Chaos", USAF Institute paper, April 1997

[NRC99] NRC, Trust in Cyberspace, National Academy Press, 1999

[OGRXX] Ogryzko, V., Information and Physics, http://everest.ento.vt.edu/~sharov/biosem/txt/ogr3.html

[PER84], Percy, W., Lost in the Cosmos, (Semiotics digression), Picador Press, 1984

[ROG99] Rogers, M., Psychology of Hackers: Steps Toward a New Taxonomy
[SAK97] Sakinthinos, A. and Lee, E.S., A General Theory of Security Properties. In IEEE. 1081-6011:94-102.
[SEMI00], Glossary of Semiotics
[SHA48] Shannon, C. E., A Mathematical Theory of Communication. The Bell System Technical Journal. 27:397-423.
[SHAR98] Sharov, A., Signs and Values, http://everest.ento.vt.edu/~sharov/biosem/txt/isas98.html
[SHI99] Shirey, R., "Internet Security Glossary", RFC 2828
[SHO99] Shostack, A., Blake, S., Towards a Taxonomy of Network Security Assessment Techniques
[SNO99] Snow, B., The Future is Not Assured -- But It Should Be. IEEE. 1081-6011:240-241.
[SUB99] Subrahmanyam, A., Nuts and Bolts of Transaction Processing.
[SUN00] Sunstein, C. R., The Human Variables, http://www.thenewrepublic.com/080700/sunstein080700_print.html

[TSI82], Tsichritzis, D., Lochovsky, F.H., Data Models. Brian W. Kernighan, advisor. Prentice-Hall Software Series. Englewood Cliffs, NJ: Prentice-Hall, Inc., 1982

Other Relevant References:

Military Site on System Engineering: http://www.acq-ref.navy.mil/turbo2/areas/syseng.cfm

System Engineering Fundamentals (Military Handbook) http://www.dsmc.dsm.mil/pubs/gdbks/pdf/systemengfund.pdf

Earth Orbital System (EOS) System Engineering Documents (esp. this one) http://esdis-it.gsfc.nasa.gov/SYSENG/document.html

Toward a Secure System Engineering Methodology http://www.counterpane.com/secure-methodology.html

System Engineering With Models http://www.isis.vanderbilt.edu/activities/mic96/byron/byron.html

International Council on System Engineering http://www.incose.org/

System Security Engineering http://www.tcs-sec.com/system-security-engineering.html

System Engineering Bibliography http://www.incose.org/lib/sebib3.html

System Engineering Links http://freney.sys.virginia.edu/outside/SYSTEMS_ENGINEERING_SITES.html

Another Systems Engineering Bibliography http://mijuno.larc.nasa.gov/dfc/biblio/sysengBiblio.html

http://www.sie.arizona.edu/sysengr/index.html

Software Architecture Representation Tool http://www.mitre.org/technology/tech99/quads_summary_99/computing_software/chase_summary.html

System Engineering at NASA http://akao.larc.nasa.gov/dfc/syseng.html
Another bibliography http://akao.larc.nasa.gov/dfc/biblio/sysengbiblio.html
Requirements Engineering http://akao.larc.nasa.gov/dfc/re.html

Function Analysis http://akao.larc.nasa.gov/dfc/fa.html

3. Economics (Market-based Approach)

3.1 Overview

The application of economic market theories can help define which transactions will take place over a network and provide the basis for building tools which assess the vulnerability of various nodes to attack by contrasting:

- the value of the information to other parties,
- the protections in place via available access routes,
- the effect of degradation of protections over time as technology improves, vulnerabilities in widely deployed mechanisms are discovered, or offensive tools proliferate.

The purposeful interactions that occur on a computer network can be considered as an economic system of transactions by both willing and unwilling entities through an established exchange mechanism. The commodity being bartered in each of these transactions is information. A price is set for exchange (or protection) of information by supplying entities, and a purchase price is set by the procuring entities. Whether transactions take place in an authorized (willing) or unauthorized (surreptitious) fashion is driven solely by the implicit setting of these internal, market, and customer values for the information desired.

3.2 Technical Approach

We began to investigate these ideas through the development of an attack tree analyzer, where we treated three different attack parameters: the cost or difficulty of conducting an attack, probability of being detected during the course of an attack, and the probability of success for an attack as independent variables for which we could construct a utility function: f (c, d, f) which could be altered to evaluate a "cost" function for various types of attackers. We chose this set of variables in order to make use of the RT00-10 data set consisting of a set of four attack trees compiled by a professional Red Team and which had enumerated measurements for each of the parameters at each attack node.

