Information Security Management Handbook

Sixth Edition

Edited by

Harold F. Tipton, CISSP · Micki Krause Nozaki, CISSP

Volume 5

CRC Press
Taylor & Francis Group
AN AUERBACH BOOK

Information Security Management Handbook

Sixth Edition

Volume 5

OTHER INFORMATION SECURITY BOOKS FROM AUERBACH

A Practical Guide to Security Assessments
Sudhanshu Kairab
ISBN 978-0-8493-1706-4

Adaptive Security Management Architecture
James S. Tiller
ISBN 978-0-8493-7052-6

Assessing and Managing Security Risk in IT Systems: A Structured Methodology
John McCumber
ISBN 978-0-8493-2232-7

Asset Protection through Security Awareness
Tyler Justin Speed
ISBN 978-1-4398-0982-2

Cyber Security Essentials
James Graham and Ryan Olson, Editors
ISBN 978-1-4398-5123-4

Data Mining and Machine Learning in Cybersecurity
Sumeet Dua and Xian Du
ISBN 978-1-4398-3942-3

Defense against the Black Arts: How Hackers Do What They Do and How to Protect against It
Jesse Varsalone and Matthew McFadden
ISBN 978-1-4398-2119-0
Publication Date: September 09, 2011

FISMA Principles and Best Practices: Beyond Compliance
Patrick D. Howard
ISBN 978-1-4200-7829-9

Information Security Risk Analysis, Third Edition
Thomas R. Peltier
ISBN 978-1-4398-3956-0

Information Technology Control and Audit, Third Edition
Frederick Gallegos and Sandra Senft
ISBN 978-1-4200-6550-3

Introduction to Security and Network Forensics
William J. Buchanan
ISBN 978-0-8493-3568-6

Machine Learning Forensics for Law Enforcement, Security, and Intelligence
Jesus Mena
ISBN 978-1-4398-6069-4

Managing an Information Security and Privacy Awareness and Training Program, Second Edition
Rebecca Herold
ISBN 978-1-4398-1545-8

Mobile Device Security: A Comprehensive Guide to Securing Your Information in a Moving World
Stephen Fried
ISBN 978-1-4398-2016-2

Practical Risk Management for the CIO
Mark Scherling
ISBN 978-1-4398-5653-6

Secure and Resilient Software: Requirements, Test Cases, and Testing Methods
Mark S. Merkow
ISBN 978-1-4398-6621-4

Secure Java: For Web Application Development
Abhay Bhargav and B. V. Kumar
ISBN 978-1-4398-2351-4

Secure Semantic Service-Oriented Systems
Bhavani Thuraisingham
ISBN 978-1-4200-7331-7

The Security Risk Assessment Handbook: A Complete Guide for Performing Security Risk Assessments, Second Edition
Douglas Landoll
ISBN 978-1-4398-2148-0

Security of Mobile Communications
Noureddine Boudriga
ISBN 978-0-8493-7941-3

Security Patch Management
Felicia Nicastro
ISBN 978-1-4398-2499-3

Security Strategy: From Requirements to Reality
Bill Stackpole and Eric Oksendahl
ISBN 978-1-4398-2733-8

AUERBACH PUBLICATIONS
www.auerbach-publications.com
To Order Call: 1-800-272-7737 • Fax: 1-800-374-3401
E-mail: orders@crcpress.com

Information Security Management Handbook

Sixth Edition

Volume 5

Edited by

Harold F. Tipton, CISSP · Micki Krause Nozaki, 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

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

Version Date: 20110707

International Standard Book Number-13: 978-1-4398-5346-7 (eBook - PDF)

Contents

DOMAIN 4: APPLICATION DEVELOPMENT SECURITY
System Development Controls

DOMAIN 5: CRYPTOGRAPHY
Cryptographic Concepts, Methodologies, and Practices

DOMAIN 6: SECURITY ARCHITECTURE AND DESIGN
Principles of Computer and Network Organizations, Architectures, and Designs

DOMAIN 7: OPERATIONS SECURITY
Operations Controls

Introduction

From the earliest editions of the *Handbook*, we recognized a growing need for professionals who are qualified to meet the challenges of complex technologies and escalating threats to information security.

However, as risks mount and information technology becomes that much more complicated, certified information security professionals must increasingly partner with skilled staff from sister disciplines such as risk management, business continuity, and law.

Today, maintaining information technology security, as well as keeping pace with competing standards, onerous regulations, and competitive markets, requires a village—a well-trained, well-educated, and well-informed team.

And so we offer our current edition of the *Handbook of Information Security Management*, with its virtual toolset of essays and dissertations addressing the whole of risk management, including people, processes, and technologies.

The information provided is practical, useful, and hands-on. The pieces are written by dedicated and committed persons 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.

It takes teamwork …

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

As always, we wish you the best.

Hal Tipton
Micki Krause Nozaki

Editors

Harold F. Tipton is an independent consultant and past president of the International Information System Security Certification Consortium, and has been 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 president of the national ISSA organization (1987–1989). He was added to the ISSA Hall of Fame and the ISSA Honor Roll in 2000 and elected an ISSA Distinguished Fellow in 2009. 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 bachelor of science in engineering from the United States Naval Academy and his master of arts 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, an Information Systems Security Architecture Professional, and an Information Systems Security Management Professional.

He has published several papers on information security issues for Auerbach Publishers in the *Handbook of Information Security Management and Data Security Management,* and other publishers, in the *Information Security Journal,* the National Academy of Sciences' *Computers at Risk,* DataPro Reports, various Elsevier publications, and the *ISSA Journal*.

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 International Information Systems Security Certification Consortium [(ISC)$^{2®}$]; Frost & Sullivan; University of California, Irvine; California State University, Long Beach; 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 Harold F. Tipton Lifetime Achievement Award in 2001.

Micki Krause Nozaki, MBA, CISSP, has held positions in the information security profession for the past 20 years. Nozaki 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 security. She received the Harold F. Tipton Lifetime Achievement 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 ISSA and the $(ISC)^2$, 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.

Contributors

Sandy Bacik, CISSP-ISSMP, CISM, CGEIT, CHS-III
Bacik Consulting Service

Samuel Chun, CISSP
Hewlett-Packard Company

Juan M. Estevez-Tapiador, PhD
Carlos III University of Madrid

Todd Fitzgerald, CISSP, CISA, CISM, CGEIT, PMP, HITRUST, ISO27000, ITILV3
National Government Services

Chris Hare, CISSP, CISA, CISM
Verizon

Foster Henderson, CISSP, CISM, SRP, NSA IEM
United States Government

Paul Henry, CISSP
Forensics & Recovery LLC

Julio Cesar Hernandez-Castro, PhD
Carlos III University of Madrid

Rebecca Herold, CIPP, CISSP, CISM, CISM, FLMI
The Privacy Professor®

Carl Jackson, CISSP
Pacific Life Insurance Company

Georges J. Jahchan, CISA, CISM and BS7799-2 Lead Auditor
Quattro Associates

Leo Kahng, CISSP
Cisco Systems, Inc.

Salahuddin Kamran, CISSP, CISA, CFE
Alvarez & Marsal

Seth Kinnett, CISSP
The Goldman Sachs Group, Inc.

James C. Murphy, CISSP-ISSMP, GSEC, CISA, CISM
North Carolina Office of Medicaid Management Information System Services

David O'Berry, CSSLP, CRMP, CISSP-ISSAP, ISSMP, MCNE
The South Carolina Department of Probation, Parole, and Pardon Services (SCDPPPS)

Pedro Peris-Lopez, PhD
Carlos III University of Madrid

Robert Pittman, CISSP
County of Los Angeles

Sean M. Price, CISA, CISSP
Independent Security Researcher and Consultant

Ralph Spencer Poore, CISSP, CFE, CISA, CFE, CHS-III, CTGA, QSA
Cryptographic Assurance Services, LLC

Pradnyesh Rane
Persistent Systems Limited

Edward Ray, CISSP
NetSec Consulting

Arturo Ribagorda, PhD
Carlos III University of Madrid

E. Eugene Schultz, PhD, CISSP
Emagined Security Consulting

Rob Shein, CISSP
Hewlett-Packard Company

Anne Shultz
Illinois Institute of Technology

Robert M. Slade, CISSP
Consultant

Jeff Stapleton, CISSP
Bank of America

DOMAIN 1

ACCESS CONTROL

Access Control Techniques

Chapter 1

Whitelisting for Endpoint Defense

Rob Shein

"Whitelisting" refers to an approach for control whereby a list of "known good" activities is maintained. Any actions that correspond to that list are permitted, while all others are blocked or disallowed. A classic example of this is proper firewall configuration, whereby only predefined "acceptable" traffic is allowed to pass, and all other traffic is dropped by a default rule. There is little debate that, given the numerous and ever-expanding ways in which attackers learn to overcome defensive measures, a whitelisting approach is far superior to blacklisting. The challenge, however, is in producing an all-encompassing list of precisely what behavior constitutes "acceptable," and doing so in advance with enough reliability that the security function in question will not break or hinder continuing operations. For networking traffic, this is relatively simple, since there are a finite number of protocols (either as defined by Internet Engineering Task Force Requests for Comments (IETF RFCs) or by port numbers), a finite number of endpoints that receive inbound connections, and simple methods by which the activity of existing systems may be observed and categorized in advance. In other realms, whitelisting has proven more challenging and thus only now has become feasible. This chapter will examine application whitelisting including differing approaches, the overall maturity of the industry, benefits and disadvantages over traditional antimalware approaches, and considerations when considering adoption of the technology.

Ashes to Ashes, Bytes to Bytes: The Malware Life Cycle

A useful construct in discussion of antimalware solutions is the "malware life cycle," which starts with the creation of a variant of malware and ends with the termination of the last remaining instance of that malware in the wild. For some forms of malware (such as custom-developed instances, intended for targeted use against a specific organization) the life cycle may be quite short. For others, such as worms with broad infection footprints and highly effective counterdetection mechanisms (like Conficker), the life cycle may be extremely long.

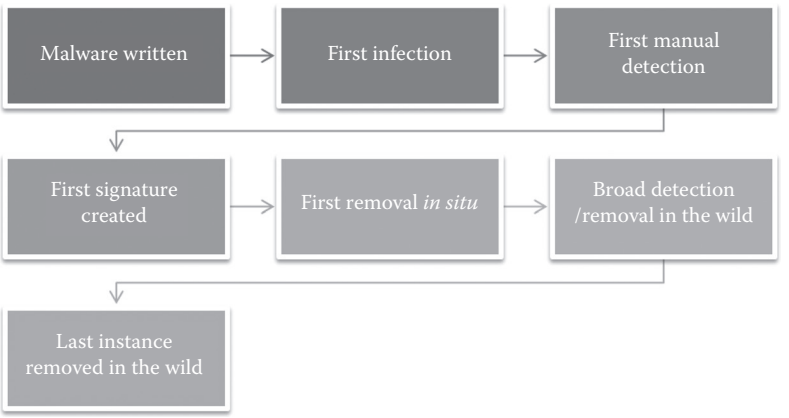

Figure 1.1 Malware life cycle.

