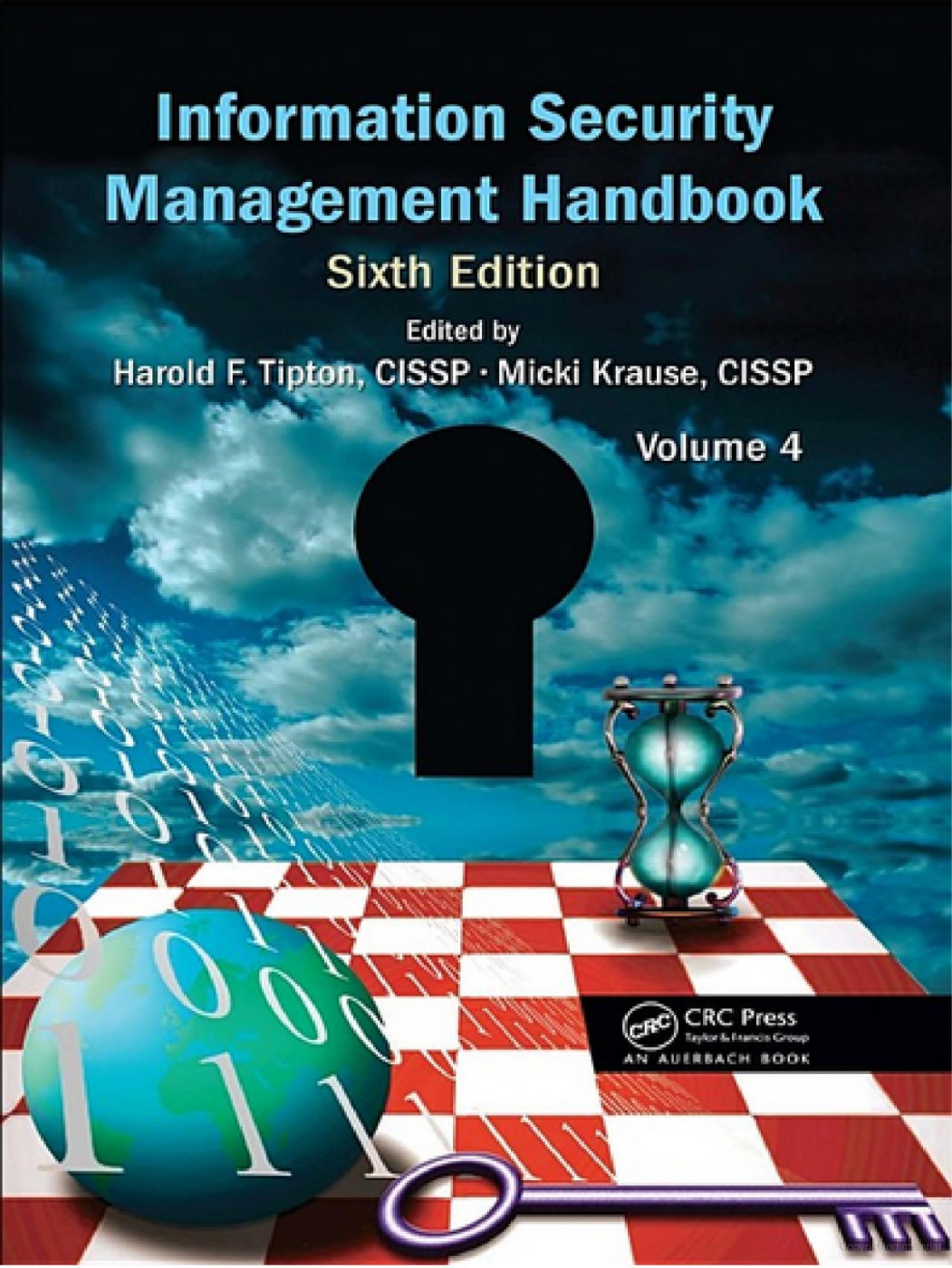
Information Security Management Handbook

Sixth Edition

Edited by

Harold F. Tipton, CISSP · Micki Krause, CISSP

Volume 4



CRC Press
Taylor & Francis Group
AN AUERBACH BOOK

Information Security Management Handbook

Sixth Edition

Volume 4

Edited by

Harold F. Tipton, CISSP • Micki Krause, CISSP

CRC Press
Taylor & Francis Group
Boca Raton London New York

CRC Press is an imprint of the
Taylor & Francis Group, an **Informa** business
AN AUERBACH BOOK

Auerbach Publications
Taylor & Francis Group
6000 Broken Sound Parkway NW, Suite 300
Boca Raton, FL 33487-2742

Auerbach Publications is an imprint of Taylor & Francis Group, an Informa business

No claim to original U.S. Government works

International Standard Book Number-13: 978-1-4398-5886-8 (ePub)

Contents

Preface

The days of wringing hands and warning of "fear, uncertainty, and doubt" have given way to thoughtful and intelligent approaches to protecting information. This is due, in great part, to the adoption of comprehensive and far-reaching standards that foster the practice of integrating security into the business.

So, although some may still be "atwitter" with loud and dramatic cries for enhanced, strengthened, and enforced security, many organizations are realizing the benefits of embedding appropriate controls into ongoing operations, thereby yielding effective and efficient safeguards.

Enter the *Information Security Management Handbook*, which for over a decade has offered a virtual toolset of essays and dissertations addressing people, processes, and technologies. The information herein is practical, useful, and hands-on. The chapters are written by dedicated and committed authors who seek to share their "been there, done that" stories with those who may benefit from them. Within each of the chapters, you will find personal histories and problem solving that each author has been gracious enough to share. We

thank them.

Further, the handbook's mission is to be used by a wide audience. Yes, the chapters are of substantial value to the security professional; however, they also address issues applicable to managers, executives, attorneys, risk managers, technology operators, and beyond. So, read hearty. If you learn one thing or find one idea to apply, we have succeeded.

As always, we wish you the best.

Harold F. Tipton
Micki Krause Nozaki

Editors

Harold F. Tipton, currently an independent consultant, was a past president of the International Information System Security Certification Consortium and a director of computer security for Rockwell International Corporation, Seal Beach California for about 15 years. He initiated the Rockwell computer and data security program in 1977 and then continued to administer, develop, enhance, and expand the program to accommodate the control needs produced by technological advances until his retirement from Rockwell in 1994.

Tipton has been a member of the Information Systems Security Association (ISSA) since 1982. He was the president of the Los Angeles Chapter in 1984, and the president of the national organization of ISSA (1987–1989). He was added to the ISSA Hall of Fame and the ISSA Honor Role in 2000.

Tipton was a member of the National Institute for Standards and Technology (NIST), the Computer and Telecommunications Security Council, and the National Research Council Secure Systems Study Committee (for the National Academy of Science). He received his BS

in engineering from the U.S. Naval Academy and his MA in personnel administration from George Washington University, Washington, District of Columbia; he also received his certificate in computer science from the University of California, Irvine, California. He is a certified information system security professional (CISSP), ISSAP, and ISSMP.

He has published several papers on information security issues for

Auerbach Publishers— *Handbook of Information Security Management*

Data Security Management

Information Security Journal

National Academy of Sciences —*Computers at Risk*

Data Pro Reports

Elsevier

ISSA "Access" Magazine

He has been a speaker at all the major information security conferences including the following: Computer Security Institute, the ISSA Annual Working Conference, the Computer Security Workshop, MIS Conferences, AIS Security for Space Operations, DOE Computer Security Conference, National Computer Security Conference, IIA Security Conference, EDPAA, UCCEL Security & Audit Users Conference, and Industrial Security Awareness Conference.

He has conducted/participated in information

security seminars for (ISC)²®, Frost & Sullivan, UCI, CSULB, System Exchange Seminars, and the Institute for International Research. He participated in the Ernst & Young video "Protecting Information Assets." He is currently serving as the editor of the *Handbook of Information Security Management* (Auerbach). He chairs the (ISC)² CBK Committees and the QA Committee. He received the Computer Security Institute's Lifetime Achievement Award in 1994 and the (ISC)²'s Hal Tipton Award in 2001.

Micki Krause Nozaki, MBA, CISSP, has held positions in the information security profession for the past 20 years. Krause was named one of the 25 most influential women in the field of information security by industry peers and *Information Security* magazine as part of their recognition of Women of Vision in the field of information technology (IT) security. She received the Harold F. Tipton Award in recognition of sustained career excellence and outstanding contributions to the profession.

She has held several leadership roles in industry-influential groups, including the Information Systems Security Information (ISSA) and the International Information Systems Security Certification Consortium (ISC)², and is a passionate advocate for professional security leadership.

She is also a reputed speaker, published author, and coeditor of the *Information Security Management Handbook* series.

ACCESS CONTROL

Access Control Administration

Chapter 1

Back to the Future

Paul A. Henry

Contents

About the Author

Network security appears (at least to the author), in some respects, to have come full circle. Many of today's so-called innovations in network security can in fact, at least in part, be traced back to having originally been implemented in one form or another in the decades-old Orange Book standards. The author, having worked with a firewall vendor that in the early 1990s had developed the first (and only) firewall to achieve an Orange Book "B Level" certification, gained a perhaps unique—firsthand perspective—of the security benefits of the components of an Orange Book–compliant security implementation.

Ironically, many of the features of Orange Book that were shunned in the commercial marketplace decades ago are now being embraced in one form or another in security implementations as the only sensible solutions to the environment we find ourselves in today. Perhaps Orange Book requirements were simply decades ahead of their time.

In our efforts to solve the most pressing issues that we face in network security today, perhaps a trip "Back to the Future" and a reexamination of the security provisions of the decades-old Orange Book are in order.

Revisiting Orange Book

In the late 1980s–early 1990s, the methodologies that

were core components of trusted computer systems often referred to as Orange Book–based security were adopted by a small number of network security product vendors. While no one can argue that adopting Orange Book security did not provide for a higher level of attainable security, the commercial marketplace literally shunned them as overkill, administratively burdensome, and relegated it as old technology.

For those of us working for security product vendors at that time, it was widely felt that anything above an Orange Book B1 level was simply not achievable and sustainable in a commercial security product. See Figure 1.1.

Official Overview of Orange Book Classes

Class (D): *Minimal protection*

This class is reserved for those systems that have been evaluated but that fail to meet the requirements for a higher evaluation class.

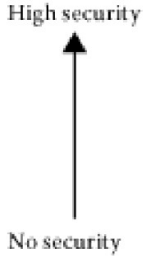
Class (C1): *Discretionary security protection*

The Trusted Computing Base (TCB) of a class (C1) system nominally satisfies the discretionary security requirements by providing separation of users and data. It incorporates some form of credible controls capable of enforcing access limitations on an individual basis, that is, ostensibly suitable for allowing users to be able to protect project or private information and to keep other users from accidentally reading or destroying their data. The class (C1) environment is expected to be

one of cooperating users processing data at the same level(s) of sensitivity.