3.3 Results

Results and Direction of the Effort as of November 2000

As part of the Cartography of Cyberspace project, we produced an algorithm to analyze an attack tree generated by a Red Team to try to find a set of ``optimal'' paths through the tree to the goals, we analyzed the results of the RT0001 Red Team attack trees and provided the results of the analysis for input into a DISCEX paper on RT-0001.

We first developed attack trees using a tool called Decision Pro (by Vanguard). This allowed us to construct graphically based attack trees on which could be performed spreadsheet type analysis functions to verify whether heuristic metrics would produce the expected results. The results were promising with this approach but we found it cumbersome to implement reasonably complex evaluation algorithms using this approach (it required repeating formulas throughout the underlying data structure), therefore we began to implement an evaluating function for this data set using a simple C++ based evaluator.

Taking a cue from economics and decision theory, the criteria for ``optimal" may be varied. The attack tree is analyzed by assigning one or more values and costs to each attack in the tree. Paths through the tree are selected that maximize value and minimize cost. Since different attackers will have different utility functions (e.g., some prefer a more heavy-handed approach that quickly yields a high payoff in the form of the ability to shut down a system, while others prefer undetected access through which they can continuously accrue benefits at of surreptitious access and eavesdropping), the analysis program is able to vary the way it calculates utility.

An additional consideration is the ``leverage" of an attack: attacks that open up more paths are preferred to those that open fewer.

For the initial analysis we used the assessments assigned by the Red Team when constructing the attack tree.

We have produced a program that shows promise of being useful to a Blue Team by letting them look at large attack trees, explore what-if scenarios, and locating points of defense that may be critical to thwarting the Red Team's strategies.

Working with this tool suggests that it would be even more useful to a Blue Team in combination with another tool that *generates* likely attack trees. In the real world, an ``enemy Red Team" is not going to pass their attack tree and assessments of the difficulty of the attack to the defenders of an installation --- the defenders will have to perform their own attack analysis.
Structure of the Blue Team Assistant (BTA)

The BTA could consist of two components: an attack-tree analyzer and an attack-tree generator.

The analyzer finds paths through the tree based on a combination of the following criteria:

- expectation of success on the first few nodes along the path.
- likelihood of detection along the path.
- estimate of the effort required to mount the attack.

The attack-tree analyzer allows the Blue Team to model different types of attacker by varying the weights placed on these criteria. In economic terms, the analyzer lets us vary the modelled attacker's utility function. For example, an intelligence agent is likely to emphasize avoiding detection, while someone mounting an attack during hostilities might emphasize the amount of damage that can be done in a short time, with no concern about detection.

Additional criteria employed are:

- number of paths opened by success on each subsequent node.
- length of path.

Paths are ordered by vulnerability, and recommendations for the correction of security vulnerabilities, placement of sensors, and other security measures are made. System administrators can mark vulnerabilites in the database as ``fixed'' in a given node and regenerate the attack tree, permitting an analyst to play ``what-if'' scenarios.

We intend the BTA's attack-tree analyzer to use data gathered from attack trees generated by any source. However, both for testing purposes and for practical use, it seems like a good idea to augment it with an attack tree generator.

Observations of Blue Teams indicate that it is difficult to find, and retain, the expertise needed to build effective attack tree models. Further, the attack trees that get built are strongly dependent on the expertise of the members of the team building the tree. A BTA program equipped with a knowledge base does not have this shortcoming: it's expertise is always present and, unlike the human beings on a Blue Team, the BTA does not get tired or bored. It can generate an attack tree that includes all (known) attacks on a networked system, producing an attack tree that is more complete than one assembled by an ad hoc team.

Further, unlike a Red Team, the BTA builds its attack tree with pre-existing knowledge of the network it is protecting and the distribution of hosts, operating systems, and critical tasks on that network. This, in turn, gives the Blue Team equipped with the BTA an advantage over a Red Team that is trying to surreptitiously determine architectural information.

The generator will take as inputs:

- a loose representation of the network topology (network A is connected to network B through gateways A1, A2, A3; B is connected to network C through gateways B1 and B2).

- this representation will note the type of network (particularly whether the network is a broadcast network enabling eavesdropping or not).

- a representation of the ``goal nodes'' on the network. This representation may be fairly loose, e.g., ``can execute code on node X'', ``can modify files on node X'', or ``can transmit data from node Y to node X'' (the latter presumed to enable sending errant commands to a network server).

- a parsed representation of security vulnerability databases such as the CERT database, the Mitre database, and also from the vulnerabilities probed by security scanners like Nessus and SATAN.