The first three steps of the life cycle relate to the period of time when the malware is developed, released, and first detected. This period preludes the existence of any signatures by which organizations may protect themselves using blacklist-based solutions. The second period begins with the establishment of automated means for detection and (if possible) removal of the malware variant. During this period, the overall threat of the malware declines steadily. Finally, in the last step of the life cycle, the malware variant reaches extinction. It should be noted that this final step may be delayed for years, simply due to the presence of infected machines that are neither cleaned (due to limited use of the systems or a user with limited expertise who has come to accept the impact of virus infection) nor retired until long after obsolescence (Figure 1.1).

During the first three steps, blacklisting offers no protection whatsoever. In the absence of detection by manual means, there can be no signature to detect the malware; in the absence of a signature, there can be no automated, broad-ranging detection that would prevent further infection by the malware. Symptoms may manifest themselves boldly, but until detection of the malware itself (and the isolation of the offending file or files that must come with it), few options exist to address the threat effectively on any manner of scale.

During the next three steps, risk to the general populace (resulting from the malware variant) decreases fairly rapidly. Once a first signature is created by one vendor, others are certain to follow shortly thereafter. In fact, there is a fair amount of what could arguably be called plagiarism in the antivirus industry, as demonstrated by Kaspersky Labs in early 2010. Kaspersky put out signatures for nonmalicious files. When other antivirus vendors quickly followed suit, Kaspersky revealed that the binaries in question were entirely innocuous and were in fact developed by Kaspersky as nothing more than innocent applications intended as the basis for their own deliberately fraudulent signatures, to illustrate the degree to which antivirus companies copy each others' work with no validation of the work being copied. And thus, once a single antivirus vendor has issued a signature and malware classification, others are close to follow.

How Traditional Antivirus Fails

When talking about a new approach to a problem, it is useful to first discuss old approaches and why a new method is worth considering. In the end, all innovation must be weighed against the

status quo. Traditional antimalware approaches (most commonly referred to as "antivirus" solutions, although they typically protect against more than just viruses) use a blacklisting approach, in the form of signature-based detection of malicious executable files. This approach is often bolstered by a behavioral system of some form, which consists of a blacklist of suspect behaviors commonly used by malware. The challenge posed by the first approach is that only files known to be malicious are detectable. As a result, it follows that at some point in the life cycle of detection, any new form of malware is undetectable using a blacklisting approach. It is the viral equivalent of a zero-day exploit. At the level observed within a single endpoint, this period of time extends not only to the point where a signature has been developed for the specific instance of malware, but also to the point where a signature is available for the brand of antivirus software it uses and the point in time when that signature has been downloaded. This is an important point, as it explains both the race for antivirus software vendors to keep current with their competitors and the necessity for frequent updates to their software.

There is an additional challenge to antivirus, where the life cycle is circumvented. Some malware employs tactics like polymorphism or self-encryption/decryption to evade detection in combination with other tactics (like kernel hooking) to interfere with the ability of antivirus solutions to block their activity. In the worst cases, each new infection is like its own standalone release of a virus, undetectable by any signature that is based upon a different instance of that virus. This kind of malware is the digital equivalent of human immunodeficiency virus (HIV); the core payload is the same, but the surrounding shell of data is sufficiently unpredictable to make immunization infeasible using current technology. Approaches exist to detect such malware based upon behavior, but this results in a specialized approach for each worm or virus, and is not a scalable or feasible approach.

The manifestation of this problem with blacklisting is well documented. Looking at the statistics provided by VirusTotal (a free, open-to-the-public service to analyze suspicious files using existing antivirus engines), two interesting facts appear side by side. The first statistic relates to how many files submitted within the last 24 hours had malicious content that went undetected by one or more of the antivirus engines (39 of them at the time this was written) that are used. At the moment this sentence was written, that statistic showed that only 2,764 out of 73,016 files were detected as malicious by every antivirus engine. The second statistic is the number of updates that have occurred within the last 24 hours. At the time of writing, that period of time showed an average of 9.2 updates per hour combined across the entire spectrum of products. That amounts to an average of 5.7 updates each day for each of the software products. At this rate, it is impossible to perform any meaningful kind of change control or quality testing at a customer site before deploying a signature update; before the testing for one update was complete, the next one would have arrived, restarting the cycle. This would not be a problem if the signatures could be relied upon to be accurate or safe.

Unfortunately, signatures are regularly released that incorrectly flag innocuous (and often critical) software components as being viruses. In some instances, system binaries of Microsoft Windows have been affected, causing machines to crash or even become unbootable. (In a turn of irony, the submission of this chapter to the publisher was actually delayed by the release of a faulty signature in early 2010, which rendered the author's laptop unbootable until it could be fixed. Unfortunately, whitelisting is not nearly as widely deployed in enterprise environments as traditional antivirus and no home-user version of whitelisting exists as of yet.) *Virus Bulletin* did a test of 60 different antivirus products in early 2010; of these, 20 were said to have failed the test due to false positives, in many cases related to software from Google, Microsoft, Adobe, and Sun Microsystems. While some that failed the test were from smaller and less mature vendors,

solutions by Microsoft, Norman, and Fortinet also failed. The operating system upon which the solutions were tested was Windows XP, which is hardly a new platform.

So, with these problems, it seems natural to hold the entire industry at fault for such failings. The approach taken, after all, is akin to setting up security guards around a military base that are supposed to let everyone in except the bad guys. Such an approach is foolhardy at best, and destined to fail, yes? Perhaps, but it's important to take note of the fact that until recently, technology hadn't reached the point where another method was feasible. This point will arise later on, but in short the antivirus industry has evolved around a blacklisting approach because for most of the time since the PC became commonplace, blacklisting was the only workable approach to malware detection.

Another Way: How (and Why) Whitelisting Works

Whitelisting takes an opposite approach to malware prevention. Instead of trying to block only that which is known to be bad, it allows only that which is known to be good. There are several benefits to this approach. But to understand the benefits, first it is important to understand the underlying mechanisms of whitelisting itself.

There are effectively two forms of whitelisting: file-based and system call-based. The first focuses on the actual executables that are called into memory (either from disk or network I/O), while the second focuses on specific system calls at a very low level that are executed by software. The second approach is now frequently referred to as a form of Host-based Intrusion Prevention System (HIPS), but in reality is another form of whitelisting. In both cases, there is a learning period, during which the whitelisting application undergoes "supervised learning," and is essentially told what the definition of "good" is. For most file-based whitelisting applications, this is a matter of having the software recursively search the file system to find every executable present and add them to the policy for that machine. This policy governs the list of applications that are allowed to execute and what they are permitted to do. For call-based whitelisting, the process is more involved. A test system is set up and run with the whitelisting application in a mode whereby it will alert on policy violations, but not block the activity behind the alerts. Much like the tuning of a firewall, the alerts become the basis for policy rules, and after a time all normal executable activity is accounted for, after which point the policy can be deployed safely to endpoints.

For file-based whitelisting, the typical identification of executables revolves around a tuple of hash, filename, file size, and file location. In combination, these characteristics provide for robust identification of specific files and serve to detect tampering as well. For call-based whitelisting, these components come into play, along with specifics related to low-level activity by the file in question (such as what directories it reads from/writes to, what network activity it engages in, and so on). One benefit of this method over traditional antivirus is that all the software needs to do when an executable runs is to hash it. This is less demanding in both memory and processing power than the task of checking an executable's entire length (the primary executable for Microsoft Outlook 2007, for example, is over 12 megabytes in size) against a long list of signatures (which range into the millions). The relatively static nature of both approaches lends itself to proper change control, while preventing the kinds of issues that faulty antivirus signatures tend to cause. Even more importantly, this approach requires no system-wide scans past the initial learning phase, as opposed to the necessary but nonetheless extremely annoying scans that are typically needed on a weekly basis by other antivirus products, just to ensure that no malicious content has slipped onto a system.

There are other benefits as well, based around business processes. For example, when a corporate laptop is out of touch (such as when a consultant is traveling) from the corporate network, it is unable to receive updates to its antivirus signatures or the policy that controls antivirus software. However, it is at greater-than-normal risk from viruses and other malware, as it connects to Wi-Fi networks in hotels and coffee shops without the same protections that defend it while on a corporate network. On the other hand, whitelisting products require no interaction with the central management point to maintain a steady state of protection. For most products, approved changes can even be made to the endpoint's policy without the ability to communicate back to the central management point. So in this way whitelisting also trumps antivirus for protection based upon its "accept known good, reject all else" methodology, and the simplicity that comes with it.

In terms of the malware life cycle, everything changes. Instead of being released into an environment that is hospitable until otherwise decided, malware is unable to gain any foothold whatsoever. Even tactics such as polymorphism are turned on their head; if a piece of malware should somehow accidentally be added to the whitelist, and thus allowed to run, the very nature of its polymorphic routine would make it unable to infect other machines. The evasive tactics that help it evade detection by traditional antivirus would render additional instances of the malware unmatchable by the permissive policy setting. In the context of an environment with whitelisting on every endpoint, the path goes directly from malware creation to simultaneous detection and extinction, with no interim steps. Additionally, whatever executable content exists is still preserved, but in a form of stasis, unable to run even to clean up after itself. This, along with the detailed information provided by the whitelisting application, makes for powerful forensics tools (Figure 1.2).

Once described in light of other methods, this approach seems like an obvious choice, particularly given the way such paradigms work in the real world. For example, we control access to our homes with keys, and only give copies of those keys to people we know and trust. We don't attempt to build doors that will let everyone in except burglars, or distribute keys widely but forbid select untrustworthy individuals from owning copies. And the same philosophy works well in the world of computer security as well. Programmers are taught to sanitize inputs not by blocking characters that are known to be problematic, but instead by only allowing those which would be used for valid purposes. So why hasn't this approach been adopted long ago?

The answer lies in the challenge of defining "good," managing that definition, and enforcing it. For file-based whitelisting, the challenge is a bit easier, since all that is needed is the identification of the executable itself, but for call-based whitelisting it can be more challenging. Every possible valid behavior of every piece of valid software on the system must be accounted for, in advance, or significant problems will result. In addition, issues can be introduced as a result of the added interaction with regard to network or disk I/O. One product, when tested on a Domain Name System (DNS) server system intended for heavy use, introduced failures this way; the network shim acted as a stateful firewall, but did not have sufficient buffer to maintain state for the 3000+ connections per second (even though User Datagram Protocol (UDP) is stateless, any good firewall will maintain a sense of which inbound responses to expect) and, therefore, caused failures in

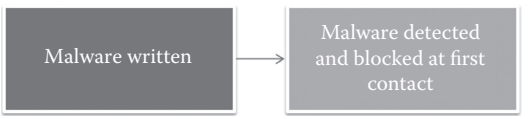

Figure 1.2 Malware life cycle under whitelisting.

the server. The vendor, with assistance from the project that performed the testing, modified and improved their whitelisting product to address this problem, but it demonstrated the peculiar ways in which such low-level interaction can negatively impact applications.