Class (C2): *Controlled access protection*

Systems in this class enforce a more finely grained discretionary access control than (C1) systems, making users individually accountable for their actions through login procedures, auditing of security-relevant events, and resource isolation.



A1	Verified design
B3	Security domains
B2	Structured protection
B1	Labeled security protection
C2	Controlled access protection
C1	Discretionary security protection
D	Minimal protection

Figure 1.1 Orange Book Security Classes. (From Department of Defense, *Trusted Computer System Evaluation Criteria*, DOD 5200.28-STD, December 1985, Appendix C, pp. 88–89.)

Class (B1): *Labeled security protection*

Class (B1) systems require all the features required for class (C2). In addition, an informal statement of the security policy model, data labeling, and mandatory access control over named subjects and objects must be present. The capability must exist for accurately labeling exported information. Any flaws identified by testing must be removed.

Class (B2): *Structured protection*

In class (B2) systems, the TCB is based on a clearly defined and documented formal security policy model that requires the discretionary and mandatory access control enforcement found in class (B1) systems to be extended to all subjects and objects in the ADP system. In addition, covert channels are addressed. The TCB must be carefully structured into protection-critical and non-protection-critical elements. The TCB interface is well defined and the TCB design and implementation enable it to be subjected to more thorough testing and more complete review. Authentication mechanisms are strengthened, trusted facility management is provided in the form of support for system administrator and operator functions, and stringent configuration management controls are imposed. The system is relatively resistant to penetration.

Class (B3): *Security domains*

The class (B3) TCB must satisfy the reference monitor requirements that it mediate all accesses of subjects to objects, be tamperproof, and be small enough to be subjected to analysis and tests. To this end, the TCB is structured to exclude code not essential to security policy enforcement, with significant system engineering during TCB design and implementation directed toward minimizing its complexity. A security administrator is supported, audit mechanisms are expanded to signal security-relevant events, and system recovery procedures are required. The system is highly resistant to penetration.

Class (A1): *Verified design*

Systems in class (A1) are functionally equivalent to those in class (B3) in that no additional architectural features or policy requirements are added. The distinguishing feature of systems in this class is the analysis derived from formal design specification and verification techniques and the resulting high degree of assurance that the TCB is correctly implemented. This assurance is developmental in nature, starting with a formal model of the security policy and a formal top-level specification (FTLS) of the design. In keeping with extensive design and development analysis of the TCB required of systems in class (A1), more stringent configuration management is required and procedures are established for securely distributing the system to sites. A system security administrator is supported.

Security Architecture Models in the Era of Orange Book

1. Bell–La Padula
   The Bell–La Padula confidentiality model provides the "mandatory" component of a mandatory access control system with the following mandatory access control parameters:
   a. Top Secret level subjects
      i. Top secret level subject can create as well as write only top secret level objects
      ii. Can read top secret level objects as well as lower sensitivity level objects—secret and confidential

iii. Cannot write "down" to lower sensitivity level object—secret and confidential

b. Secret level subjects

i. Secret level subject can create as well as write secret level objects and top secret level objects

ii. Cannot read "up" in top secret level objects

iii. Can read secret level objects as well as lower sensitivity level objects — confidential

iv. Cannot write "down" to lower sensitivity level object—confidential

c. Confidential level subjects

a. Confidential level subject can create as well as write confidential level objects as well as secret and top secret level objects

b. Can read only confidential level objects

c. Cannot read "up" in top secret or secret level objects

A common theme among applications of mandatory access control is the "No read up—No write down" policy applied to each subject's sensitivity level. This is the "mandatory" part of mandatory access control.

It is the implementation of the Bell–La Padula security model:

i. *Simple security property*
The subject cannot read information from an object with a higher sensitivity level than the subject's

ii. *Star property*

The subject cannot write information to an object with a sensitivity level that is lower than the subject's

2. Biba
   The Biba formal model was written by K.J. Biba in 1977 and is the basis for the "integrity": aspects of the mandatory access control model. The Biba formal model provides for three primary rules:
   a. An access control subject cannot access an access control object that has a lower integrity level
   b. An access control subject cannot modify an access control object that has a higher integrity level
   c. An access control subject cannot request services from an access control object that has a higher integrity level
3. Clark–Wilson
   The Clark–Wilson formal model was written by Dr. David D. Clark and David R. Wilson in 1987, was updated in 1989, and like the Biba formal model, it addresses integrity. However, unlike the Biba formal model, the Clark–Wilson formal model extends beyond limiting access to the access control object by adding integrity considerations to the processes that occur while using the access control object.

   The Clark–Wilson formal model effectively provides for the integrity of the access control object by

controlling the process that can create or modify the access control object.

Further, the Clark–Wilson formal model also provides for the separation of duties. This aspect of the Clark–Wilson formal model establishes guidelines that require that no single person should perform a task from beginning to end and that the task should be accomplished by two or more people to mitigate the potential for fraud in one person performing the task alone.

a. Well-formed transaction
   The well-formed transaction is the basis of the Clark–Wilson model and provides for integrity through the use of rules and certifications applied to data as it is processed through various states. A well-formed transaction also employs the use of separation of duties whereby the implementer of a transaction and the certifier of a transaction must be separate entities.

b. Access Triple
   Historically, the Clark–Wilson Triple referred to the relationship between an authenticated user, the programs that operate on the data items, and the data itself. Similarly, an Access Triple refers to an authenticated user having permission to use a given program upon a specific set of data.

4. Brewer–Nash—Chinese Wall
   The Chinese Wall adds an additional element—the interrelationships of data to other models. In an

example of the addition of a Chinese Wall to the Bell–La Padula, not only would a given user be restricted to only accessing a specific set of data, but a further consideration of what other data sets the user had previously accessed would be examined before permitting access to the data. In an example of Clark–Wilson augmented with a Chinese Wall, not only is access to data restricted to a given process, but consideration is also given to which other data the processes had been used upon.

An Unofficial View of Orange Book Classes

C1, C2—Simple enhancement of existing systems that did not break applications

B1—Relatively simple enhancement of existing systems that will break some applications

B2—Relatively major enhancement of existing systems that will break many applications

B3—Systems that failed A1 certification

A1—Complete top-down design and implementation of a new system from scratch

While originally written for military system usage, the security classifications that were at the very core of Orange Book are today, decades later, being adopted and being used within current generation network security products. A few, perhaps "overly simplified," examples we will discuss in this chapter are

- Positive security model
- Mandatory protection
- Use of data classification labeling
- Covert channels

Driven by the changing threat environment

If we look back at 1988, only a single advisory was published by CERT for the entire year. In 2000, for the first time in history, BugTraq reported that the number of new vulnerabilities reported monthly had exceeded 100 (Figure 1.2). By 2006, the number of annual vulnerabilities cataloged unofficially by CERT had grown to 8064 (Figure 1.3). By 2007, obfuscation of malware had become a common practice, and by 2009, due to the use of obfuscation, the number of unique samples of malware found in the wild exceeded 5,500,000 samples annually (Figure 1.4).

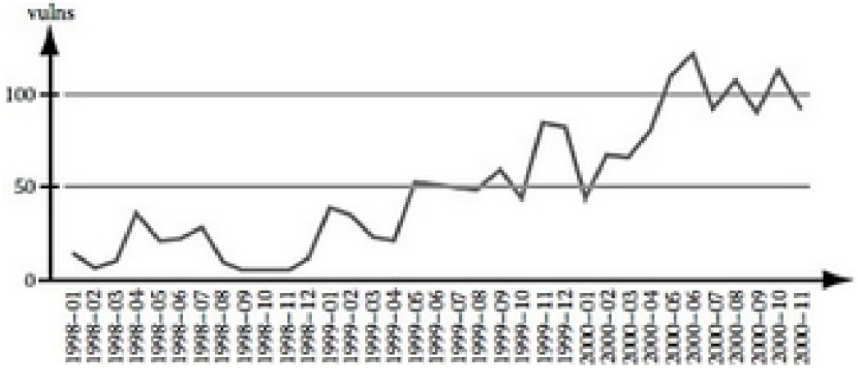


Figure 1.2 In 2000, BugTraq reported that the number of new

vulnerabilities reported monthly had exceeded 100.

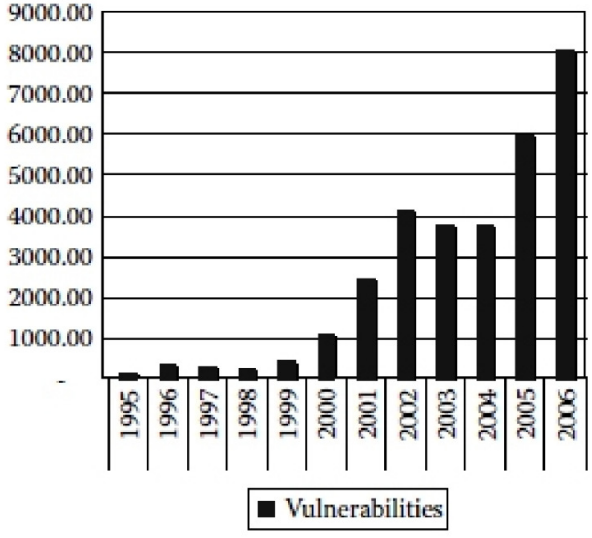


Figure 1.3 By 2006, the number of annual vulnerabilities recorded by CERT had grown to 8064.

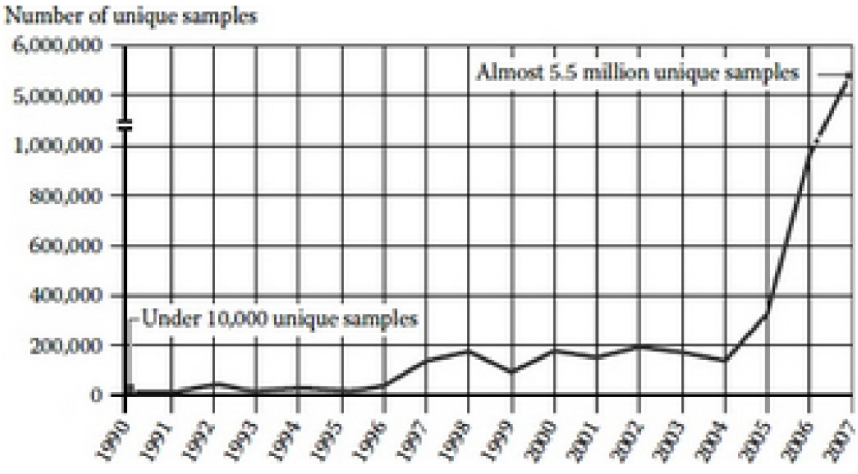


Figure 1.4 By 2007, the number of unique samples of malware found in the wild exceeded 5,500,000 samples annually.