  The entry of this information into the database may be largely automatic (in the cast of things extracted from the Nessus exploit database) to largely clerical (in the case of extracting data from the informal format used by the CERT database). We estimate that it should take only a few minutes to insert a new vulnerability into the attack database.

From these it can first parse the nodes of the networks into equivalence classes (e.g., machines running the same OS attached to the same broadcast network need only be represented as one ``node" in the threat topology of the network). It then lists possible paths (from node to node representing equivalent machines) from the outside world to the goal nodes. These paths are then annotated with the sorts of security exploits known for machines running the given OS and set of servers used in a given node. This data is then used to generate the attack tree.

The resulting trees will be quite large, but not intractably so (e.g., four layers of network, with six different operating systems represented on each layer, where each OS has database of 200 vulnerabilities, will generate an attack tree with only about 2400 nodes).

How the BTA Differs from Security Scanners

A security scanner operates on the principle that you make your system more secure by fixing all the holes. This is a monumental task, and is large enough that it often does not get done.

The BTA differs from security scanners primarily in focusing its attention on protecting those nodes that are most critical to the network's mission. The BTA suggests which holes represent the biggest threats to the mission, and which therefore should be fixed first.

Secondly, the BTA is (or at least can be) passive. It does not actively probe your system, ringing alarms in intrusion detection (nor can its probes serve to camouflage a cleverly surreptitious attack - indeed, if one has something eavesdropping on a network shared with a host running one of these scanners, one can learn as much about the vulnerabilities of the network as the scanner, without actually doing any probing).

In an environment where one can run a security scanner, it will be useful to adapt the scanner's results as input into the attack-tree generator, since the scanner will refine the picture of what vulnerabilities exist on the network. It should be noted, however, that it is easier to document the existence of an attack than it is to write a test that scans for it, so the vulnerability database is likely to be more complete than the security scanner.

How the BTA Differs from an Intrusion Detection System

The BTA differs from a straightforward intrusion detection system by being a preventative tool. However, the BTA's analysis of its attack-trees can serve as a knowledge-base to focus the attention of an intrusion detection system on monitoring for the most likely attacks --- including monitoring that might otherwise be onerous if done throughout the network, but which becomes practical when focused on a few particularly vulnerable nodes.

The BTA can also be used in conjunction with an intrusion detection system to indicate what other problems to search for when an intrusion is detected. If an intrusion is detected in an inner layer of the network, the attack tree analyzer can be asked to indicate what sorts of other security breaches came first, opening the door for the detected intrusion.

Support for Graceful Degradation Under Attack

The principal support for graceful degradation of service while under attack would come from the fact that one can strengthen only those security measures that are effective against the given attack. One may also be able to trace the attack backwards through the attack tree and isolate the attacker further away from their goal.
Proposed Experiments

Once the BTA it built, Red Team experiments should be run in which the Blue Team uses the BTA to help them design their defense. These experiments will measure the effectiveness of our theories modeling Red Team utility functions, and will also help refine the attack tree generator.

3.4 Bibliography

https://archive.ia.isotic.org/dscgi/ds.py/View/Collection-387

DARPA Information Assurance Program. RT0001 Experiment Plan, Adversary Course of Action Determinination,BBN Technologies, May 2000

RT0001: Red Team Attack Summary.

These two documents describe an information assurance experiment that profiled a series of attacks upon a network whose mission was order processing.

[SCH99] Schneier, Bruce.
Attack Trees: Modeling Security Threats. Dr. Dobb's Journal, December 1999.

4. Thermodynamics

4.1 Overview

This task was exercised as a small literature research adjunct to the Cartography effort. There are very many similarities between the mathematics of thermodynamics and that of information theory. At a very rudimentary level, analogies can be made in terms of work required to gather an information cache, its requirement for confidentiality to remain confined, and that of the work required to concentrate a heat source and keep it contained in order to provide useful work later. Crude concepts analogous to heat, entropy, and work can be contrived which pertain to both systems (entropy has been one of the most apt descriptors of an information base ever since Shannon's pioneering information theory work in the 1940s). Thermodynamic or Statistical Mechanic theory develops rather abstract but ultimately useful concepts by virtue of the statistical nature of many-bodied problems – which also apply to information networks.