Considerations When Implementing Whitelisting

Whitelisting products work quite differently from traditional antivirus solutions, and as such they have radically different challenges with regard to selection and implementation. While it is important with all technologies to prepare adequately before rollout, this is especially true when it comes to whitelisting. The first set of challenges relates to the process by which system policies are created for the endpoints. If a policy is created on a single system for rollout to other systems, then two things need to exist. One, there needs to be a reasonable assurance of the homogeneity of the environment to which that policy is being deployed, as each variation will result in adverse or unexpected behavior. Two, there needs to be a process for addressing such variances. It's also important to recognize that there will almost inevitably be systems that are infected with some form of malware and that these will cause variances as well.

Whitelisting is obviously best suited for environments with mature controls over endpoint configuration and use. When end users have been given control over what is installed on their own systems, their configurations drift from the original deployed definition. Some of these deviations are relatively harmless (like addition of valid search bars to web browsers) while others are more dangerous (free screensavers, anyone?). Either way, they will all add to the complexity and difficulty of a whitelisting solution, both during deployment and immediately after. In such environments, there needs to be a fundamental shift in user behavior. A standard user login must not be permitted to make changes to endpoint policy configuration. If it can, then any malware that executes will execute in that user's context, and thus will be allowed to alter the endpoint. This nullifies the point of whitelisting entirely, and renders it absolutely useless. Even for power users, who can be (or, for business reasons, need to be) trusted with the ability to install new applications, there needs to be an alternate login that they use, so that they know precisely when they are (and more importantly, are not) adding new approved executables to their desktop configuration. It is worth noting that this approach works well with most file-based whitelisting, but does not work with call-based systems.

Challenges When Deploying Whitelisting

The deployment of whitelisting can provide extraordinary insight into the current state of an environment, as variances from the expected norm will almost always be found in surprising places. So in the exception handling process during rollout, a distinction must be made between innocuous (but unaccounted for) executables and malicious ones, or else those performing the rollout will simply lump any viruses or trojans in with other applications that will be allowed to exist on the network. This approach works best in environments where there is a standard desktop build, and especially well where end users do not have local administrator privileges, for obvious reasons. An accounting should be made for add-ons to existing valid applications (like browsers, which will have both good and bad search bars and helpers, plus add-ons like Flash, which will exist in multiple versions in any environment). A decision should be made whether temporarily accepting old versions or forcing updates will be the normal operating procedure during rollout as well.

If, on the other hand, policies will be autogenerated for each system, then there needs to be assurance in advance that every system is clean from malicious content. It is tempting to plan for manual inspection of the policies after the fact, to search for anything that seems unusual; this method is doomed to failure. A policy for file-based whitelisting will have thousands of entries at the very minimum; every .exe, every .dll, every driver will be listed there. Examining a policy like this quickly makes one aware of just how little they know of the underlying software that drives any desktop or server, and there is no human way possible to spot a malicious file with any degree of assurance whatsoever. When looking at a policy for call-based whitelisting, it's even worse, as you're not looking at a list of files but a list of specific granular behaviors. In the future, analytic tools will probably exist to determine delta between multiple systems to find what is and is not the common norm for a specific form of desktop, but in the meantime systems must be proven clean prior to autogeneration of whitelisting policies. Obviously, given the shortcomings of other antimalware approaches, this is a challenge for endpoints that have been in use for any length of time (particularly desktops). So, there's an added benefit to deploying whitelisting during a rollout of new systems, a switch to virtual desktops/thin clients, or establishment/implementation of a standard core desktop deployment. In each of these cases, endpoints are freshly deployed from a "known good" image or definition, and can be assumed to be free of harmful content. In this case, however, even more care must be taken to ensure that the deployment is properly planned, or else the whitelisting may cause complications during rollout and initial testing (or be blamed incorrectly for problems that stem from other causes). Still, this is a case of trading one kind of assurance (making sure endpoints are clean prior to policy generation) for another (planning a clean and well-supported whitelisting rollout), and the latter form of assurance is one that should be performed to a fair degree anyway.

Postdeployment Challenges to Whitelisting

Once a whitelisting solution has been fully deployed, activities switch to maintaining the solution, which is comprised of four things. One, the policies on the endpoints will need to change over time, as new applications are deployed and existing ones are updated/patched. Two, the processes and procedures around the whitelisting solution will need to be refined as exceptions to user behavior and rights as well as special requirements for specific business purposes come to light. (There will be more on that later.) Three, updates to the central management points of the whitelisting solution must be performed, and in such a way as to maintain continuity of the solution. And four, updates to the end-point agents must be performed as well. Whitelisting is a relatively new technology and as such the vendors tend to make frequent and valid improvements, resulting in many software updates over the course of a year. Even more importantly, these updates provide significant improvement, driven by the observations and experiences of the vendors' customer base.

Policies on the endpoints will require updates for a number of reasons, but the most frequent changes will result from patches to applications and operating systems. At the bare minimum, "Patch Tuesday" will result in a number of alterations as .dlls and .exes are replaced on Windows-based devices. For file-based whitelisting, there needs to be a means to identify the source of the changes and identify it as a trusted source. For most applications, this can be done either by identifying the service (when automated patching is used) that is implementing the patch or by doing patching with credentials that are authorized to make changes to the whitelisting policy. The first approach is often called "trusted agent," while the second is known as "trusted user."

For call-based whitelisting, changes to application behavior must be enumerated and added to the end-point policy before the patch can be deployed, or such changes must be provided by the whitelisting vendor (where they've done the fingerprinting for their customer base) in advance. The first approach is time-consuming and must precede the normal testing cycle for patches (thus elongating the patch cycle), while the second approach leaves the environment vulnerable to errors by the vendor in their fingerprinting, or at the mercy of any subtle differences from the vendor's test systems. This is another of the reasons that file-based whitelisting has been more rapidly accepted than call-based whitelisting.

In most environments, culture must change with regard to user behavior when whitelisting is implemented. As stated earlier, users must not be allowed to operate with administrator rights and the ability to alter the whitelisting policies as a matter of normal operations. For most users, the ability to alter whitelisting policies should not be granted; for those users that are the exception to this, alterations to those policies should not be possible with their normal login. This is, in effect, another way of ensuring that end users cannot install or update their own applications, and represents a shift in how things are done for most environments. The effects of this will manifest in unexpected ways, and the best thing that can be done is to provide a means to quickly and easily respond to requests from users for new or updated applications. The most unexpected form of this need may be in the need to install or update applets or other web-based controls; most online collaboration tools, for example, rely upon such components, and the end user will tend to discover their inability to install them at the worst possible moments. The ability to quickly intercede, install the applet/ActiveX/plugin on short notice, determine the change to policy, and then push that policy out to all endpoints is usually the best way to address this, but different whitelisting solutions will provide different options to address short-turn requests of this sort. And on the other hand, the fact that all software installation needs to go through a review process, no matter how short, helps maintain control over desktop configurations and prevent not only the introduction of malware but also work-inappropriate or pirated software as well.

When Not to Employ Whitelisting

There are several types of users and user environments where whitelisting is not an effective or feasible approach. Home users are not well suited for whitelisting; in fact, there is no consumer-oriented whitelisting product on the market today. The dual challenges of determining a starting policy and maintaining it (including both patching and using proper diligence in vetting changes to that policy) are beyond the scope of nearly all home users. Additionally, the principle of using dual logins (one for normal use and another for making changes to whitelisting policy) is not only not likely to be followed by a typical user, it also is not in conformance with the way that home-oriented versions of Windows operate. Added to this are the challenges with troubleshooting the effects of incorrect policy settings, and the basic familiarity with how applications function that is needed in order to administer properly a whitelisting system. In comparison, normal antivirus products need only be installed and left alone, from the perspective of a casual user. (Until a flawed signature is released, that is.)

Other environments that are ill suited for whitelisting are ones where development work is being performed. Every new build of an executable will have a different signature and so the developer will be unable to perform much (or any) testing without either constantly running as a trusted user (which breaks the whole point of whitelisting) or constantly adding the new executables to

the existing policy of allowed applications. It is potentially possible to configure whitelisting so that it will recognize the compiler as a trusted agent, but this is sometimes easier said than done.

Evaluating Whitelisting Products

At the time of writing, the whitelisting product space was remarkably mature, despite being (in terms of years) relatively young. A recent comparison of file-based whitelisting products by a major publication gave high marks to every product reviewed, and while one product did stand out above the others, none were considered to be inferior. (For the most part, call-based whitelisting seems to have fallen to the side and is used for cases where extreme security and assurance are required, for reasons described earlier here.) It is worth noting that, with the advent of Windows 7, there is a whitelisting capability native to Windows: AppLocker.

Before choosing (or even evaluating) any technical solution, it is absolutely essential to identify requirements. This cannot be stressed enough and is particularly true with regard to whitelisting. Unless it is known what capabilities, characteristics, and features are needed or desired (and the difference between the two), it is impossible to properly meet those needs or desires. This is particularly true with whitelisting, where initial planning and preparation are even more important than with most other technologies. That said, there are a number of components/attributes to whitelisting products that can be compared when evaluating against those requirements.

The first aspect of whitelisting functionality that can be evaluated is manageability. The endpoint agents have the ability to utterly nullify systems or applications; without a means of properly administering the entire whitelisting solution, there is the potential to wreak havoc throughout an environment. Also, as policies evolve, there should be a way to track changes or break them out into components. One example would be a specific policy that prohibits the execution of certain applications (like iTunes, or other applications that are forbidden by a company's acceptable use policy), but which can stand apart from the individual policies that govern each individual endpoint. Other important things to consider are reporting and the ability to get reliable visibility into the overall state of end-point agents. In very large environments, it's crucial to have the ability to organize information (like the list of endpoints under control) into manageable groups. Imagine what it would be like sorting through a list of thousands or tens of thousands of systems, looking for a particular one, when doing troubleshooting. When determining requirements for the management capabilities, think about what information will be needed or desirable to gather, and what forms of control should be centralized.

Deployment is another factor. The best solutions will have the ability to deploy to endpoints remotely, from the management console. The question then becomes one of determining which agents have failed to deploy properly (since all whitelisting solutions hook the kernel, kernel-hooking malware that has already infected an endpoint can be a cause of this) and addressing the problem. Automatic discovery (either through network discovery or pulling information from Active Directory) is enormously helpful; make sure that there is a way to identify systems that should not have whitelisting agents installed, if there is reason to believe that such systems will exist. Otherwise, the same capability that makes for a simple and rapid rollout will also cause problems when machines that are not intended for whitelisting control (such as development systems) have it imposed upon them by an overzealous or inattentive administrator. This could be accomplished either by placing such systems into a special organizational unit within the management system or by applying a policy whereby whitelisting control is not enforced.