Positive Security Model

In a positive security model–based defense, the security administrator must first configure the security product (such as a firewall) to match the business needs of the organization.

Denied by default—simply put if not configured as "known good," is determined to be necessary to meet the business needs of the organization and is explicitly allowed to pass by formal policy, a given packet is simply blocked by default. The security afforded by a positive security model–based firewall includes multiple layers of defense:

- Dramatically reduces the organization's threat

envelope by allowing only those packets that meet the business needs of the organization to pass

- Provides for complete protocol validation thereby eliminating entire classes of attacks, such as buffer overflows (common in day zero attacks)
- Provides for application anomaly detection

Positive Security Model: Gateway Considerations

While clearly a more secure alternative to the negative security model, the positive security model failed to gain popularity not due to a flaw in the model but in the authors opining because the vendors supporting it failed to keep up with the rapidly growing number of applications that it needed to support in order to work in the business environment. Simply put, if the vendor did not fully support the application, the Positive Model–based security product would in fact break the application by not allowing it's associated traffic to pass, and effectively the organization utilizing the product was not able to conduct their normal business.

If we look back to 1988, only a handful of applications were necessary to be supported in order to conduct business:

SMTP

FTP

HTTP

Finger

Telnet

Gopher

Today, the numbers of application that are used in the enterprise environment have nearly reached 1000. Failure to provide support for any one of these applications could very well prevent an enterprise from conducting its normal business.

In the early days of Firewalls, vendors that provided application proxy technologies were the first to use the positive security model in a pitch for their products. But as the number of applications began to quickly outnumber those they could provide full application proxy support for, they themselves began to include negative security model–based technologies within their products in order to simply make them useable. Quickly, the line blurred between positive and negative security models for these products as you ended up with a limited positive security model for those applications proxies that could work effectively in a given environment, and a negative security model in the form of nothing more than a packet filter with signatures (IDS) layered on top of it to provide for the identification of known bad packets to support those applications that could not be supported in a positive security model.

Obfuscation renders negative security model–based firewalls obsolete: In a negative security model, all traffic is allowed to flow freely and only that traffic that is identified as "bad" is blocked. Back in 1999

when the rate of new vulnerabilities were running at only 25 new vulnerabilities reported each month, negative security model–based firewalls could offer a reasonable level of risk mitigation as vendors could "keep up" in creating new defensive signatures. In today's threat environment with an ever-increasing number of reported application vulnerabilities combined with the use of obfuscation, we are seeing 5,500,000 unique samples of malware annually. Keeping up with the number of necessary signatures has become a trying, if not impossible, task for those responsible for creating the necessary defensive signatures.

This signature problem was clearly exacerbated with hacking tools like "VOMM" also referred to as "Evade-O-Mastic" that obfuscates Web-based exploits rendering their attack completely undetectable by negative security model firewalls and their associated signatures. Negative security model firewalls are also plagued with being unable to detect self-mutating exploits like the Straton worm, which automatically alters its program code at a faster rate, than vendors can create new signatures. In October 2006, the Straton worm was the most prevalent worm reportedly seen by security vendors on the Internet. In the month of October alone, over 300 different variants of the Straton worm were detected. Today's exploit obfuscation tools and self-modifying exploits have effectively rendered negative security model firewalls as well as other signature-based defenses such as Intrusion Prevention Systems

(IPS) as obsolete.

From a firewall and positive security model perspective, Orange Book was clearly decades ahead of its time. Only now, decades later, in 2009 does a vendor seem to be "ahead of the curve" and able to support the nearly 1000 applications that may be found within the enterprise and provide for a positive security model without breaking the applications necessary for the enterprise to conduct their daily business.

Positive Security Model: Antivirus Considerations and Application Control Considerations

Malware regularly slips through current defenses wreaking havoc within enterprise networks. Antivirus products are struggling today with their inability to keep up with both the shear number of new virus and worms (malware) that are spawned out of the dramatic increase in newly reported vulnerabilities as well as the stealth employed by current technology self-mutating malware. Antivirus products have grown to be too dependent upon signatures for detection of malware and have not placed enough emphasis on newer technologies such as advanced heuristics that can detect malware without using an associated signature. Another issue has arisen that is increasing the difficulty of antivirus products from affording a reasonable level of risk mitigation—targeted attacks. In a targeted attack, the malware is not broadly distributed across the Internet, the delivery is reduced to a finite number

of targets. Antivirus vendors have grown accustomed to the luxury of the broad distribution of malware, affording them the opportunity to capture and reverse engineer the malware for signature creation early in the malware life cycle. In a targeted attack, it is highly unlikely that signature-dependent antivirus vendors will be able to capture a sample of the malware in order to create a defensive signature, hence they will be unable to offer any defensive capability. The failure of antivirus products to operate effectively without the use of signatures has perpetuated the rise in day zero attacks. This antivirus signature issue combined with the increased prevalence of targeted attacks, as compared to traditional broadly distributed malware, is quickly rendering many antivirus solutions obsolete.

Many are now beginning to recognize that the era of other negative security model–based products, such as traditional signature-based antivirus, is quickly nearing its end. A Positive Model–based alternative known as White-Listing or Application Control is quickly gaining popularity as a replacement for traditional antivirus solutions.

Table 1.1 displays the results of a recent test of AV software by Virus Bulletin. At first glance, one may conclude that a rating of 99.8% is quite effective. However, consider that the rating when applied to the 1,164,662 samples used in the test still allowed 2,329 pieces of malware through to infect a network. Consider the worst-case performance reported at 65.5% that left 401,808 pieces of malware through in the

testing. Now, to fully appreciate the scope of the issue, apply the ratings from the testing to real-world numbers, such as the current reported run rate of actual unique malware samples at 5,500,000 annually. With 99.8% effectiveness in your AV solution, you are still potentially allowing 11,000 pieces of malware to slip through—more than enough to devastate and/or wreck havoc in your network.

Negative Security Model

Also known as default allow—In contrast to a positive security model is of course the negative security model. The most popular network security products to date historically have been those that worked in a negative security model. Simply put, they rely on their ability to identify undesirable/known bad traffic and prevent it from entering. It is very much like having a list at a country's port of entry, which identifies known criminals. When people travel into the country, their passports are checked against this list, and if they are not on it, they are allowed in. This design is effective to the degree that it catches known criminals, but what about those who have not yet committed any acts of terror, or have not yet been caught for their crimes, or who should also be considered a risk because of their associations or reputation? In fact, it is not always possible to determine whether somebody, or in the case of the network, a particular packet of traffic, is undesirable based on known parameters. The most effective security policy revolves around one statement:

"Trust no one." That is why the best firewalls operate on a "positive" security model, which denies all access unless it is explicitly allowed.

Application Control

A current generation implementation of the Positive Security Model

Application Control is quickly emerging to complement and even, perhaps, to replace traditional antivirus solutions. Rather than relying on the constant creation of new signatures to protect assets from emerging threats in a Black List or negative security model, Application Control uses a positive security model or white list approach. In the simplest of terms, controls are established to permit or deny all applications and supporting scripts and macros on all workstations across the enterprise. This approach reminds the author of the Orange Book Default Deny methodology (in limited respects)—if the application, script, or macro is not explicitly approved via the Application Control policy and confirmed via hash, it by default is not permitted to execute.

From a risk mitigation perspective, the time has clearly come for the shift from the negative security model to the positive security model. In the simplest of terms, the number of new and potentially bad things that must be blocked in a negative security model implementation now easily outweigh those that need to

be permitted to facilitate the business needs of an organization. From an administrative burden perspective, the tide has turned and today it is simply more effective to manage the "known good" than to keep up with the explosive growth of the "known bad."

Table 1.1 Results of Test of AV Software by Virus Bulletin

Product	Malware on Demand (%)
AntiVir (Avira)	99.80
Avast! (Alwil)	99.30
AVG	95.80
AVK 2008 (G Data) (1)	99.20
AVK 2009 (G Data) (2)	99.80
BitDefender 2008	97.70
BitDefender 2009	97.60
CA-AV (VET)	65.50
ClamAV	88.50
Dr Web	84.90
eScan	97.80
Fortinet-GW	92.60
F-Prot (Frisk)	94.80
F-Secure 2008	98.20
F-Secure 2009	99.20
Ikarus	99.50

K7 Computing	92.10
Kaspersky	98.40
McAfee	93.60
Microsoft	97.70
Nod32 (Eset)	94.40
Norman	96.30
Norton 2008 (Symantec)	97.80
Norton 2009 (Symantec)	98.70
Panda 2008	86.40
Panda 2009	91.80
Rising	83.40
Sophos	97.50
Trend Micro	91.30
TrustPort	99.50
VBA32	90.50
VirusBuster	89.00
WebWasher-GW (3)	99.70
ZoneAlarm	97.80

Source: Data from *Virus Bulletin*, September 2008. http://www.virusbtn.com/news/2008/09_02

The five phases of the implementation of an application control solution are

1. Discovering and monitoring the application ecosystem

2. Assigning rights
3. Pilot rollout
4. Enforcing protection
5. Fine-tuning the application ecosystem

Discovering and Monitoring the Application Ecosystem

There are multiple approaches to establishing a baseline. One could use a third-party database of known good application hashes or simply create a custom database by scanning clean and known good machines (Figure 1.5) that had not been connected to the network or public Internet that contained the applications necessary to complete the business objectives of the organization.

Assigning Rights

In an Application Control solution, assigning rights can be as simple as assigning all validated applications and supporting scripts to a "Everyone Group" or can be accomplished with a high level of granularity for individual users and/or groups by leveraging existing LDAP, Active Directory, or eDirectory resources (Figure 1.6). Application Control solutions often provide for integration with IT change control solutions to reduce the administrative burden of ongoing system maintenance.

Pilot Rollout

A pilot group of users is normally selected to test the completeness and accuracy of the white list and system configuration. A good selection for a pilot group would be a set of users that does not include development and/or IT maintenance—related workstations as they typically run the most nonstandard applications. In the initial phase of a pilot program, the solution is operated in a monitor-and-report-only mode and does not implement enforcement.

Figure 1.5 A unique SHA-1 signature is calculated for each binary file, together with the filename, path, size, and product version. This information is recorded on the Lumension server whitelist, defining what programs can run on all selected computers.

Figure 1.6 The User Explorer module lets you use the Microsoft Active Directory to map users and groups to the whitelist.