Discrete information caches in a system are similar in several respects to discrete heat sources in a thermodynamic system. The gathering of a unique information source requires the input of work. Once collected it will tend to disperse unless insulated from the outside world. It has an instrinsic "energy value" to outside world. When information is concentrated in distinct locations, it is possible to extract work from the system and the system left unto itself will tend to equalize so that information is distributed (increase of entropy). When information is distributed evenly throughout a system, no work is possible (although the information is preserved, when everyone knows the same thing, information flow drops to zero; no work can be extracted; just as a high temperature in a thermodynamic system is of no practical value if *everything* is at that temperature

Given the reduced scope for this task, the approach that we adopted was to review the published literature for recent advances in the field and begin to summarize the work being done in both areas in the context of information. What we found was that there has been an amazing amount of recent work in the field - the most promising link of which coming to fruition as three interwoven concepts: Quantum Communications, Quantum Computing, and Quantum Information Theory. As a result of this very promising effort, we focused our efforts primarily in research the Quantum-related work in the area and began to come up to speed on the current thinking in the area.

At the time this task was being redirected in Fall 2000, the DARPA DSO/ITO offices released a Broad Agency Announcement in the Quantum Information Science and Technology (QuIST) area. We utilized the background research that we had performed on this effort as a base to participate on two coalitions within the applicability of the QuIST interest area. BBN proposed on two of the QuIST efforts, one as the prime contractor, focused on building a Quantum Information testbed in conjunction with a team of quantum researchers from Harvard and Boston Universities; the other as a subcontractor to Northwestern University (in conjunction with Telcordia), where BBN will be supplying networking protocol support for Quantum Technology originating at Northwestern, complementing the optical layer support being supplied by Telcordia. Both of these proposal efforts were successful.

4.2 Results

A month-by-month synopsis of the research performed on this effort follows:

June

Continued collecting research materials, reading the literature, and begin formulating an approach. The approach that presently appears most useful, based on plausible developments over the next 5-15 years, is to reposition the traditional concepts of Information Assurance and Security in a framework based on Quantum Information Theory rather than Classical Information Theory (e.g. the more modern approach to the concept of entropy). This would include introducing those concrete concepts that change in the quantum viewpoint, and their ramifications for the future (quantum computing, e.g. factoring, quantum key exchange, quantum cryptography; and quantum communication, e.g. quantum teleportation). And then extending this to include those concepts from the physical world (concerning measurements, experimentation, and observables, etc.) that change the way one looks at the future directions of computation and communication concerning security issues.

July

Our approach was refined as a result of now having reviewed and assimilated a sizeable fraction of the available literature. We are focused on repositioning the traditional concepts of IA&S in a framework based on Quantum Information Theory rather than Classical Information Theory (e.g. the more modern approach to the concept of entropy). Our goal is to delineate concepts that change in the quantum viewpoint, and their ramifications for the future (quantum computing, e.g. factoring, quantum key exchange, quantum cryptography), and extend this to include those concepts from the physical world (e.g. measurements, experimentation, and observables, etc.) that change the way one looks at the future directions of computation and communication concerning security issues.

The thermodynamic effort is leading us to believe that it may be advantageous to reposition the lowest taxonomy layers in terms of Quantum Information Theory concepts. This may yield two advantages: (1) it allows the greatest extensibility (permitting one to go beyond traditional classical constraints); (2) and permits the widest variety of higher level emergent concepts to be expressed (even contradictory ones). In addition, this is consistent with our goal of having the greatest freedom of expression at the highest levels of our taxonomy. For example, it does not hinder (and might help) the portrayal of game-theoretic concepts from economics and efficient spectral analysis compression and event correlation techniques. Furthermore, it does not constrain our critical requirements of using an object-oriented approach in a UML description language

August

Continued collecting research materials, reading the literature, and refining the formulated approach. As mentioned previously, the approach that presently appears most advantageous is to use a layered structure for an IA&S taxonomy, and reposition the lowest layers in terms of Quantum Information Theory concepts. This gains two distinct advantages: (1) it allows the greatest extensibility (permitting one to go beyond traditional classical constraints); (2) and permits a wide variety of traditional classical emergent concepts to be expressed at the higher levels (even contradictory ones). This month, the essential framework of the necessary minimal modifications to the lowest levels of a taxonomy to incorporate quantum mechanical systems (and the laws of quantum physics) as their fundamental starting point (root) was clarified. There are four key concepts:

At the lowest level are *purely quantum* systems (obeying the laws of quantum physics). These are coherent non-local systems that have some of their physical properties (information) encoded in the relative phases between their constituent parts.
At a higher level, with an increase in the *complexity* of the quantum mechanical system, classical behavior begins to emerge. The differences between quantum mechanical systems and classical systems are due to the complexity of the latter (classical) systems. When one attempts to examine (e. g. measure) a small part (sub-system) of a large complex quantum system, the interactions between all the other parts of this complex system produces a rapid *thermalization* of the unspecified degrees of freedom. In other words, the relative phases between the different parts of the system are destroyed, leaving behind a purely localized state with classical properties. In particular, it is the interaction with this *thermal reservoir* that destroys the non-local aspects of the quantum mechanical interactions (the relative phases average to zero).