A related but separate thing to consider is the way the policies themselves are defined. Each system will, in effect, have its own policy. That said, there is a need for the ability to define policies that will govern entire groups of systems, much like the way Group Policy Objects operate in Active Directory. While better solutions will have the ability to define policies in this manner, this also results in the potential for confusion (also, as can happen in Active Directory with Group Policy Objects). The management system should allow not only for the definition of policies in multiple contexts, but also the ability to see precisely which policies affect which systems, and the ability to view the effects of inheritance. If one policy allows an application to run and another prohibits it, which one wins?

The last set of considerations relate to end-user experience. For the most part, whitelisting applications are transparent to the end user, with two exceptions: when an application is blocked and when the user is permitted to override or modify that prohibition. When looking at this functionality, it's important to consider the user base, and both the percentage of users who will interact with the whitelisting endpoint agents (for example, as trusted users) and how technically savvy they are. The amount of information displayed to them in a notification will have an effect on troubleshooting and should also be sufficient to make a sensible determination as to whether or not something that has been blocked is hostile, or simply an executable that needs to be permitted. In short, think about the different types of users that you will have (from the context of the whitelisting application) and be sure to get a feel for how the end-user experience will play out for each of those user types.

Summary

In closing, the important things to remember about application whitelisting are as follows:

- Whitelisting operates under a "Permit Good, Deny All Else" philosophy.
- There are fundamentally two approaches:
  - Whitelisting that looks at specific granular behaviors at the system level (call-based).
  - Whitelisting that looks at the specific executables that want to load into memory and operate (file-based).
- File-based whitelisting has come into prominence, mostly due to greater ease of implementation and management.
- Whereas traditional antivirus (which uses a blacklisting approach) incurs significant processor and I/O performance hits (especially during a full system scan), whitelisting is relatively lightweight.
- There is reduced risk to using whitelisting, owing to its better coverage with regard to new forms of malware and the ability to maintain it using proper change control methods. In comparison, traditional antivirus signatures are prone to false negatives and come out so frequently that change control is infeasible; this problem is all the more alarming given that on a fairly regular basis incorrect signatures are released by vendors with harmful effects.
- Any whitelisting deployment must be carefully planned and executed; whitelisting has the potential to cause significant disruption if it is improperly implemented.
- The vendor space for whitelisting is still evolving, but the products already show a high degree of functionality and usability.
- As yet, no feasible options exist for home users who wish to use application whitelisting.

- Whitelisting works best in environments where end users are not normally allowed to install applications and where there is a high degree of standardization among endpoints.
- Requirement gathering and development is a must when evaluating whitelisting solutions.

About the Author

Rob Shein, CISSP, is a cyber security architect for HP. During the past 30 years—starting with writing his first program at age 11—Rob has focused almost exclusively on security. His role at HP has spanned the utility, outsourcing, financial, government, and manufacturing sectors in the Americas and Asia. Rob's areas of specialization range from policy work to penetration testing, implementation/architecture, C2 systems, pure consulting related to security processes, and development of security solutions and standards around Smart Grid and SCADA systems. He has authored a novel and contributed to numerous publications, and is a frequent presenter at conferences.

Chapter 2

Whitelisting

Sandy Bacik

Access control consists of permitting or denying the use of a particular resource. Within networking environments, particularly at the network perimeter, enterprises have used blacklisting. Blacklisting consists of banning a list of resources from access. As the unauthorized and invalid access attempts increased, the blacklist continued to grow. This method allowed everything unless explicitly denied, i.e., default allow. Enterprises are now doing the reverse, only allowing authorized access, i.e., whitelisting, the "known good." Whitelisting turns blacklisting upside down, categorizing everything as bad except for a small group. Whitelisting is listing entities that are granted a set of privileges (access, services, validity, etc.) within an environment. A whitelist is solely used to define what is allowed to be executed, whereas anything that is not included on the whitelist cannot be executed.

Due to compliance, audit, and regulatory requirements, the enterprise resources and assets function should be documenting assets and resources. Resources can be groups, services, applications, computers, servers, routers, websites, etc. In small enterprise environments, a general purpose server is used for all manner of things (surfing the Web, reading e-mail, running enterprise applications, evaluating new software, etc.) and it is very difficult to keep whitelisting restrictions up to date for access. On the other hand, when a server has very few functions (like one used for just reading e-mail), using whitelisting can greatly improve security. Unfortunately, most enterprise systems fall somewhere near the middle between these two extremes. There are many types of whitelists an enterprise can utilize to assist in implementing whitelisting over blacklisting:

- *E-mail*: An e-mail whitelist is a list of contacts that the user deems are acceptable to receive e-mail from and should not be sent to the trash folder, similar to spam filters.
  - Internet Service Providers (ISPs): ISPs receive requests from legitimate companies to add them to the ISP whitelist of companies.
  - Noncommercial whitelists: Noncommercial whitelists are operated by various nonprofit organizations, ISPs, and other entities interested in blocking spam.
  - Commercial whitelists: Commercial whitelists are a system by which an internet service provider allows someone to bypass spam filters when sending e-mail messages to its subscribers in return for a prepaid fee, either an annual fee or a per-message fee.

- *Local Area Network (LAN) whitelists*: Many network admins set up Media Access Control (MAC) address whitelists, a MAC address filter, or subnets to control who is on their networks. This can be used when encryption is not a practical solution or in tandem with encryption. However, it's sometimes ineffective because a MAC address can be faked. Many firewalls can be configured to only allow data traffic from/to certain (ranges of) Internet Protocal (IP) addresses.
- *Program whitelists*: Enterprises should keep a list of valid software within the network. If an organization keeps a whitelist of software, only titles on the list will be accepted for use. The benefits of whitelisting in this instance are that the school administration can ensure itself that students will not be able to download and/or use programs that have not been deemed appropriate for use.
- *Application whitelists*: Enterprises should do regular application inventories for license agreements. One approach to combat viruses and malware is to whitelist software which is considered safe to run, blocking all others.

Let's compare using blacklists and whitelists for access control (see Table 2.1). There are more advantages and fewer disadvantages in using whitelists. Yet, there are two potential glaring issues with whitelisting. First, most organizations are apprehensive about going the whitelist route because the IT department does not want to increase the resources needed to manage the impact of keeping track of valid resources and impacting users. On the other hand, many organizations see explicitly denying things via a blacklist is not the most effective or productive way to manage and protect the environment. So it boils down to: What to do and why? The best approach depends on the solution to prevent execution of applications, services, and code. For example, if the implemented solution contains a very basic enforcement method that uses the "yes" or "no" to determine executability, then the enterprise might want to look elsewhere. Trying to use a

Table 2.1 Whitelist and Blacklist Advantages and Disadvantages

Blacklist Advantages	*Whitelist Advantages*
Easy to manage	More secure
Easy to install	More accurate
Can download updates quickly	Minimizes false positives
	Can be created at various levels within the enterprise
	Easy to customize
Blacklist Disadvantages	*Whitelist Disadvantages*
Exponential growth	More time to manage
Many false positives, potentially denying valid access	Requires additional time to install
Continual updates are required	
Hard to switch to whitelisting	

whitelist with this logic could turn into a management nightmare while also dramatically impacting end-user productivity. Another example would be a solution using the methodology by defining rules. This methodology is flexible enough to effectively balance enforcement, management, and productivity. This is definitely not the endgame in endpoint protection. It does have a built-in target solution and can be easily maintained.

In looking at today's enterprise, there are many requirements, standards, and policies that require access control to be implemented and reviewed on a regular basis for governance, audit and compliance. Implementing whitelisting will assist in making the audit and compliance reviews simpler to complete. Enterprises should have a list of valid applications, network equipment, customers, partners, sites/locations, employees, roles/groups, contractors, consultants, services, and ports. If an enterprise has these documented, they have a start on implementing a whitelisting solution for access control. See Table 2.2 for a sample listing of how whitelisting can be implemented for access control using some of the list above.

Using the information in Table 2.1 and the examples in Table 2.2, an enterprise can better manage access control of resources and limit the risk to those resources.

In conclusion, documenting all network resources and being able to use whitelisting will give the enterprise more control over those resources and lessen the risk to the enterprise. The upfront work for implementing whitelisting will require a larger effort. Once completed, the whitelisting will enable the enterprise to specifically know what resources are available and who has access to what resources. Overall, implementing whitelisting will reduce the risk of findings during a compliance audit.

Table 2.2 Using Whitelists

Asset List	*Whitelisting Use*
Applications	This allows an enterprise to track what application can and cannot be used within the network. Along with assisting in access control, this can reduce viruses and malware, and assist with license compliance
Network equipment	This allows an enterprise to be able to segment and route traffic based on network devices. It can allow or limit the access from partner, customer, and Internet sites to stop unauthorized access to finding unsecured resources
Groups	This allows an enterprise to have easier access control to applications and network resources by maintaining group memberships rather than have separate access control lists for each application or network resources
Ports	By knowing which ports an application or service uses, a perimeter firewall can be locked down to only permit the required ports to required network devices, again limiting the security risk to the network environment
Contractors or Consultants	By knowing who the contractors and consultants are within the enterprise, contract audits and access can be reviewed more quickly and removed, if necessary

About the Author

Sandy Bacik, CISSP-ISSMP, CISM, CGEIT, CHS-III, author and former CSO, has over 14 years of direct development, implementation, and management information security experience in the areas of audit management, disaster recovery and business continuity, incident investigation, physical security, privacy, regulatory compliance, standard operating policies and procedures, and data center operations and management. Ms. Bacik has managed, architected, and implemented comprehensive information assurance programs and managed internal, external, and contracted and outsourced information technology audits to ensure various regulatory compliance for state and local government entities and Fortune 200 companies. She has developed methodologies for risk assessments, information technology audits, vulnerability assessments, security policy and practice writing, incident response, and disaster recovery.

Access Control Administration

Chapter 3

RFID and Information Security

Salahuddin Kamran

Introduction

This chapter provides an overview of radio frequency identification (RFID) technology and some thoughts on privacy and security issues concerning RFID systems, and highlights some of the areas that have to be considered in designing and deploying RFID systems.

RFID is a technology that facilitates the automated identification of objects. While people are generally skillful at visual identification of a range of objects, computers are not. The task of identifying a coffee mug as a coffee mug is one that many bleary-eyed people perform naturally and effectively every morning in a variety of contexts. For computing systems, this same task can pose a challenging exercise in artificial intelligence. The simplest way to ease the process of automated identification is to equip objects with computer-readable tags. This is essentially what happens in a typical supermarket. Through a printed barcode on its packaging, a box of cereal identifies itself automatically to a checkout register. While a checkout clerk must manually position items to render them readable by a scanner, printed barcodes alleviate the overhead of human categorization and data entry. Over the course of decades, they have proven to be indispensable timesavers and productivity boosters.