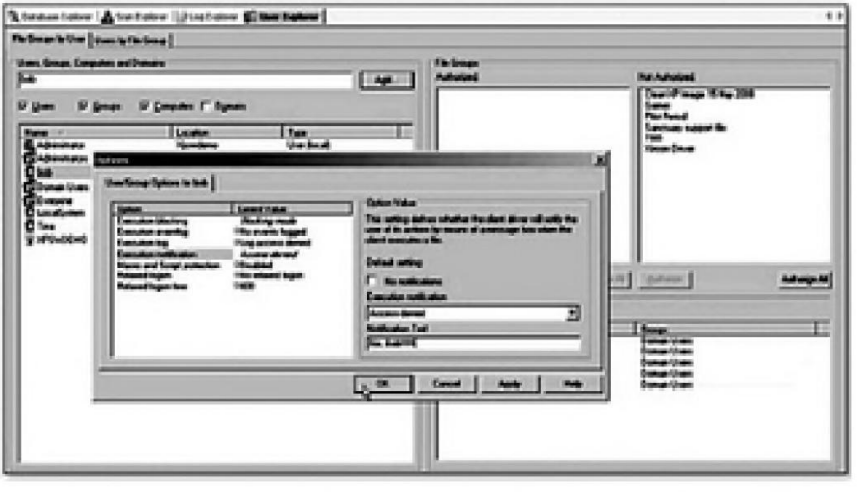


Figure 1.7 Use warning messages to explain why applications won't run after blocking is turned on—perhaps with more information than "No, Bob!".

Enforcement

In the pilot program, careful monitoring of exceptions will allow reconfiguration of policies to provide for operation with minimal exception alerts. As the exception list shrinks to a manageable level, the pilot can switch to an enforcement mode, whereby those applications that are not explicitly permitted for users/groups are denied, and exception reports are generated and made available to the management console, where adjustments can be made to the operating policy (Figure 1.7).

Once you are comfortable that you have developed a manageable configuration, the pilot can begin to be rolled out across the enterprise. Departments can simply be added initially in a reporting-only mode. Once confirmed to be configured properly, whereby a minimum level of exceptions are reported, the department can be switched to an enforcement mode of operation.

Fine-Tuning the Application Ecosystem

User awareness is a big part of a successful implementation of application control. Employees must be alerted as to which applications are approved for operation on departmental workstations and which are

not. If you have not prepared your employees for the change, your help desk could be overloaded. In addition, alerts can be configured to be displayed on the users' workstation to alert the users when they have attempted to run an application or script that is not permitted by policy. A carefully crafted message can go a long way in reducing help desk calls for assistance (Figure 1.8).

To facilitate an effective and manageable application control solution, the following capabilities need to be addressed by the application control vendor:

- Automated application discovery that provides flexible options to update white lists
- Spread check mechanisms that can automatically disable applications when it is discovered that too many users may have used local authorization to enable an application that potentially places the organization at risk
- Active directory integration to facilitate automatic maintenance of users and groups used within the application control policies
- Automatic authorization of vendor software updates to eliminate the risk of automatically restricting user community's access to frequently updated applications
- Script and macro protection to extend policy enforcement beyond applications to their supporting scripts and associated macros

- Flexible file authorization to identify new files to be included in the authorization database
- Local authorization for trusted power users to offer flexibility without giving up administrative control; any locally authorized application is reported to the administrator for review
- Off-line protection to ensure that remote or disconnected users are constantly protected by keeping a local copy of respective application hashes and remissions on each machine
- Standard file definitions that include classifications of all preloaded applications across all supported operating systems

Figure 1.8 You can write custom warning messages for users when they attempt to launch an application that is not on the whitelist.

The only question remaining for moving to a positive security model in the use of application control is

perhaps where it is best to implement it. Some would suggest application control is best handled at the gateway to reduce administrative burden and focus protection on a single set of protective devices. In the authors opinion, it is this kind of thinking that has brought us to the perilous point we are at today, whereby hackers that are able to pierce the perimeter defenses have an open reign within our networks. That thought along with the current increasing insider threat leads the author to conclude that application control is best accomplished in a layered approach both at the gateway and on the desktop.

An example of the benefit of positive security model–based application control:

The scenario:

Small network < 100 Windows XP Machines

Current antivirus at the gateway and on the desktop with the latest signatures

Firewall with a rule to permit internal users with Internet access over ports 80 and 443

URL filter to block access to known malicious Web sites

1. The internal user while surfing the Internet is redirected to an official-looking Web site that initiates a fake security scan of the users PC.
2. The URL was not blocked by the URL filter as the Web site had not yet been classified as a malicious Web site.
3. The page displayed on the user's PC (Figure 1.9)

did not include the typical browser tool bar and the user assumed it was an official company application scanning his PC.

4. When the scan was completed the user in the interest of quickly getting back to work selected to remove the malware that reportedly was found on the PC.
5. Upon selecting "Remove," the user's PC unknowingly downloaded malicious executable files to the user's PC.
6. The gateway antivirus server and the user's desktop antivirus software did not block the download as the downloaded files were using obfuscation and the antivirus signatures had not yet been updated to contain signatures for this new and seemingly one-of-a-kind obscured application.
7. Application control automatically blocked the operation of the executables by default (Figure 1.10) because they were on part of the permitted applications that the user had administrative permission to execute. Further, application control blocked the malware's attempted execution of a restricted command.

Figure 1.9 Rogue security software is a type of misleading application that pretends to be legitimate security software; however, it provides little or no protection and may install the very malicious code it purports to protect against.

Ironically, the user's application aware firewall vendor, URL filter vendor, and antivirus vendor claims their products are proactive in that they will automatically protect the user from malware. Unfortunately the firewall did not block the malware as it did not yet have a signature for the unique malware in it's internal application filter database, the URL filter did not block the user's access to the URL that contained the malicious page as it did not yet have the proper classification for the URL that the user was

redirected to, and the user's antivirus solution did not block access to the malware as it lacked a signature for the new and unique malware that was delivered. The user's so-called "Proactive Defenses" were unable to be anything that resembled "Proactive." The only real "Proactive Defense" the user experienced was their last line of defense—their positive security model-based application control.

Figure 1.10 Positive security model-based application control automatically blocked the operation of the rogue software executables by default because they were not part of the permitted applications that the user had administrative permission to execute.

Continue with

- Mandatory protection
- Use of data classification labeling
- Covert channel analysis

Mandatory Protection

Mandatory protection in the form of the enforcement of the rule of least privilege was a principle component of the Orange Book. It clearly defines least privilege as a principle that "requires that each subject in a system be granted the most restrictive set of privileges needed for the performance of authorized tasks. The application of this principle limits the damage that can result from accident, error, or unauthorized use."

Perhaps one of the most recent highly visible, albeit arguably failed, implementations of mandatory protection would have to be Microsoft Windows Vista user account control (UAC). Many would argue that one of the reasons Vista has not enjoyed the popularity obtained by its predecessor Windows XP is specifically due to the implementation of UAC. In fact, reportedly, a vast majority of Vista users have disabled UAC. An example of the annoying interruptions provided by UAC is the pop-up received when a user would attempt to open the common command prompt with administrative privilege (Figure 1.11) or attempt to run any program with administrative privilege (Figure 1.12).

The problem with UAC stems from Microsoft's legacy of giving every user administrative rights by default.

Simply put, UAC is perhaps Microsoft's first attempt at breaking away from the tradition of every user being an administrator by default. When a common user has standard user rights in Windows Vista, UAC will pop up each and every time the user attempts to do anything where their administrative rights are necessary. Perhaps it is too much nagging interaction with the user that fueled the distain for UAC. In a Mac or Unix environment, the user by default does not have administrative rights, hence the user is only prompted when they attempt to perform a task that actually requires administrative rights and they are silently prevented from doing other things that may require administrative rights.

Figure 1.11 An example of the annoying interruptions provided by UAC is the pop-up received when a user would attempt to

open the common command prompt with administrative privilege.

Figure 1.12 An example of the annoying interruptions provided by UAC is the pop-up received when a user would attempt to run any program with administrative privilege.

Mandatory Security in the Mainstream

The U.S. government's Information Security Automation Program (ISAP) is an initiative to provide for the standardization of technical security operations. Using tools that support the Security Content Automation Protocol (SCAP) enables literally pushing standard policies out to every desktop within an organization and then monitoring those machines for compliance with the policy. While you simply will not

find the words "mandatory security" within the ISAP specifications, clearly, it is a tool that can in fact be used to provide for an enterprise-wide baseline of mandatory security.

It is the policies that are used that in fact afford the baseline of the deployed security configuration—mandatory security. Further, ongoing monitoring of compliance can provide for automated compliance reporting that has become a requirement of standards such as the Federal Desktop Core Configuration (FDCC) and the Payment Card Industry (PCI-DSS) and other custom policy implementations.

A cursory review of a current FDCC policy configuration would include approximately 300 security-related requirements in a Windows XP or Vista computer. The immediate impact felt by users in an environment where FDCC policy is enforced is as follows:

- Password changes will be more frequent. Instead of every 90 days, your password will have to be changed every 60 days.
- Your login will not be saved when you log on. You will need to fill in the login and the password each time you log on to your computer.
- Administrative privileges will be taken away, which means you will not be able to download new applications. Unless you obtain a waiver to have these privileges, you will need to open a ticket with the Help Desk and have them work with your

local IT support to make changes to your computer or install software.

- Some applications may not work properly because they require administrative access to the operating system, application directories, and registry keys. For example, there is a known problem with Visual Studio Suite accessing files that only an administrator can access. It has also been reported that in some cases, Remedy is unable to access the user preferences, which are stored in the user's profile, which requires administrator access.

Checklists are freely available for various FDCC configurations and are downloadable for review at http://checklists.nist.gov/chklst_detail.cfm?config_id=129.

An example of the processes involved in automating mandatory security across an entire enterprise using a SCAP-compliant product (Figure 1.13):

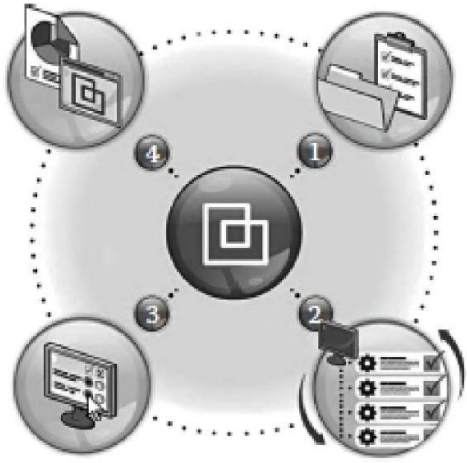


Figure 1.13 An example of the processes involved in automating mandatory security across an entire enterprise using a SCAP-compliant product.

1. Manage Security Configuration Policy: Define, edit, and import/export security configuration policies and best practices by leveraging the SCAP. Automatically map these regulatory or internal security policies to your own agent policy set, enabling you to standardize and secure your endpoint configurations and easily demonstrate compliance. Thanks to open standards, security specifications can also be added or edited to create custom security configuration policies.
2. Assess Policy Compliance by Group and Device: Apply desired security specifications to your network device groups and application

configurations. Automatically (or manually, where applicable) assess policy compliance with security configuration specifications for device groups as well as individual devices.

3. Report Policy Compliance Results: Demonstrate policy compliance by reporting configuration status against regulations and industry standards such as FDCC and PCI-DSS, as well as customized policies.
4. Enforce Policy Compliance: Achieve and maintain compliance with security configuration policies and best practices, leveraging automated remediation and policy enforcement.