Depending on the particular property (sub-system) of the quantum mechanical system being measured (as determined by the experimental set-up selected), different classically logically contradictory results may be obtained in different experiments (e. g. particle Vs wave). This is correct (consistent with experiment), as the logic of quantum mechanics is based on an Ortho-Algebra of Vector Spaces, that is not distributive, and is not a Boolean Algebra. Thus, classical Boolean logic can not be applied when comparing classical physical properties across different experiments. Hence, a *wider array of emergent* *concepts* (even at the higher classical levels) is permitted in a logical structure grounded in a quantum mechanical framework.

September

We continued collecting research materials, reading the literature, and refining the formulated approach. The essential framework for the necessary minimal modifications to the lowest levels of a taxonomy, to incorporate quantum mechanical systems (and the laws of quantum physics) as their root was clarified last month. These modifications were then expressed as four conceptual states, each with the mechanism for their transformation (simplicity permitting "pure" coherent quantum systems, complexity introducing classical behavior, thermalization destroying non-locality, and measurement selecting properties). This month the effort to integrate these lowest, quantum mechanical, taxonomy levels into the ongoing effort concerned with the higher traditional classical levels of an Information Assurance (IA) taxonomy began in earnest. It appears that concentrating on a small number of properties, or a single property such as information, and how its semantic meaning (or content) changes as one moves up through the higher emergent layers would be a more informative process. This has a tradeoff, as it produces a smaller scale "toy" IA taxonomy, rather then a full-fledged IA taxonomy of the Internet, but is also more consistent with a research-oriented goal and the available resources.

In addition, it is becoming clear that a number of other constraints, which are emerging from this effort must be adhered to if a consistent taxonomy is to be produced. First, a layered taxonomy must be very asymmetric, it is one way, an emergent concept arises in the (n+1)'th level to simplify the complexity in the n'th level (as a short-hand gestalt). Thus, emergent concepts emerge upward, never downward. Second, to avoid paradoxical situations due to this asymmetry, each layer's content should be definable (and describable) in terms of concepts from its own layer. In particular, the definition of items in a lower layer should never require the use of emergent concepts from a higher layer (so as to avoid bootstrapping and isolated cycles). For example, in a layered structure consisting of physical components, operating systems, functionality, and intent; it is clear that a particular intent (such as calculating profit/loss) is an emergent shorthand (at the highest layer) for describing an operation by a large number of components (in the lowest layer). And it should not be needed or used in defining what a particular component (such as a transistor) is at the lowest level.

October

Due to the recent change in the IASET charter, work was stopped on this task during October. A related item of note: on October 23 & 24, Bill Nelson and Mike Frentz attended the DARPA DSO/ITO Quantum Information Science & Technology (QuIST) Workshop to assess the relevance of our thermodynamic/quantum-computing interests developed under the Cartography effort to the new initiatives. We will be submitting two abstracts in response to the DARPA QuIST announcement (BAA 01-11). BBN is the proposed prime on one of the abstracts and is participating on a team with Northwestern University on another. No ISO program funds were expended due to our participation on this effort (but we mention it in the context that our participation was a direct outgrowth of the original IASET effort)

4.3 Bibliography

1. Entropy on the World Wide Web
   http://www.math.washington.edu/~hillman/entropy.html
   Claude Shannon, "A Mathematical Theory of Communication"
   Source for papers on entropy and Classical Information Theory.
2. David Deutsch "The Fabric of Reality"
   http://www.qubit.org/people/david/David.html
   For a more recent "further out" world-view (note: you don't have to accept his "many-worlds" viewpoint to understand his main points).
3. Overview of Quantum Computing
   http://xxx.lanl.gov/abs/quant-ph/9708022
4. An Introduction to Quantum Computing for Non-Physicists
   http://xxx.lanl.gov/abs/quant-ph/9809016
5. Quantum Information Theory Graduate level course
   http://www.theory.caltech.edu/people/preskill/ph229/#describe
6. Sam Treiman "The Odd Quantum" non-expert introduction to Quantum Mechanics.
   ISBN 0-691-00926-0, 1999

☆U.S. GOVERNMENT PRINTING OFFICE: 2002-710-038-10225

MISSION
OF
AFRL/INFORMATION DIRECTORATE (IF)

The advancement and application of Information Systems Science and Technology to meet Air Force unique requirements for Information Dominance and its transition to aerospace systems to meet Air Force needs.