The purpose of an RFID system is to enable data to be transmitted by a portable device, called a tag, which is read by an RFID reader and processed according to the needs of a particular application. The data transmitted by the tag may provide identification or location information, or specifics about the product tagged, such as price, color, date of purchase, etc. The use of RFID in tracking and access applications first appeared during the 1980s. RFID quickly gained attention because of its ability to track moving objects. As the technology is refined, more pervasive and invasive uses for RFID tags are in the works.

In a typical RFID system, individual objects are equipped with a small, inexpensive tag that contains a transponder with a digital memory chip that is given a unique electronic product code (EPC). The interrogator, an antenna packaged with a transceiver and decoder, emits a signal activating the RFID tag so it can read and write data to it. When an RFID tag passes through the electromagnetic zone, it detects the reader's activation signal. The reader decodes the data encoded in the tag's integrated circuit and the data is passed to the host computer for processing.

RFID tags promise in the near future to become the most numerous computational devices in the world. Their impending pervasiveness owes much to the power and flexibility that they achieve through starkly minimalist design. These tags come in a wide variety of shapes and sizes. Some tags are easy to spot, such as the hard plastic antitheft tags attached to merchandise in stores. Animal tracking tags that are implanted beneath the skin of family pets or endangered species are no bigger than a small section of pencil lead. Even smaller tags have been developed that can be embedded within the fibers of a national currency.

While barcodes have historically been the primary means of tracking products, RFID systems are rapidly becoming the preferred technology for keeping tabs on people, pets, products, and even vehicles. One reason for this is because the read/write capability of an active RFID system enables the use of interactive applications. Also, the tags can be read from a distance and through a variety of substances such as snow, fog, ice, or paint, where barcodes have proved useless. Developments in RFID technology are yielding systems with larger memory capacities, wider reading ranges, and faster processing. In response, the market for RFID tags is growing explosively and is projected to reach $10 billion annually within the decade.

History

RFID systems have gained popularity, and notoriety, in recent years. A driving force behind the rapid development of RFID technology has been the rise of pervasive commerce, sometimes dubbed the quiet revolution. Pervasive commerce uses technologies such as tracking devices and smart labels embedded with transmitting sensors and intelligent readers to convey information about key areas where consumers live and work to data processing systems. To gather this data, retailers can choose from a range of options.

RFID systems may be roughly grouped into four categories:

1. *Electronic Article Surveillance (EAS) systems*: Generally used in retail stores to sense the presence or absence of an item. Products are tagged and large antenna readers are placed at each exit of the store to detect unauthorized removal of the item.
2. *Portable Data Capture systems*: Characterized by the use of portable RFID readers, which enables this system to be used in variable settings.
3. *Networked systems*: Characterized by fixed position readers that are connected directly to a centralized information management system, while transponders are positioned on people or moveable items.
4. *Positioning systems*: Used for automated location identification of tagged items or vehicles.

These RFID systems enable businesses to have real-time access to inventory information, as well as a broader, clearer picture of consumers' buying habits. RFID technology also enables retailers and corporations to peek into the lives of consumers in ways that were, until recently, off limits. Products embedded with RFID tags can continuously transmit information ranging from an EPC identifier, to information about the item itself, such as consumption status or product freshness. Data processing systems read and compile this information, and can even link the product information with a specific consumer.

This composite information is vastly superior—and more invasive—than any data that could be obtained from scanning bar codes, or even loyalty cards. Frequent shopper cards link consumers to their purchases, but this limited information gives retailers only a narrow view of a

consumer's in-store purchasing trends. In contrast, RFID systems enable tagged objects to speak to electronic readers over the course of a product's lifetime—from production to disposal—providing retailers with an unblinking, voyeuristic view of consumer attitudes and purchase behavior.

Technology

RFID systems can be very complex, and implementations vary greatly across industries and sectors. For purposes of discussion in this document, an RFID system is composed of up to three subsystems:

- An *RF subsystem* performs identification and related transactions using wireless communication.
- An *enterprise subsystem* contains computers running specialized software that can store, process, and analyze data acquired from RF subsystem transactions to make the data useful to a supported business process.
- An *interenterprise subsystem* connects enterprise subsystems when information needs to be shared across organizational boundaries.

Every RFID system contains an RF subsystem and most RFID systems also contain an enterprise subsystem. An RFID system supporting a *supply chain* application is a common example of an RFID system with an interenterprise subsystem. In a supply chain application, a tagged product is tracked throughout its life cycle, from manufacture to final purchase, and sometimes even afterwards (e.g., to support targeted product recalls).

The characteristics of RFID enterprise and interenterprise subsystems are very similar to those of any networked IT system in terms of the types of computers that reside on them, the protocols they support, and the security issues they encounter.

RF Subsystem

To enable wireless identification, the *RF subsystem* consists of two components:

- RFID *tags* (sometimes referred to as *transponders*), which are small electronic devices that are affixed to objects or embedded in them. Each tag has a unique identifier and may also have other features such as memory to store additional data.
- RFID *readers*, which are devices that communicate with tags to identify the item connected to each tag and possibly associate the tagged item with related data.

Both the tag and the reader are two-way radios. Each has an antenna and is capable of modulating and demodulating radio signals. Figure 3.1 shows a simple RF subsystem configuration.

Tags

Most RFID tags contain at least two components: an integrated circuit for storing and processing information, modulating and demodulating a radio-frequency signal, and other specialized functions; and an antenna for receiving and transmitting the signal. Figure 3.2 shows samples of tags.

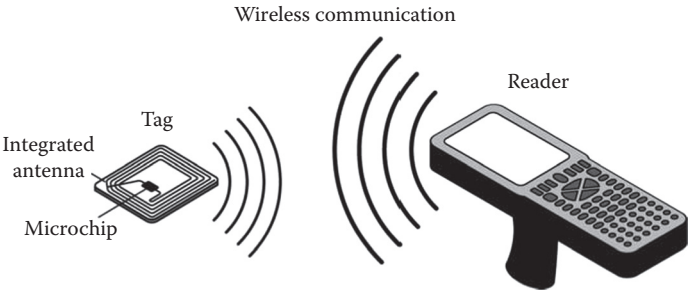

Figure 3.1 Simple RF subsystem.

The market for RFID tags includes numerous different types of tags, which differ greatly in their cost, size, performance, and security mechanisms. Even when tags are designed to comply with a particular standard, they are often further customized to meet the requirements of specific applications. Understanding the major tag characteristics can help those responsible for RFID systems identify the tag characteristics required in their environments and applications. Major characteristics of tags include identifier format, power source, operating frequencies, functionality, and form factor.

Tags are categorized into four types based on the power source for communication and other functionality:

1. *Passive*: A *passive tag* uses the electromagnetic energy it receives from a reader's transmission to reply to the reader. The reply signal from a passive tag, which is also known as the *back-scattered signal*, has only a fraction of the power of the reader's signal. This limited power significantly restricts the operating range of the tag. It also means that passive tags can only support data processing of limited complexity. On the other hand, passive tags typically are cheaper, smaller, and lighter than other types of tags, which are compelling advantages for many RFID applications.

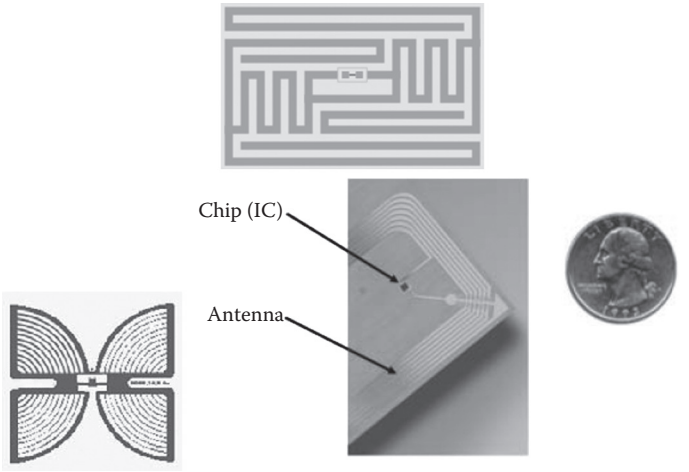

Figure 3.2 RFID tags.

2. *Active*: An *active tag* relies on an internal battery for power. The battery is used to communicate to the reader, to power on-board circuitry, and to perform other functions. Active tags can communicate over greater distances than other types of tags, but they have a finite battery life and are generally larger and more expensive. Since these tags have an internal power supply, they can respond to lower power signals than passive tags.
3. *Semiactive*: A *semiactive tag* is an active tag that remains dormant until it receives a signal from the reader to wake up. The tag can then use its battery to communicate with the reader. Like active tags, semiactive tags can communicate over a longer distance than passive tags. Their main advantage relative to active tags is that they have a longer battery life. The waking process, however, sometimes causes an unacceptable time delay when tags pass readers very quickly or when many tags need to be read within a very short period of time.
4. *Semipassive*: A *semipassive tag* is a passive tag that uses a battery to power on-board circuitry, but not to produce return signals. When the battery is used to power a sensor, they are often called *sensor tags*. They typically are smaller and cheaper than active tags, but have greater functionality than passive tags because more power is available for other purposes. Some literature uses the terms "semipassive" and "semiactive" interchangeably.

Readers

The tag and the reader must comply with the same standard in order to communicate. If a tag is based on a proprietary design, a reader must support the same communication protocol to communicate with that tag. In many cases, if proprietary tags are used, only proprietary RFID readers from the same vendor can be used.

A reader's interface with an enterprise subsystem may be wired or wireless. Most wired readers are in fixed locations and support applications in which the tags approach the reader. Some wired readers offer limited mobility using cables. Figure 3.3 shows a reader portal that reads tags on a pallet of boxes moving through the portal.

Figure 3.4 shows reader antennas mounted above each toll lane in a series of toll booths. As vehicles pass through one of the toll lanes, the reader reads a transponder that is attached to that vehicle's windshield.

In contrast, wireless readers support applications in which personnel must move around to read tags. Figure 3.5 shows an example of a mobile handheld reader.

Tag-reader communication is achieved by using a common communications protocol between the tag and the reader. Tag-reader communication protocols are often specified in RFID standards. Prominent international standards include the ISO/IEC 18000 series for item management and the ISO/IEC 14443 and ISO/IEC 15693 standards for contactless smart cards. The most recent EPCglobal Class-1 Generation-2 standard is essentially equivalent to the ISO/IEC 18000-6C standard.