With policies deployed across the enterprise monitoring of compliance can be an administrative burden, however, current generation SCAP implementations provide for a centralized graphical user interface (GUI) that can dramatically reduce administrative burden (Figure 1.14). Further, the centralized GUI provides for the necessary reporting capabilities that are a common component in today's regulatory environment.

Figure 1.14 Current generation SCAP implementations provide for a centralized graphical user interface (GUI) that can dramatically reduce administrative burden.

Some of the very same mandatory security requirements that were perhaps principle components of the original Orange Book requirements two decades ago have clearly finally found their way in to broad use today across both government and private commercial networks. One can only wonder what impact would have occurred had mandatory security become a reality back when the Orange Book requirements were first made available. Clearly, the unquestionable benefits of mandatory security and its components such as the enforcement of the rule of least privilege would have seriously altered the threat landscape we face today.

A recent study by BeyondTrust found that 92% of

critical Microsoft vulnerabilities could have been stopped or mitigated by simply eliminating the practice of giving users "administrator" rights. The study also found that eliminating administrator rights would have stopped or mitigated

> 94% of Microsoft Office vulnerabilities reported in 2008
>
> 89% of Internet Explorer vulnerabilities reported in 2008
>
> 53% of Microsoft Windows vulnerabilities reported in 2008.

Use of Data Classification Labeling

The usage of a data classification and labeling scheme was a core component of Orange Book security. Used in part to enforce mandatory access control (MAC) in environments requiring high levels of security, such as government or military systems. With MAC, the inherent problems of trying to rely upon each system owner to properly control access to each access control object is eliminated by having the system participate in applying a mandatory access policy (the system owner applies the "need to know" element). This policy affords typically three object classification levels: top-secret, secret, and confidential. Each access control system subject (users and programs) is assigned clearance labels, and access control system objects are assigned sensitivity labels. The system then

automatically provides the correct access rights based upon comparing the object and subject labels. Mandatory access controls allow multiple security levels of both objects and subjects to be combined in one system securely.

Today, data classification and labeling schemes are proving to be beneficial in data leakage prevention (DLP) systems as well as the automation of rediscovery and deduplication efforts.

A data classification and labeling system can go a long way in making organization's efforts to identify and secure their data more effective. The use of unique labels for files containing sensitive information can make it easier to find them, whether they are at rest or in transit.

These unique labels can be literally used like digital watermarks, and you can write IDS rules to identify them when the data is in transit. You can also use regular expressions for the tools that are used to search data at rest, using them to identify these labels and to help audit computers that should not be storing a particular classification of data.

Labeling data in eDiscovery and deduplication has been proven to be beneficial in reducing the complexity of fulfilling rediscovery efforts within large organizations. The shear volume of data contained within the enterprise has seen explosive growth (Figure 1.15). It has become common today for data labeling for data classification to become part of the normal

business data life cycle, (Figure 1.16) and in at least some abstract respect has its heritage traceable back to Orange Book security initiatives.

Covert Channels

Covert channels have long been the enabler of communications for the command and control of botnets and most recently have been adopted for the theft of data in data leakage incidents.

A "covert channel" can be described as "Any communications channel that can be exploited by a process to transfer information in a manner that violates the system's security policy." Essentially, it is a method of communication that is not part of an actual computer system's design but can be used to transfer information to users or system processes that normally would not be allowed access to the information.

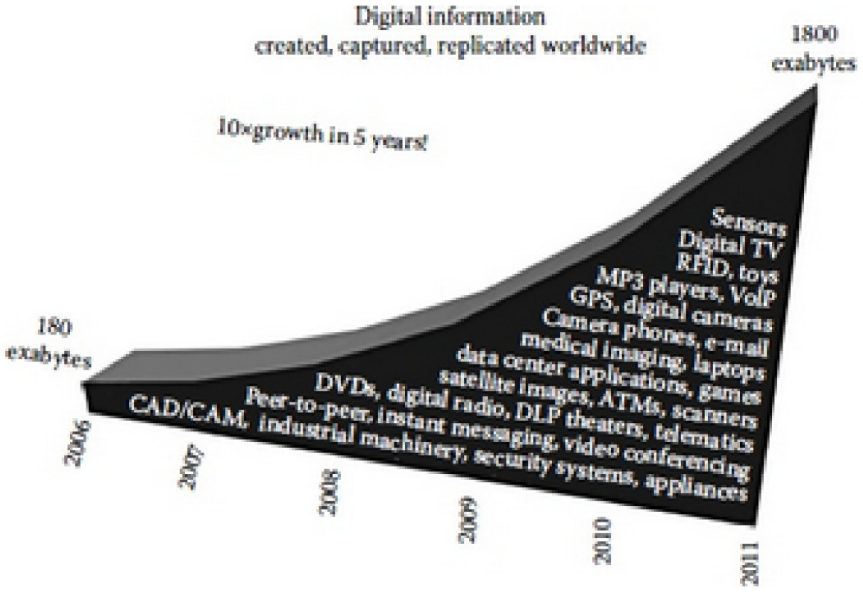


Figure 1.15 The shear volume of data contained within the enterprise has seen explosive growth. (Courtesy of IDC White Paper, *The Diverse and Exploding Digital Universe*, sponsored by EMC, Framingham, MA, March 2008.)

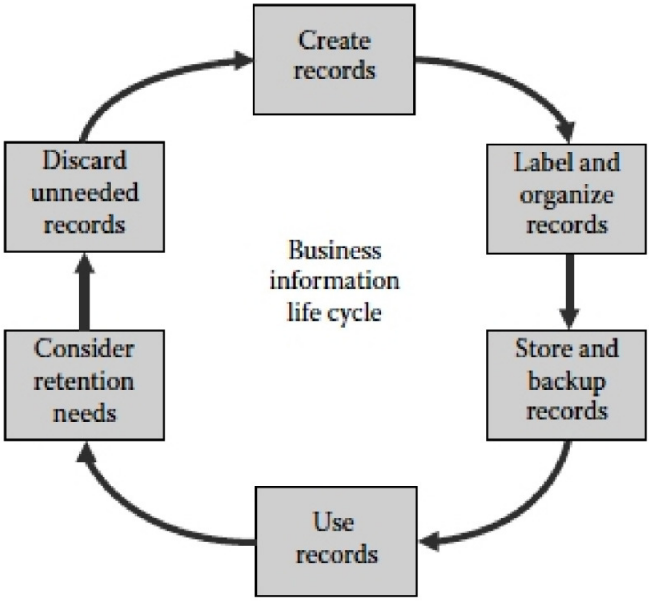


Figure 1.16 The normal business data lifecycle.

Covert channel exploits typically require a malicious client or server program operating on a PC outside the protected network and a malicious server or client program operating on a server inside the protected network.

The malicious PC outside the protected network would encapsulate the desired protocol within a given protocol that is allowed by the security policy of the protected network's firewall. The malicious PC on the outside of the protected network would then transmit this allowed protocol through the firewall directed to the IP address of the server running the malicious

receiving program inside the protected network.

The receiving program would strip off the transport protocol, thereby leaving the original malicious data in its original protocol form. These packets would either then be used by the server running the receiving program or be automatically sent to a predetermined IP address of another server or PC within the network.

Covert channels are not a new methodology; in fact the theoretical dangers of covert channels were first addressed in the National Computer Security Center's (NCSC) Trusted Computer System Evaluation Criteria (TCSEC) as early as in 1983 and 1985.

Later in 1990, as covert channels moved from the realm of theoretical to possible, in France, Germany, the Netherlands, and the United Kingdom, a testing methodology for covert channels was developed and published: Information Technology Security Evaluation Criteria (ITSEC). In the mid-1990s, as covert channels moved from the realm of possible to probable, many papers were published outside of government that explicitly detailed covert channel exploits at the application level and, in many cases, provided working source code to build a fully functional covert channel.

One of the most recent adaptions of covert channels involved a data theft incident whereby stolen credit card data was simply added to the payload of a DNS request to move the stolen data out of the network without raising any suspicion from the network administrators.

Covert channels, in their simplest form, involve encapsulating a particular protocol that if used directly would raise suspicion over a protocol that is considered normally permitted traffic. A timely example would be IRC traffic: if an administrator saw IRC traffic flowing across the network, it would immediately raise suspicion as it has long been associated with nefarious activities. It is trivial to encapsulate the IRC traffic over the normally permitted HTTP traffic that flows across the network and escapes the detection of the network's security mechanisms.

Simply put, firewall vendors have known since the late 1980s about the risk of covert channels and yet only a select few actually did anything definitive, such as adhere to Orange Book standards for risk mitigation of Covert Channels. The problem is much broader when you consider that the vast majority of security mechanisms actually aid in facilitating Covert Channels by strictly adhering to a port-centric view of network traffic.

A good example of this failure would be the evolution of Instant Messaging (IM) and our early reliance on blocking the specific ports used by the specific IM client. IM client providers quickly realized that continued use of specific ports would allow security administrators to block their usage, so they adopted the capability to use not only the typically defined port associated with the IM client but also literally any port it could find open and available through the network's defenses. Today, you will find

nearly all IM applications tunneling their traffic over the most commonly open port in a network gateway—port 80 (normally associated with HTTP).

The shortsightedness that a port-centric view brings to firewalls is still a problem today as the most popular firewalls currently in use are limited to applying their protective policies only against the specific port numbers normally associated with a specific traffic flow. Once you create a rule to allow traffic to flow through your defenses, such as opening port 80 to allow your users to have internet access with their browser, any traffic including traffic from any malicious applications that may be residing within your network have the ability to use that open port and simply encapsulate their traffic on top of the HTTP protocol and evade your ability to block it.

Historically, the author is only aware of a small number of legacy firewalls such as the Cyber Guard Firewall and Secure Computing Sidewinder Firewall (who's legacies can ironically be traced back to Orange Book) that afford the necessary application awareness to allow a specific application to use a specific protocol and to automatically deny by default any ability for a foreign protocol to be encapsulated across that permitted protocol. However, these firewalls are severely limited by the small number of protocols they can fully support, and in the current environment, some simply find them unusable as they have a tendency to break applications that they are not fully aware of.

Today, a different approach in next generation

firewalls is quickly taking shape: The PaloAlto Firewall, which was noted earlier in this chapter as having the ability to control any of the nearly 1000 applications necessary to conduct business in today's enterprise environment, can be found operating literally over any port or service. Simply put, it does not classify traffic based solely on a given port or service but identifies the protocol using a combination of heuristics and learned behavior on the wire regardless of any inference to a specific port or service. This eliminates the port-centric issue of traditional firewalls and returns our ability to construct a workable deny-by-default approach to network security, whereby only those applications specifically permitted by policy are permitted to traverse the network regardless of the port or service it happens to be operating over.

Another promising approach to the covert channel issue takes a completely different approach to the problem. Rather than trying to control the ability of a malicious or undesired application from being able to move traffic across the network, Lumension Application Control eliminates the ability of the application itself from being able to operate on the workstation in the first place. The application fingerprint and centralized management approach utilized all but eliminates the administrative burden one might associate with the methodology. Using this approach brings back the ability to construct an enterprise network that is able to operate in a default deny state for any application that is not explicitly permitted by the organization's security

policy. Able to support thousands of current generation and legacy applications, it can operate without the issue of legacy solutions that suffered from a severe limitation of their breadth of the applications associated with a modern enterprise.

With the ability returning to effectively enforce a default deny policy for the enterprise, we can again begin to address the covert channel issues identified decades ago in the Orange Book. For most, the remaining question is where is it best to address the issue—at the gateway or on the endpoint? In the others, opinions have been long convinced that a layered approach to network security is the only effective approach and the issue should be addressed both at the gateway and on the endpoint.

In Closing

Long ago, those responsible for network security for those networks connected to the public Internet made a decision in a relatively immature and low-threat environment where unique threats on the wire could be measured in the hundreds annually. That ease of use and performance were more important than the ability to control a wide range of seemingly distant security threats. The security provisions of the Orange Book were relegated as being too difficult to implement, too CPU intensive to offer acceptable performance, and for the most part relegated as "old school technology" with no place in the modern enterprise.

In our current threat environment where threats are now exceeding over 5,500,000 unique malicious threats annually on the wire, current generation solutions for antivirus and firewalling have been found to be nothing short of completely overwhelmed and simply unable to scale to meet our future defensive needs and are effectively obsolete. Further, many of the risks originally addressed by the security provisions outlined in the Orange Book standards have come to fruition and unfortunately remain even today, two decades later, unsolved by the most popular solutions available in the security marketplace.

Perhaps the author is simply showing his age by taking a nostalgic look back on what could have been but cannot help but wonder what the threat environment we face today would be like if we had simply adopted the principles outlined in the Orange Book some two decades ago. It would most certainly not resemble the mess we currently find ourselves in today.

About the Author

Paul A. Henry, MCP + I, MCSE, CCSA, CCSE, CISSP-ISSAP, CISM, CISA, CIFI, CCE, ACE, GCFA, is president of Forensics & Recovery LLC, Ocala, Florida.

TELECOMMUNICATIONS AND NETWORK SECURITY

DOMAIN 2

Communications and Network Security

Chapter 2

Adaptive Threats and Defenses

Sean M. Price

Contents

The survival of living organisms is often dependent on their ability to compensate for changes in their environment. The ability of an organism to compensate for changes encountered is referred to as adaptation.

Predominately, the methods of adaptation involve changes in the organism's behavior, physical characteristics, or both. Some creatures are able to learn new skills or tricks that allow them to cope when changes occur. In other cases, an organism might undergo a genetic mutation that provides it with a slight advantage over its rivals allowing it to survive better given the changed conditions. Adaptation can also occur with the combination of altered behaviors and new mutations. The ability to adapt is also exhibited in the cyber realm by threats and defenses. This chapter is primarily focused on the adaptability of attacker malware and defender security tools.

Threats and defenses have evolved over the years. The emergence of the first forms of malware and hacker tools was followed by defensive tools and techniques. As new methods of attack are pursued, defensive measures arise to counter the threat. This constant struggle between attackers and defenders is sometimes referred to as an ongoing arms race (Carlsson and Jacobsson 2005). The goals of attackers and defenders are equally opposed to each other. Attackers seek to exploit a system while the defenders attempt to prevent compromises. The objectives for each of these competitors could be summarized with the following:

Threat objectives

- Discover new weaknesses
- Exploit new and old vulnerabilities
- Hide presence

- Retain a foothold in compromised systems

Defense objectives

- Counteract known threats
- Detect deviations from normal activity
- Identify abuse of the system
- Mitigate known vulnerabilities

Evolution of Threats and Defenses

Over time the objectives of threats and defenses has not changed much. However, the methods used to achieve their objectives have substantially evolved. In the early days, threats were single purpose and could be generally categorized according to its attack vector. Initially, the taxonomy of malware was predominately marked by viruses, worms, backdoors, keystroke loggers, and Trojan horses. Human threats included hackers, crackers, and social engineers. Adaptations soon appeared with the emergence of malware such as spyware and remote-access Trojans. Similarly, the human threat evolved with the new uses of spam and phishing techniques. More recently, threats and defenses began to exhibit adaptability by use techniques from different categories (Geer 2005). The use of multiple categories is regarded as a compound threat or defense.

Attackers quickly learned that combining attack vectors enabled deeper penetration and more automation. Malware authors began to incorporate a

variety of attack methods into their code. Instead of a worm simply infecting one system after another through a single exploit, it would drop packages enabling further compromise of the system. Bots, for example, are a recent evolutionary step in malware that are perhaps the most troubling. They automate much of the manual activity previously accomplished with hacker tools.

To a lesser extent compound defenses have emerged. Many security products now incorporate multiple defensive measures such as antivirus, anti-spyware, phishing filters, spam blockers, and firewalls (Greiner 2006). These efforts appear to be more about consolidation and rivalry between the products of security vendors as opposed to focused efforts to compete against malicious code. The impact of compound defenses seems much less substantial than the effect of compound attacks.

Adapting Threats

The ability of a threat to retain its relevance is strongly tied to its capability to adapt. Automated and external human threats often exploit a weakness to gain further access within a system. As weaknesses are corrected or countermeasures put in place, the relevance of a threat is diminished. Threats must change their malware to adapt to these changes that prevent or restrict their tools from performing their devious tasks.

It is important to bear in mind that automated threats such as malware are largely dependent on the hackers

that code them. Aside from polymorphism, malware adaptations are strongly linked to human intervention. Yet the source code influencing the polymorphism is a human-generated response that is driven by the need to adapt as a method of detection evasion (Christodorescu and Jha 2004). In the not too distant future, malware integrated with machine intelligence may be capable of generating original source code, discovering new vulnerabilities, and create unique methods to exploit any vulnerability. Although some might suggest that view is close to reality, there is little evidence to suggest that machine intelligence is close to achieving this level of abstract cognition. Nevertheless, it is likely that some malware will incorporate some or all of these attributes on a limited scale.

The evolutionary nature of threats is manifested from two points of view. First, changes in the behavior of the threat provide one method in which an adaptation can be achieved. In this regard, behavior reflects the actions malware takes to achieve its objectives. Relevant actions include file operations, registry manipulation, and network activity (Lakhotia et al. 2005). Process spawning is also pertinent to behavior as well. The second point of view is that of mutation. The predominate manifestations of a mutation involves changes to the code logic or structure of the binary. Changes in behavior will likely induce mutations in the underlying code. However, a mutation is also a tactical maneuver supporting adaptations that allow it to avoid detection.

Behavioral Changes

The actions and activity of a threat is an indicator of its behavior. The longer a threat uses the same behavior the more likely it is that defenses will detect and deploy countermeasures against it. Threats, therefore, adapt by changing the methods and techniques used to attack and retain a stronghold in a victim system. Continuous adaptations in malware, such as bots, are a common occurrence (Holz 2005).

Attack Vectors

Attackers regularly seek new methods to accomplish their objectives. An attack vector is the methods and techniques used to exploit a particular vulnerability. It is essentially the cumulative steps taken to exploit the flaw. For any given vulnerability there may exist a multitude of ways to exploit it (Ma et al. 2006). Threats can adapt their attack vector behavior by modifying the actions pursued to compromise the system.

Vulnerability Exploitation

A threat agent may attempt to exploit one or more vulnerabilities to achieve its objective. In time, due to awareness and flaw remediation, a targeted vulnerability might disappear, become irrelevant, or prove too difficult to effectively exploit. To remain pertinent, the threat must be capable of choosing different vulnerabilities to attack. A changing list of vulnerabilities to choose from provides the threat with a means to alter its attack behavior. The affect of this

behavior enables the threat to adapt to environments where some vulnerabilities are mitigated. Bots, in particular, are often coded with the capability to exploit multiple vulnerabilities (Geer 2005). Having the ability to select among vulnerability options can also make it more difficult for defensive mechanisms to target a particular threat.

Command and Control

Much of the malware in the wild today rely on some form of command and control. This enables the threat agent to communicate with and direct the activities of the malware. With respect to botnets, command and control is recognized as an important aspect of their value (Schaffer 2006). The three main types of communication methods can be categorized as follows:

- Independent—In this category, a malware opens a communication channel and listens for commands. The listening activity could be TCP, UDP, or both types of ports. In these cases the malware threat does not know where its command will come from.
- Centralized—Some malware know how to contact their master. This could be a particular Web site or e-mail address, but the most common is an Internet Relay Chat area. In these cases the malware looks to a primary address to receive commands.
- Decentralized—One trend among attackers is to organize the malware as a collective entity. This has the advantage of ensuring the malware can

survive and can make it more difficult to detect the command and control origin. This type of control is similar to peer-to-peer (P2P) networks.

A threat can adapt its behavior within each of these categories. For instance, malware using an independent method of command and control could change the port on which it is listening. It may also change the application protocol used, imitating other known services or something completely novel. Centralized threats can exhibit behavior change by contacting different or new centralized command centers to obtain instructions. Lastly, decentralized malware might try to mimic legitimate P2P, change its underlying application protocol, or use encryption. It is worth noting that a sufficiently "intelligent" malware agent might be capable of selecting among all three categories. This type of behavior could make it more difficult for network monitoring to detect its presence.

System Interaction

In most cases, a threat exploiting a weakness results in the appearance of executable binaries on the compromised system. The binaries are usually standalone, but could be attached to other objects, in the case of a virus. These malicious software components may interact with the system in a variety of ways. Some of the most common instances follow:

- Executables—An executable file runs as an independent process. The threat might be contained

entirely in the executable or it may rely on it as a way to initiate other activities such as downloading other malware.

- Extensions—A malicious library could be used to extend existent malware or have it loaded into legitimate applications allowing it run more discretely.
- Injections—Similar to extensions, a library might be injected into the execution space of another process. Some viruses create their own thread of execution within the host process. Although it is not truly an injection, the execution of a virus has many of the same implications.
- Rootkits and drivers—Malware at this level has the capability to hide its activities or that of participating malicious processes.

Threats can alter their behavior by interacting with the system through any and all of the aforementioned methods. By changing the way a threat interacts with a system it increases the likelihood that it will either avoid detection or make it more difficult to remove.

Storage and Configuration

The code enabling the threat to execute on a system must exist somewhere in storage and also requires some configuration method to prompt its activation. Threats will alter their behavior by a variety of methods that obfuscate their storage locations. They may change their file names and/or extensions to hide their presence.

Configuration entries are generally needed to assure that the threat is launched regularly.