Enterprise Subsystem

The *enterprise subsystem* connects readers to computers running software that can store, process, and analyze data acquired from RF subsystem transactions to make the data useful to a supported business process. For example, an RFID system in a retail clothing store has an RF subsystem that can read the identifier associated with each tagged garment. The enterprise subsystem matches the identifier to the garment's record in a database to determine its price and the number of other items of a similar type that remain in inventory. Some simple RFID systems consist of an RF

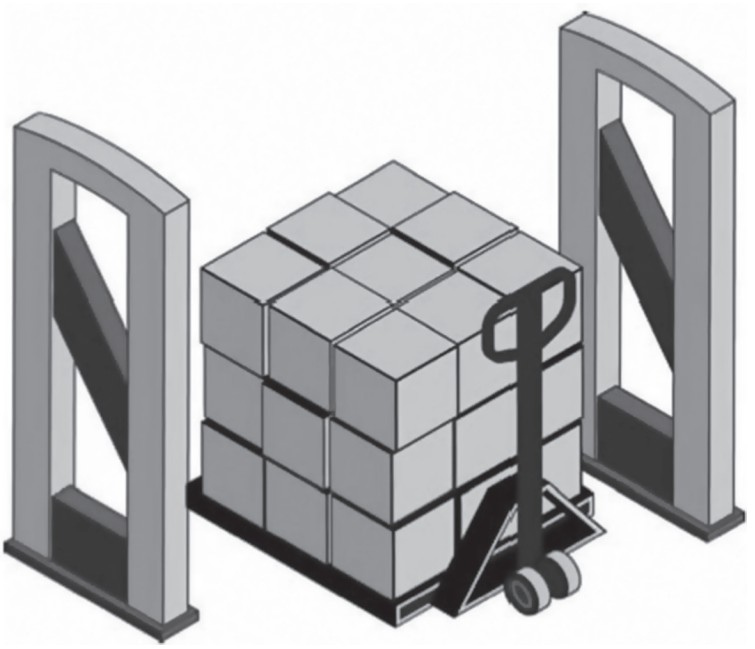

Figure 3.3 Reader portal.

subsystem only (e.g., RFID-based key systems in which a reader can make an access control decision without access to other computers). However, most RFID systems have both an RF subsystem and an enterprise subsystem.

The enterprise subsystem consists of three major components, which are shown in Figure 3.6.

Middleware: RFID *middleware* is responsible for preparing data collected from readers in the RF subsystem for the analytic systems that directly support business processes. Middleware hides the complexity and implementation details of the RF subsystem from the analytic systems.

Analytic systems are composed of databases, data processing applications, and Web servers that process the data outputs of middleware based on business requirements and user instructions. They contain customized business logic for each business process they support.

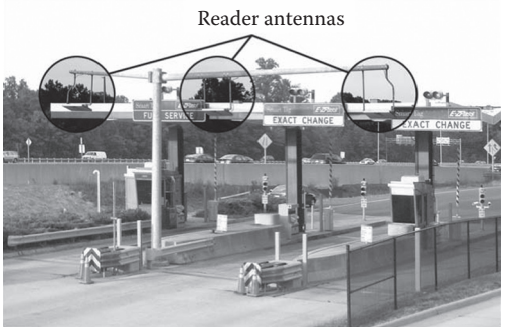

Figure 3.4 Reader antennas.

Figure 3.5 Mobile handheld reader.

Network infrastructure enables communication between the RF and enterprise subsystems, as well as among components of the enterprise subsystem.

Interenterprise Subsystem

The *interenterprise subsystem* connects enterprise subsystems together when information needs to be shared across geographic or organizational boundaries, such as in a supply chain application. Not all RFID systems contain interenterprise subsystems. The largest government interenterprise subsystem is currently the US Department of Defense's (DoD) Global Transportation Network. The DoD improves its logistics and operational efficiency by tracking DoD assets and personnel from their origin to their destination.

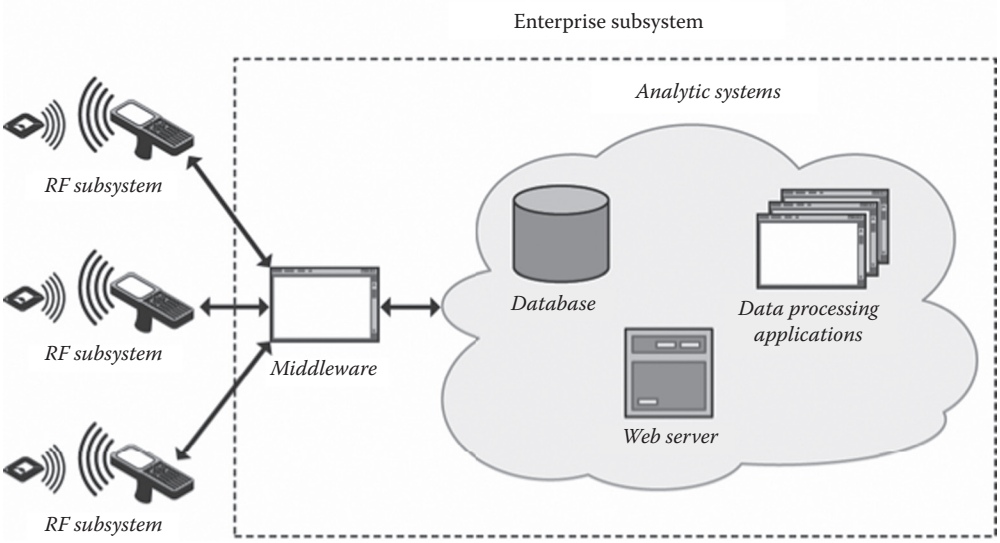

Figure 3.6 Enterprise RFID subsystem.

Open and Closed RFID Systems

RFID systems are either open or closed. Closed RFID systems are systems that do not have links with an outer environment. According to the intention of the designer, data that are collected within the system do not trespass the system's boundaries and remain entirely within the system. Data from outside the system will not trespass the system's boundaries either. Open systems are systems in which data that are collected within the system may be shared with other systems.

An example of a closed system is a logistics system that uses proprietary solutions for dealing with the data it collects from its tags. An example of an open system is a public transport ticketing system which is used in conjunction with an electronic shopping system, for instance by adding e-payment functionality to the transport ticketing card for shopping at shopping malls.

Keeping track of the collected data becomes more problematic in an open situation; relations may exist with third parties outside the system who use the information collected for other purposes.

Applications

RFID Application Types

There are many types of RFID applications, of which some of the most common are asset management, asset tracking, automated payment, and supply chain management. The key characteristic differentiating one RFID application from another is the purpose of identifying the tagged items. Table 3.1 lists reasons why an organization might want to identify an item and the general application type that best corresponds to those reasons.

Application types are not mutually exclusive; an implementation can combine elements of several application types. For example, both access control systems and sophisticated asset management systems include tracking features. Supply chain management is a tracking application that spans organizational boundaries and often includes process control and payment transactions.

Personnel responsible for designing and implementing RFID systems should understand what application types apply to their implementation so that they can select appropriate security controls. For example, the security controls needed to protect financial transactions in automated payment systems are different from those needed for tracking applications. The personnel should

Table 3.1 RFID Application Types

Identification Purpose	*Application Type*
Determine the presence of an item	Asset management
Determine the location of an item	Tracking
Determine the source of an item	Authenticity verification
Ensure affiliated items are not separated	Matching
Correlate information with the item for decision-making	Process control
Authenticate a person (holding a tagged item)	Access control
Conduct a financial transaction	Automated payment

also understand that an adversary may leverage RFID technology for an unintended purpose. For example, a warehouse may use RFID technology to determine what items it has in its current inventory, but an adversary may use the same system to track an item's whereabouts after it leaves the warehouse. In this case, an asset management system is later used to enable an unauthorized tracking application, perhaps used by an adversary to locate high-value targets.

Risks

RFID technology enables an organization to significantly change its business processes to:

- Increase its efficiency, resulting in lower costs
- Increase its effectiveness, improving mission performance and making the organization more resilient and better able to assign accountability
- Respond to customer requirements to use RFID technology to support supply chains and other applications

As described earlier, the RFID technology itself is complex, combining a number of different computing and communications technologies to achieve the desired objectives. Unfortunately, both change and complexity generate risk. For RFID implementations to be successful, organizations need to effectively manage that risk, which requires an understanding of its sources and its potential characteristics.

Privacy Aspects

In relation to RFID, privacy and security are two sides of the same coin and require an approach in which they are both tackled together and it might be possible to include security safeguards that may have positive implications on privacy. The double-sidedness of privacy and security requires attention, since it deals with embedding privacy regulations in the standards that are being developed on the various aspects of the RFID system (tags, readers, middleware, and the back-end information systems).

RFID is a means for identification. This identification can be of products, services, or persons. In most cases, RFID tags are related to products. When, however, a person is correlated to specific products by means of a token, an index, or another pointer, the identified information becomes personal information (or information that enables the identification of a person). Due to the 'enabling' characteristics of RFID tags the threat to privacy is a major concern, for the general public, companies, and governments alike.

In Figure 3.7, two direct privacy threats are identified: one in relation to the tag-reader system, and one in relation to the information that is collected and disseminated outside the tag-reader system.

The first kind of threat is the one that is most directly related to RFID. It focuses on the privacy implications of the tag-reader system itself. The second kind of threat relates to the use of data collected by means of an RFID system. The data that are disseminated by the tag-reader system may be collected in a database, for instance to monitor pallets in a supply chain management system. This kind of threat is not uniquely determined by the RFID system, but due to RFID the threats may be aggravated and have very specific dimensions.

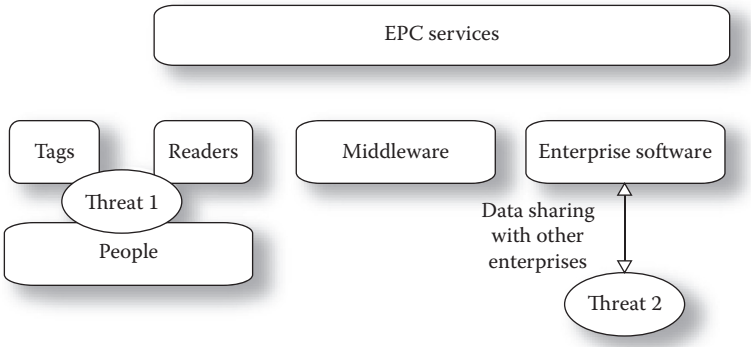

Figure 3.7 RFID privacy threats.

Privacy Threats within the Tag-Reader System

A tag may contain personal information. Passports, identity cards, and specific forms of public transport cards contain identifiable information. They may contain directly identifiable information on the card such as name and birth date. They also may contain data that functions as a key for a database in which personal information is stored. The privacy threat in the first case is obvious. In case a card holds indirectly identifiable information (a pointer that may refer to information about an identifiable person in a database), the privacy threat is indirectly present. Only when the intruder is able to link the pointer to the real person will privacy be invaded.