- Obscure file and directory names—This can include names that are randomized. Some threats create random names while others select from a pre-populated list within the malware package.
- Alteration of registry entries—Some threats create their own entries or rely on the entries of legitimate software.
- Alternate data streams—Malware hiding in an alternate data stream is not easily observed with standard system tools.
- Changes to configurations files—This is similar to the methods used for registry entries.
- Masquerading as legitimate files—A threat might select a file name that is similar to legitimate software. In some cases, the threat might actually rename or replace a legitimate binary file. The malware then is loaded whenever calls are made to the replaced file. In these cases, the threat will proxy the requests to the actual library whether it is in another file or is included with the threat itself.

Malware will often change their storage locations and configuration methods to remain a step or two ahead of defensive countermeasures. This adaptive behavior of modern threats enables it to persist within a compromised system.

Recruiting

A small number of threat agents can multiple their capabilities by duplicating their efforts. The common denominator of this behavior is to entice people to execute their malicious code. Malware are increasingly acting as the intermediary between the human threat agent and the human victim. A number of enticements are commonly used to recruit new victims.

- Trojaned freeware—An offer for a new free toy with hidden strings attaching to the victim's system.
- Pornography—The promise of a gratuitous glimpse into an act of indiscretion through a link or attachment.
- Financial—A lure to easy riches that turns out to be true for the attacker.
- Spyware—The participant is promised reduced rates or access to a particular application by using a particular software product. Often times, much more is disclosed than what was agreed to.
- Scareware—The end user is informed that their system is infected with malware and encouraged to download an antivirus tool to help clean their system. They are commonly enticed to purchase the fake antivirus product. This is reportedly a big income generator for many attacks.
- Phishing—Masquerading as a legitimate entity, such as an online bank, but really duping the e-mail recipient into disclosing their private information.

- Problem solving—Unwitting participants solve reverse Turing test problems, such as CAPTCHAs, that are too difficult for malware to deduce. Often this is used with other enticements such as pornography.

Threat Mutations

Code updates to malware has parallel attributes to evolution in living organisms. In time, a given piece of malware must adapt or it will be more readily recognized by defensive measures such as antivirus and anti-spyware tools. Mutations in this regard are essential for the malware to maintain relevance. Mutation to avoid detection is a common malware tactic (Edge et al. 2006). The following summarizes some of the reasons why threat mutations are an aspect of adaptation.

Defeat Signatures

An unchanging or static nature of an attacker makes detection easier over time. Researchers and security product vendors constantly seek the telltale signs and behaviors of attackers. Once the attributes are learned the data is compiled into tools and techniques that can be used to detect the presence of an attacker. Threats must, therefore, change their code and alter techniques to defeat signature analysis. New versions of the malware or polymorphic techniques are common methods used to defeat signature based defenses (Hsu et al. 2006).

Code Improvements

Some malware is just plain buggy. It is not uncommon for malware to cause poor performance or even disrupt applications (Schmidt and Arnett 2005). In recent years, the shift from hacking by rogue amateurs to those of nation-states and organized crime are accompanied by improved code reliability. Today malware is less likely to affect performance. However, it is important to consider that operating system improvements may have also contributed to more stable performance even when misbehaving applications are present. In any case, attackers will regularly attempt to upgrade their malware allowing it to better adapt to its environment.

Detection Avoidance

Overtime malware methods have become more sophisticated. Much of the efforts for improvements are related to techniques that hide the presence of the malicious code. Malware adaptations are increasingly disguising their activities to mimic legitimate system and network activity (Borders et al. 2006). Several years ago much of the malware resided in a single executable or may have included a small number of libraries. The existences of these tools were often readily observable in the file system and could be seen as distinct processes when executing. The next evolutionary jump emerged as add-ons to existing products. Malware increased the ability to cloak their activities by taking advantage of legitimate software features that enable extensions. Examples of these include system hooks, add-ons for

office productivity software, and browser helper objects. By running as a loaded module the malware avoids detection of some process monitoring techniques, but are still observable through system tools. In case of extensions the malware plays by the rules of the operating system. In contrast, other techniques such as vulnerability exploits and process injection are used to force a target application to run the attackers code of choice. These approaches to detection avoidance are more stealthy and not easily identified. The latest evolution of detection avoidance involves the use of rootkits and system drivers. These methods allow the tool itself and accomplice malicious processes to operate largely undetectable by most system and security tools. Malware with stealthy capabilities hide their behavior by intercepting and filtering application programming interface (API) calls that could be used to reveal their presence (Wang et al. 2005). The trend of malware and attackers has gone from brazen attacks defacing popular sites or sensational attacks against a well-known Internet presence to discrete compromises as chilling as any clandestine espionage activity could achieve.

Added Capabilities

New features incorporated into malware increase its value and potentially expand the influence of an attacker. Increased capabilities represent a maturation of the malware, which is a type of adaptation for survival. For example, an update might give the attacker the capability to scan other hosts for weakness or act as

a relay for other malicious activity. Increased capability enables the threat to adapt to an environment and potentially sustain or propagate its existence.

New Objectives

An attacker may periodically change targets or attack vectors. This is a common occurrence in botnets where the bot-herder rents out the zombies to service their customer requests (Geer 2005). The ability to change objectives is a tactical adaptation that makes for a superior weapon. Older malware usually had limited objectives that were not altered. Nowadays malware can attack new targets using different vectors or exploits through the receipt of software modules embedded with the new objectives and commands (McLaughlin 2004).

Upgrade Survival

Overtime, systems are upgraded or reinstalled. Threats must be able to adapt to new technologies for the relevance to remain. As an example, a system owner might migrate from one technology (i.e., e-mail client or Web browser) to another. An adaptable threat will be able to accommodate the change and continue unimpeded if the migration represents a vector for exploitation. Upgrades to the underlying operating system can also affect the ability of the threat to endure. Adaptable threats anticipate or respond to these changes through code changes that allow their existence to continue. Although, a threat outside of a supply chain injection may not be capable of surviving a fresh

installation of the OS or applications, it can attempt to persist by incorporating itself with legitimate applications and data. A threat that carefully infuses itself with data and applications targeted by managed backups enable malware longevity due to upgrades.

Self-Preservation

It is not uncommon for malware to disrupt, disable, or destroy security controls in a system. Some aggressive threats will alter access controls to protect themselves. Others reportedly disable host-based firewalls and antivirus software (Abu Rajab et al. 2006). This sort of activity ensures communications with the threat agent will remain intact. Yet, other malware may be so bold as to delete programs or audit data that could be used to detect or disrupt it. The techniques and methods used for self-preservation must adapt according to changes in technology and the environment of the compromised system.

Competition

Imagine that a zombie computer is under the influence of different bot-masters. Serving multiple masters might produce erratic behavior on the machine. There appears to be an unspoken consensus in the evil realm of malware creators that a zombie should not exhibit personality disorders. This perceived consensus is most likely imaginary. In reality, some malware attack and remove other malware (Osorio and Klopman 2006). Additionally, some malware reportedly patch existing

vulnerabilities (Abu Rajab et al. 2006). The reasons for this competition probably include rivalry, dominance, or economic advantage. In this regard competition among living organisms spills over into the cyber realm and is witnessed as malware on malware attacks. The escalation of malware competition is yet another dimension of the adaptable nature of threats.

Adaptive Defenses

Agile defenses are necessary to counteract adaptable threats. Defensive countermeasures are continuously challenged by the rapid changes they must deal with. On one hand defenses must adapt to changes in their environment. Network expansions and new technology can easily introduce exploitable weaknesses. On the other hand, threat aggressiveness continues to escalate. The rapid evolution of malware puts continuous pressure on defenses to adapt. From the perspective of a defender, adaptation is an imperative that must meet the challenges of environmental changes while remaining competitive with adaptive threats.

Attackers continuously conduct new and inventive assaults on network defenders. New attack methods brought about by malware adaptability are met with adaptive defenses. The discoveries of new attacks are often shared in the security community. Conjectured exploits by security researchers or actual exploits discovered in the Internet are reported by numerous public and private organizations. This new information is often integrated into defensive countermeasures

resulting in an adaptation to the threat.

Defenses exhibit adaptation through behavior modification and mutations. The objectives of defenses tend to be more reactionary to threat activity. In contrast, the adaptations of threats are more exploratory and proactive. In this regard, defense adaptations tend to lag those of threats.

Behavior Modification

Defensive controls face more challenges than do their nemeses. Tools used to defend a system require behavior modifications to account for changes to the environment as well as proliferating and changing threats. Behavior modifications entail the methods used to accommodate the rapid changes in the organization, technology, and known threats.

Frequency

Adaptive defenses may alter the period with which they conduct their surveillance. For instance, vulnerability scanners might ordinarily be used on a monthly basis. If the organization is experiencing substantial growth or more frequent compromises then the frequency of the control is increased. The timeframe between the moments of detection activity presents opportunity for a threat to attack a system. Increasing the frequency behavior is an adaptive approach to counteract rising malware instances.

Breadth

Security controls are not always deployed in every possible location in a network due to resource constraints. Furthermore, a control might also peer into a narrow band of activity in its attempt to identify attacks. In some cases, the purview of a control can be expanded to cover a larger area. This could be through increased instances in a system or through expansion of the band of activity monitored. Altering the breadth of a control impacts its behavior. Essentially, an increase in the horizontal nature of the control adapts the behavior of the defensive mechanism to detect malicious activity.

Depth

Viewing activity in a system from the perspective of the Open Systems Interconnection (OSI) model provides a vertical perspective regarding the behavior of a security control. A security control might ordinarily operate at only one layer of the model. An adaptive defensive tool might occasionally perform inspections at other layers of the model to detect attacks or actual compromises. This type of capability demonstrates a change in behavior that could be very useful in detecting adaptive threats.

Indicators

Many defensive mechanisms rely on precompiled indicators or signatures of known attacks. Adaptable defenses have the capability to compare system activity against new and prior indicators to detect active attacks. Adaptability through indicators is one of the most

predominate behavior modifications of defensive components.

Baselines

Well-managed systems have a number of documented baselines. These include baselines for hardware, software, network connectivity, and configurations. Defensive tools with the capability to detect system components and configurations can validate baselines. Security controls with the ability to make comparisons between system changes and a documented baseline exhibit adaptable behavior.

Learning

Perhaps the most intriguing representation of behavior modification occurs when a security control actually learns something. Machine learning techniques are commonly found in security tools designed to look for anomalous activity. Two common implementations of machine learning are used by intrusion detection and spam filtering. Intrusion detection products make use of neural networks and support vector machine algorithms (Mukkamala and Sung 2003). Spam filters typically use Bayesian techniques (Pelletier et al. 2004). In both cases, the tools learn what is normal versus what is not and raise alerts when anomalous features are encountered.