The unauthorized reading of tags is considered to be the most prominent privacy threat. Unauthorized reading is possible, especially in case of using UHF-based tags with reading ranges of approximately 20–30 ft.

Using tags to track persons is usually identified as being the second biggest threat to privacy. Tracking of persons via objects presupposes the linkage of identification data with individual track movements. Identification data can be acquired on the basis of electronic payments, loyalty cards, electronic ticket cards in public transport, etc. When a person carries an object that is linked to that person (such as a wristwatch) the data of the tag attached to the wristwatch may be used as identifiable information for the person carrying the wristwatch. It is possible to track the movements of this person by surveying the movement of the object for which the tag data are known. The information can for instance be used to retrieve personal preferences.

Privacy Threats at the Backend of the RFID System

The threats mentioned above correlate directly with the RFID reader to tag communication and the direct use of these data. Most privacy threats, however, refer to the collection and subsequent use of information outside the tag-reader system. Tags with unique IDs can easily be associated with a person's identity and smart cards with their own processing capacities may contain sensitive personal information.

There are three privacy threats related to the use of data outside the tag-reader system:

1. Using data for aggregating personal information
2. Using data for purposes other than originally specified
3. Using data to monitor specific behaviors

By means of data-mining techniques it is possible to find correlations between hitherto separated objects (and subjects). Deducting social networks may be especially interesting for intelligence agencies, for instance in trying to discover social networks of criminals: if one criminal can be traced, it is possible by data mining and pattern recognition techniques to sort out who else has a similar pattern of movement. This privacy threat is closely related to the following one.

RFID data may be collected for use in specific settings, but subsequently used in other settings. This is an example of "function creep": though originally not perceived, data collected for a specific purpose turns out to be useful for other purposes as well.

Monitoring can be done in real time, but it also can be done on the basis of aggregated data, that are subsequently analyzed in order to deduct specific patterns of behavior. An example of using RFID technology for individual monitoring is the business that uses an identifiable token (such as a loyalty card) to collect information on shopping behavior and uses this information to base decisions related, for example, on pricing without the consent of the customer.

Solutions that are more directly related to RFID are the ones that try to keep control over the data flow to the user (by means of killer and blocker tags, for example) in order to prevent information being disseminated against the user's wishes, and offer the users an "opt-in" choice. These solutions are based on the technical functioning of the RFID system, especially in the communication of RFID tag and reader. Other proposed solutions in this vein are using a Faraday cage to shield the tag from being read and reducing or removing the antenna (in the first case as a means to reduce the read range, while in the latter as a means to disable the tag). "Privacy by design" means that compliance with the privacy principles is sought by means of appropriate technical measures.

Another problem is the fact that low-cost RFID devices do not have the computational resources to use selected cryptographic methods. The kill tag, though appealing through its radical approach, may kill beneficial uses of the information that is hidden on the tag as well.

Security Aspects

This section contains a general overview of security threats of an RFID system, consisting of an RFID tag and a reader. The security threats are classified as either threats for the tag, or the wireless interface between the tag and the reader, or the reader.

Security Threats for the Tag

Falsification of Contents

Data can be falsified by unauthorized write access to the tag. This type of attack is suitable for targeted deception only if, when the attack is carried out, the ID (serial number) and any other security information that might exist (e.g., keys) remain unchanged. This way the reader continues to recognize the identity of the tag correctly. This kind of attack is possible only in the case of RFID systems that, in addition to ID and security information, store other information on the tag.

Falsification of Tag ID

The attacker obtains the ID and any security information of a tag and uses these to deceive a reader into accepting the identity of this particular tag. This method of attack can be carried out using a device that is capable of emulating any kind of tag or by producing a new tag as a duplicate of the old one (cloning). This kind of attack results in several tags with the same identity being in circulation.

Deactivation

These types of attack render the tag useless through the unauthorized application of delete or kill commands. Depending on the type of deactivation, the reader can either no longer detect the identity of the tag, or it cannot even detect the presence of the tag in the reading range.

Physical Destruction

Tags could be physically destroyed by chemical or mechanical means, or by using strong electromagnetic fields (like in a microwave oven). Active tags could also be shut down by removing or discharging the battery.

Detaching the Tag

A tag is separated physically from the tagged item and may subsequently be associated with a different item, in the same way that price tags are "switched." Since RFID systems are completely dependent on the unambiguous identification of the tagged items by the transponders, this type of attack poses a fundamental security problem, even though it may appear trivial at first sight.

Security Threats for the Wireless Interface

Eavesdropping

The communication between reader and transponder via the wireless interface is monitored by intercepting and decoding the radio signals. This is one of the most specific threats to RFID systems. The eavesdropped information could, for example, be used to collect privacy-sensitive information about a person. It could also be used to perform a replay attack, i.e., the attacker records all communicated messages and later on can either simulate this tag towards the reader or simulate this reader towards the tag.

Blocking

So-called blocker tags simulate to the reader the presence of any number of tags, thereby blocking the reader. A blocker tag must be configured for the respective anticollision protocol that is used.

Jamming

Jamming means a deliberate attempt to disturb the wireless connection between reader and tag and thereby attacking the integrity or the availability of the communication. This could be achieved by powerful transmitters at a large distance, but also through more passive means such as shielding. As the wireless interface is not very robust, even simple passive measures can be very effective.

Relay Attack

A relay attack for contactless cards is similar to the well-known man-in-the-middle attack. A device is placed in between the reader and the tag such that all communication between the reader and the tag goes through this device, while both tag and reader think they communicate directly to each other. Smartly modifying this communication could, for example in payment systems, lead to charging the wrong electronic wallet (a smart card with an RFID tag). To make this attack

more practical one could increase the distance between the legitimate card and the victim's card by splitting the device into two components: one communicating with the reader and one with the victim's card. The communication between these two components could be implemented by any kind of fast wireless technology.

Security Threats for the Reader

Falsifying Reader ID

In a secure RFID system the reader must prove its authorization to the tag. If an attacker wants to read the data with his own reader, this reader must fake the identity of an authorized reader. Depending on the security measures in place, such an attack can be "very easy" to "practically impossible" to carry out. The reader might need access to the backend in order, for example, to retrieve keys that are stored there.

Security Threats for Other Parts of RFID Systems

When considering the security challenges of RFID in a broader perspective, one has to take into account the infrastructure, including a back office where additional information of all tags is stored, and the aspect of convenience in use. A general RFID architecture is depicted in Figure 3.7.

RFID readers are generally connected to the middleware using modular drivers, much like Windows uses device drivers to communicate with a graphics card. This allows different readers to be used with the middleware, without having to modify the middleware. In addition to event processing, the middleware handles different kinds of user interfaces. A user interface is generally provided for system-management purposes, for example to modify the series of filters through which an event is passed. There will also be user interfaces that allow regular users to access the system and use it. For example, in a supermarket distribution center, there will be a user interface that provides information on the current stock levels.

The middleware also communicates with other software systems, which implement the application's business logic. To stay with the supermarket example, it is likely that the supermarket RFID system is connected to a stock management system, which orders new stock from suppliers before it runs out. When considering the broader RFID architecture, new security risks and countermeasures come to mind:

Tag-Borne Attacks at Back Office

One could foresee an attack at the back office through information stored at the tag, which was recently shown by a few Dutch students. Basically there are three types of RFID malware, which are listed in increasing complexity of implementation:

1. *RFID exploits*: Just like other software, RFID systems are vulnerable to buffer overflows, code insertion, and SQL injection.
2. *RFID worms*: A worm is basically an RFID exploit that downloads and executes remote malware. A worm could propagate through the network or through tags.
3. *RFID viruses*: An RFID virus starts with malicious content of a tag. When the tag is read out, this initiates a malicious SQL query that would disturb a database in the back office. Although such an attack has not yet been performed in practice, this type of threat cannot be excluded.

Misuse of Gateway Interface

The user interface to the gateway could be misused by unauthorized individuals to attack the integrity of the filters and to misguide the product management system.

Corrupted Drivers

The drivers that are used by RFID readers to communicate with the middleware could be corrupted. This could be done either by modifying the driver of a legitimate reader or by replacing the legitimate reader with a fake reader that has a corrupted driver. A corrupted driver could be used to attack and misguide the gateway.

Attacking the Reader-Gateway Communication

The communication between reader and gateway could be eavesdropped or modified.

Security Measures for the Tag

Security Measures to Prevent Unauthorized Modification of Tag Data (Contents and ID)

An obvious security measure to prevent modification of tag data is to use read-only tags for which unauthorized modification is intrinsically impossible. Another effective measure, also recommended for reasons of data management, is to shift all data except the ID to the backend. Some types of tags dispose of an authentication method (like the ISO-9798 standard), through which the reader can be authenticated by the tag such that only authorized readers can modify the tag contents.

Security Measures for Deactivation

Unauthorized application of delete commands or kill commands can be prevented by using an authentication method (when available).

Security Measures for Physical Destruction

A countermeasure for physical destruction of the tag would be a close mechanical connection between the tag and the tagged item to make it difficult to destroy the tag without damaging the item. To prevent discharging the battery of an active tag one could implement a sleep mode in the tag.

Security Measures for Detaching the Tag

A countermeasure for detaching the tag from the tagged item would be a tight mechanical bond between the tag and the tagged item to ensure that removing the tag will also damage the product. In the case of active tags, an alarm function is conceivable: a sensor determines that the tag has been manipulated and transmits the alarm to a reader as soon as it comes within range. For high-value items an option would be to manually check whether the tag is attached to the correct item.

Security Measures for the Wireless Interface

Security Measures for Eavesdropping

An effective measure to reduce the effect of eavesdropping is to shift all data to the backend. More advanced tags have a module to encrypt the communication with the reader, which also prevents eavesdropping. Such advanced tags cannot be read out by intruders and are still available for legitimate use. Another measure would be to design the RFID system such that tags are used within a small range, which is just sufficient for the legitimate readers (and thereby shutting out a class of unauthorized readers).

Security Measures for Blocking

There are no technical measures to prevent the use of blocker tags, but a solution is to ban their use in the standard terms and conditions of business.

Security Measures for Jamming

It is possible to detect jamming transmitters by performing random measurements or by using permanently installed field detectors.

Security Measures for Relay Attacks

One way to guard against relay attacks is to shield the tag when it is not used, e.g., by putting the tagged card in a Faraday-like cage. Another way is to require an additional action by the user (push a button, type in a PIN code, or use a fingerprint) to activate the tagged card, although this solution eliminates some of the convenience of the contactless system.

Security Measures for the Reader

Security Measures for Falsifying the Reader ID

To prevent readers from falsifying their ID to obtain unauthorized access to a tag, an authentication method (when available at the tag) can be used to authenticate the reader towards the tag. This risk can be further reduced when the reader has to access the backend during the authentication procedure, e.g., to retrieve cryptographic keys. Note that these measures are designed to assure the integrity of a reader that is about to communicate with the tag. For measures like shielding, which prevent an unauthorized reader from communicating, see “Security measures for eavesdropping”.

Security Measures for Other Parts of RFID Systems

Security Measures for Tag-Borne Attacks at Back Office

To avoid such attacks, the content of tags should be checked by the reader and regular security measures should be taken to protect the gateway. A typical countermeasure against RFID viruses is to improve the software in the gateway that is able to distinguish a regular tag ID from an SQL query such that these attacks can be prevented from entering a database.

Security Measures for Misuse of Gateway Interface

To prevent such an attack the user interface should be provided with some kind of authentication mechanism such that only authorized users are able to access the gateway. Another measure would be to place the gateway and the user interface in a physically protected room such that only authorized employees that have access to this room can access the user interface.

Security Measures for Corrupted Drivers

A possible solution to this problem is to use only signed drivers, i.e., each legitimate driver should be digitally signed such that the gateway can check that communicating readers contain a legitimate driver. The use of drivers enables the fact that different readers can be used to communicate to the gateway. From a security point of view the use of different readers should be encouraged because an attack is likely to be specific for one type of reader or one type of driver, so a diversification of types lowers the impact of a possible attack.

Security Measures for Attacking the Reader-Gateway Communication

The communication between reader and gateway could be eavesdropped or modified.

Security Measures against Cloning

When considering one tag and one reader as a system, which has been done in the previous sections, the risk of cloning (duplication of the tag ID in a new tag) has been identified. Only in the broad view of the complete architecture, such a risk could be handled: in the database where all the different tag IDs (with respect to a specific application) are collected, a duplicate ID could be detected and in some cases even the clone could be recognized (i.e., be distinguished from the original tag).

Conclusion

This document presented an overview of some of the technical facets of RFID. The most striking lesson here is that even though RFID is a conceptually simple technology, it engenders technological questions and problems of formidable complexity. For this reason, it is unwise to view RFID privacy and security as a technological issue alone. Policymaking will also have a vital role to play in the realm of RFID. They must not only supplement the protections that technology affords, but must prove sensitive to its novelties and nuances. To be most effective, RFID security controls should be incorporated throughout the entire life cycle of RFID systems—from policy development and design to operations and retirement. A delicate balance between privacy and utility is needed to bring RFID to its high pitch of promise.

Chapter 4

Privileged User Management

Georges J. Jahchan

Introduction

On June 22, 2008, Terry Childs made national headlines when he locked access to the city of San Francisco's Fiber Wide Area Network (WAN) by resetting administrative passwords to its switches and routers and then declining to hand over those passwords.*

On October 24, 2008, an employee at Fanny Mae's Urbana, MD, data center was let go from his contract, almost two weeks after erroneously creating a computer script that changed the settings on the Unix servers without the proper authority of his supervisor.† Within 90 minutes of being told his contract was terminated, and several hours before his access to the Fannie Mae network was disabled later that evening, he embedded a malicious script inside a legitimate script that ran on Fannie Mae's network every morning. The malicious script was to trigger on January 31, but was discovered by chance by another engineer on October 29. The malicious script was planted after a page of blank lines intended to conceal it. Had that script run, it would have disabled monitoring alerts and all log-ins, deleted the root passwords to the approximately 4000 Fannie Mae servers, then erased all data and backup data on those servers by overwriting with zeros. It would have caused millions of dollars in damage and reduced or shut down operations for at least a week.

March 17, 2009: An IT contract employee was indicted on charges of sabotaging a computer system he helped set up, because the company did not offer him a permanent job.‡ He was charged with affecting the integrity and availability of an offshore platform monitoring computer system designed to detect oil leaks. While working as a contract employee, he had set up multiple accounts that he used to illegally gain access to the system after he stopped working for the company.

---

* http://www.computerworld.com/s/article/9110470/Questions_abound_as_San_Francisco_struggles_to_repair_locked_network.

† http://www.computerworld.com/s/article/9127040/Fannie_Mae_engineer_indicted_for_planting_server_bomb.

‡ http://www.computerworld.com/s/article/9129933/IT_contractor_indicted_for_sabotaging_offshore_rig_management_system.

In the three cases, had appropriate privileged user controls been implemented, the incidents could have been prevented, or at least detected in a timely manner so as to minimize damage. In the second and third case, dame luck averted disaster, but organizations should not count on their lucky star to safeguard their information assets.

With the job cuts and corporate belt tightening, resulting from the faltering economy, companies are advised to be especially vigilant with disgruntled employees.[*]

A disgruntled administrator is not the only threat to an organization. Outsiders who manage to steal legitimate privileged users' credentials and then use them to gain access to high-value targets pose an even greater threat. Hackers who band together in across-the-globe virtual communities and share knowledge are the most dangerous as they combine their diverse skills to mount innovative and highly sophisticated targeted attacks. They will typically carefully plan their attacks and break into a system outside working hours, when they are most likely to go unnoticed, plant a back door that subsequently grants them root access, and leave as quickly as possible, erasing their tracks.

Computer Emergency Response Team (CERT's) published analysis of 150 insider attack incidents[†] classified the attacks into three categories: fraud, theft of information, and IT sabotage, and identified the typical perpetrators' profiles. Following is a summary table of the results.

Type of Incident	*Typical Insider's Profile*	*Method of Attack*	*What was the Motive?*	*How was the Incident Detected?*	*How was the Insider Identified?*	*What was the Impact?*
Fraud	Nontechnical nonmanagement positions with privileged access Nontechnical means	Used their own credentials Acted during business hours from within workplace	Greed	System irregularity Nontechnical means	System logs	Financial impacts on employer Impact on innocent victims
Theft of confidential information	Male employees Half had accepted other position Half were in technical positions	Used own credentials Half compromised an account	Disgruntled Financial gain Did not know it was wrong	Half by system irregularity Nontechnical means	System logs	Financial impacts on employer Organization and customer confidential information revealed Trade secrets stolen Innocent victim murdered Insider committed suicide

[*] http://www.computerworld.com/s/article/9117138/Tough_economic_climate_can_heighten_insider_threat.

[†] A Risk Mitigation Model: Lessons Learned From Actual Insider Sabotage; Dawn M. Cappelli, Andrew P. Moore, and Eric D. Shaw. November 7, 2006.

Type of Incident	*Typical Insider's Profile*	*Method of Attack*	*What was the Motive?*	*How was the Incident Detected?*	*How was the Insider Identified?*	*What was the Impact?*
IT sabotage	Former employees Male Highly technical positions	Disgruntled Revenge for negative work-related event	No authorized access Backdoor accounts, shared accounts, other employees' accounts, insider's own account Many technically sophisticated Remote access outside normal working hours	Manually by nonsecurity personnel System failure or irregularity	System logs Most took steps to conceal identity and/or actions	Inability to conduct business, loss of customer records, inability to produce products Negative media attention Private information forwarded to customers, competitors, or employees Exposure of personal or confidential information Website defacements Many individuals harmed

CERT recommends that management must recognize the technical precursors and have the "ability to disable access on demand in the absolute, particularly for administrators and privileged users" when demoting or firing. In practice, that requires an understanding of the access paths available to insiders, which "depends on rigorous access management practices." Access management must be proactive and ongoing, as "practices tend to degrade over time without regular enforcement" and "it takes time to recover from poor access management practices."

Effective privileged user controls need to combine policies, procedures, and technologies that address the particular environment and needs of organizations. Though there may be similarities, no two organizations' environments are alike. Consequently, what works very well for one organization may not for another in the same line of business. In security, there is no such thing as "one solution fits all." While the administrative and operational controls are particular to a company, the technology controls are licensed for a variety of tools available from specialty security vendors.

At a high level, organizations use technology solutions to come as close as possible to the goal of consistently granting and controlling rights based on the principle of least privilege (or access on a need-to-know and/or need-to-do basis). Furthermore, auditors come to expect organizations to prove it. Regulations further complicate matters with regulation-specific control/audit/reporting requirements.

The technical controls to manage privileged users fall in three main categories: privileged password management, privileged user access controls, and identity and access management suites. Some vendors have products that fall into more than one category, others are platform specific (Windows-only or *nix-only), while others are cross-platform. This paper explores some of the products that help automate and enforce rigorous privileged account management practices. The content of this article is based on published information from vendors and independent sources.

The information is reported "as is"; no attempt was made to validate vendor claims, as product evaluations are beyond the scope of this article.

Full management of privileged accounts requires organizations to not only control who has access to privileged credentials, but also, once access is granted, restrict use exclusively to perform specific tasks, and maintain an audit trail of user actions. As we will see, none of the solutions reviewed meet all of the goals.

The solutions are categorized, but they are not sorted in any particular order.

Privileged User Audit Solutions

Centrify DirectAudit

Centrify DirectAudit helps comply with regulatory requirements, perform in-depth troubleshooting, and protect against insider threats for UNIX and Linux systems. DirectAudit's detailed logging strengthens compliance reporting and helps spot suspicious activity by showing which users accessed what systems, what commands they executed, and what changes they made to key files and data. With DirectAudit one can also perform immediate, in-depth troubleshooting by replaying and reporting on user activity that may have contributed to system failures. And its real-time monitoring of current user sessions enables spotting of suspicious activity.

The DirectAudit Agent continuously communicates user session activity in an encrypted, compressed format to a DirectAudit Collector Service. The Collector Service in turn stores the data in a central SQL Server repository, providing enterprise-scale performance and scalability. For increased reliability, the DirectAudit Agent continues to record session data even when there is no network connection and subsequently forwards it to a DirectAudit Collector Service when the network is available. Centrify also supports load balancing among multiple DirectAudit Collector Services when deployments of DirectAudit Agents range in the 100s or 1000s.

In the DirectAudit Console, a right-click can replay any user session on any audited system to see what commands were executed, what changes were made to key files and data, and what system output appeared. Pause, rewind, or fast-forward are similar to using a VCR. This playback feature gives a tool for monitoring activity, troubleshooting changes that may have led to a system failure, or documenting system configuration tasks.

The DirectAudit Console's out-of-the-box views provide visibility into active sessions and historical sessions, or custom-built views that show sessions by specific users, machines, time periods, or other criteria. They can also perform full-text searches to find, for example, all instances of a password command across all sessions. DirectAudit adopts a non-proprietary SQL data format, enabling reporting and querying through third-party tools.

The DirectAudit Console gives a centralized, real-time view of every user session on every audited UNIX and Linux system. For each session one can see who is logged on and one can immediately drill down to see what they are currently doing. The console allows spotting of suspicious activity and aids in troubleshooting system issues.

IBM Tivoli Compliance Insight Manager

Consul Insight Suite integrates log management, rules- and policy-based monitoring, and reporting. It normalizes, analyzes, and reports on privileged user activity.