Interactions

In the near future, technologically advanced security

controls will receive data from multiple sources. This will provide the security control with the ability to form a more coherent picture of the cyber landscape. A security control with this advanced capability would be able to make predictions or advise human counterparts and other participating security controls of the current security state of the system. According to the situational awareness from the influx of data these advanced tools will exhibit behavior unlike its archaic predecessors. The interaction among these tools will form a collective that shares threat information and alters its behavior accordingly. For now, we rely on the interactions and sharing among humans to influence the most robust defensive control—the information system security professional.

Defense Mutations

Most of the behavior modifications are realized through mutations of the affected defensive control. Adaptation by way of mutation is for the most part straight forward regarding security controls. It is important to note that a mutation need not necessarily be compiled. The inclusion of any sort of logic enabling adaptability qualifies as a mutation. Some of the most prominent mutations employed to achieve adaptability follow.

Signatures

Features, behaviors, and characteristics of malware and indicators of threat activity comprise attack signatures. In many instances the files containing the signature

information are compiled into a library or proprietary data file. From this perspective, the inclusion of the new signatures results in a mutation of the defensive tool.

Rule Sets

This type of mutation is comprised of multiple "if–then" statements. Rule sets permit the logical evaluation of witnessed activity. Some rule sets are ordered to form logic trees. This allows for granular decisions based on collected information. Changes in the environment or attacker behavior may require changes in rule sets. Altering rule sets according to changes or trends enables adaptability through this type of mutation.

Thresholds

Cumulative events can be used to activate security controls. For instance, a control monitoring access control failures might not raise an alert unless the number of failures exceeds 10 events in less than 1s. A threshold of this type is designed to detect malware behaviors given the successively repetitive failures in a period of time too fast to be driven by a human manipulating a graphical user interface. The composition and attributes of thresholds can be changed to adapt to new threats or changes in the behaviors of known malware.

Defensive Adaptation Weaknesses

The traditional model used by defenders has been to

shore up weaknesses or adapt to threats by deploying new or modified tools counteracting the particular threat. Often times this can be over an extended period of time after a vulnerability is disclosed and exploit code is available. Sadly, adaptive defenses are primarily reactionary. Defensive measures usually target specific types of attacks deemed imminent. Rarely, will an organization incorporate a new defensive measure that is not focused on a particular threat or attack vector that has not been experienced by the organization or industry. The main reason for this line of thinking has to do with risk. An organization may deem a particular threat, likelihood, or loss to be minimal. Due to the prevalence of risk management by way of qualitative assessments, coupled with the ever-present problem of scare resources, it is not uncommon for managers to be optimistic about their level or risk. As such, adaptive or forward thinking defenses are not commonly deployed. This has the unfortunate side effect of causing defensive countermeasures to play catch-up with the attackers. This is evident by the relentless cycle of patching and signature updating.

Specificity

Defensive measures often rely heavily on specific signatures. The effort required to adapt to a new threat may be greater than that needed by the threat. Signature development can also require substantial time and effort to compile that can significantly lag a threat that is rapidly propagating (Edge et al. 2006).

Considering the time and resources needed to detect, develop, and deploy an adaptation for a given threat it seems that attackers have the upper economic hand.

Timeliness

The creation of a countermeasure for a given threat may be well after significant damage occurs (Cui et al. 2005). In some cases this reactive adaptation can be too little too late. It is now common for exploits for previously unknown vulnerabilities to be found in the wild (Levy 2006). In this regard defensive adaptations are entirely reactive with respect to weaknesses and exploits.

Growth Rate

New threats are beginning to emerge at a rate faster than security defensive measures can adapt. A recent estimate claims that the volume of malicious code exceeds the production of legitimate software (Nachenberg 2008). One implication of this growth rate is that defenses may consume substantially more resources to determine if a threat is present or not. This reduction in efficiency will likely inhibit the ability of the defensive measures to adequately adapt to the ever-increasing number of threats. Furthermore, the sheer volume may also imply that a larger number of malware is circulating in the wild that are unknown to defensive product vendors. This is to suggest that false negatives (malware missed by detectors) will increase. A substantial growth rate has the effect of overwhelming

our defenses by a numerically superior enemy.

Environment

Changes in the system environment present unique challenges. Growing organizations regularly add new network equipment to accommodate a growing user base. New technologies enabling increased productivity may also include new vulnerabilities. Defenses must not only adapt to system growth but new technologies as well.

Search Space

Adaptations that attempt to characterize what is normal in a system often fail due to complexity. Defensive techniques such as anomaly detection are commonly designed to look at everything to identify features that are not normal. This requires the defensive tool to look at the entire universe of possibilities. Tuning is often used to increase performance by reducing the search space. However, the search space is often still too large allowing false positives to persist. In some cases, items ignored by the rule set can be abused by attackers and thus avoid detection.

Constraints

Whereas malware authors are free to attempt anything desired to achieve their objectives, defenders are much more constrained. The adaptability of defensive measures is reduced due to factors in their environment.

- Financial—Adaptability often requires a monetary tradeoff. Whether the cost involves time, people, or materials the lack of sufficient financial resources can constrain defense adaptations.
- Personnel—Adequately trained people must be assigned to monitor, respond, and manage defensive controls. Adaptations that are different from those existing are impacted by the abilities of those assigned responsibility. A superior adaptation is of little use if the end users are unable to implement it properly.
- Performance—An adaptation must not severely degrade system performance. Whereas malware can be careless about performance issues, defensive measures with performance problems are often unacceptable even when they provide an important adaptation to a class of threats.
- Usability—Defensive adaptations that are effective, but too difficult to use will not find favor with those who need them most. Complicated adaptations will be abandoned or circumvented by humans who are attempting to accomplish a particular task.
- Management—A properly managed system implements change control. However, this can impede deployment of the adaptable defense. An adaptable control might be altogether avoided if it is perceived to be too difficult to manage.
- Operations—Effective defenses contribute to security operations. An adaptable defense that

exists in a silo inhibits the flow of security operations.

- Design—Ideal security controls are built-in rather than bolt-on. Unfortunately, most adaptable defenses are bolt-on. Integration efforts may be hampered by the complexity of the tool.
- Perceptions—Qualitative risk assessments might lead management to conclusions that a particular adaptable defense is unnecessary. Perceptions based on insufficient or inaccurate information inhibit the acquisition and deployment of adaptable defenses.

These constraints burden defensive adaptability. Constraints impact the ability of a defensive control to compete with the unbridled capabilities of malware encountered. The competition between adaptable threats and defenses are becoming increasingly unbalanced in the favor of malware. In this regard, attackers have a distinct advantage that is evident by the continued rise in compromises and data losses.

Strengthening Defensive Adaptations

Security is first and foremost a people problem. Weaknesses in systems are going to occur. All security problems have their root in people. Some programmers will make mistakes when coding that is further missed by reviews and quality assurance. System integrators will occasionally put things together incorrectly. System administrators will introduce configuration errors or fail to follow procedures. Users will also make honest

mistakes and fall victim to an attacker's trickery. Let us not forget that malware is an offspring of the warped efforts of bad people. All sorts of unsavory individuals such as criminals, spies, and terrorists are ultimately directing the actions of malware. Our goal as security professionals should be to not only employ adaptive defensive technology, but also to establish adaptive operations that are proactive. In this regard, the reactive nature of our current adaptable defenses can be augmented with techniques and processes that are prepared for the worst. Consider some of the following during the design, implementation, and management of security operations for information systems.

Anticipate Compromises

Develop an attitude that the best plans will eventually be circumvented. Manage stakeholder expectations by advocating proactive measures that can be used as early warning detection of failed countermeasures. Note areas within the system that are at higher risk for compromise and conduct more frequent reviews.

Response Plans

Contingency planning and incident response are invaluable tools that can be used to prepare for the eventual compromise in a system. Having a plan is great, but it is only as good as those who are sufficiently familiar with its guidance. The plans should be regularly practiced and updated when weaknesses are discovered. Ensure the plans address the actions required to clean

the system and restore normal operations.

Penetration Testing

Periodically attempt to break into your system. Hire reputable professionals to do the same. Use some of the same tools attackers use to compromise a system. Penetration testing should be used to exercise contingency and incident response plans.

Operational Alternatives

Few, if any, software products have proven impervious to vulnerabilities. Unfortunately, new vulnerabilities seem to be reported weekly for some products. Critical vulnerabilities with exploitable code in the wild may subject an organization to unacceptable risk. At such times it may be prudent to deploy or have ready other products that can be used instead of the one with a critical vulnerability. Require the use of alternative applications until all instances of the vulnerable product are appropriately patched. Consider altering access controls on the affected application to prevent intentional or accidental use. The downside to operational alternatives is increased management complexity and cost. The cost of a potential exposure and cleanup should be compared with the periodic licensing and management expenses.

Defense in Depth

Traditionally, defense in depth relies on the overlapping of policy, people, and technological countermeasures

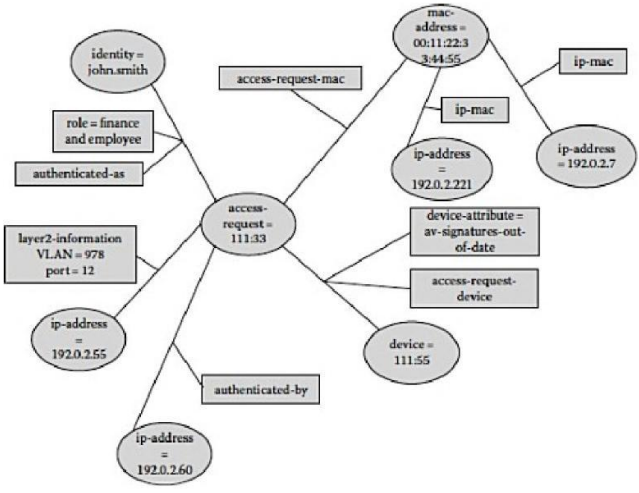


Figure 3.3 Complex diagrams may emerge using IFMAP.

Mike Fratto wrote

> IF-MAP provides a standardized framework for network and security devices to publish device state data —such as IP address, authentication, or virtually any meaningful information—to a central repository that can be used by other applications. This repository can be used for security, asset management, discovery, or any other purpose.[2]

Stuart Bailey, the CTO and Founder of Infoblox synthesized it best:

> MAP is like a MySpace or Facebook for enterprise infrastructure security pieces that each component publishes and subscribes to.[3]

Bailey said:

> This is a community of security infrastructure devices where each device can allow its circle to know what it sees on the network, and share information.[3]

Mike's description of the "potential promise" of IF-MAP especially coupled with Bailey's analogy should hit home for practitioners because each goes to the heart of what our profession is about in many instances, context. Everything has to be considered in its own context and IF-MAP fits that way of looking at things. It has the ability to transcend a simple database to be a true open standards-based repository of all types of data that in turn will create a potential economy of scale that should lower costs and raise skill levels across the board.

Evolution of "Threat State DatabasesSETI for Security Coming to a Computer Near You