

Realtime
publishers

"Leading the Conversation"

The Essentials Series

Messaging & Web Security

Volume II

by Dan Sullivan

Article 1: Issues with Identity Management and Access Control.....	1
Technical Elements of Identity Management	1
SSO	2
User Provisioning.....	2
Entitlement Management	2
Auditing and Monitoring	3
Policy Management	3
Article 2: Authentication’s Role in Access Control	4
Authentication Techniques	4
Passwords.....	4
Security Tokens	5
Biometrics	5
One-Time Passwords	5
Pros and Cons: What Works and When.....	6
Article 3: Unified Threat Management Basics	7
UTM Solutions: Basic Components	7
Advanced UTM Features.....	8
Article 4: Issues with Remote Access.....	9
How SSL VPNs Work	9
Risks to SSL VPNs.....	10
Mitigating SSL VPN Risks.....	10
Remote Device Scanning.....	10
Spyware Suppression	11
Virtual Encrypted Environment.....	11
One-Time Passwords	11
Cache Purging.....	11
Essential Security.....	11
Article 5: Combating Botnets: What Can Be Done?	12
Improved Client Device Security.....	12
Network Monitoring Techniques.....	13
Changes in ISP Market and Regulations	14
Article 6: Phishing Techniques and How to Protect Against Them	15
Phishing Techniques	15

Deception in Multiple Forms	15
Phishing Trojan Horses	16
Spyware Attacks	16
Don't Be a Phishing Victim	17
Article 7: Measuring Security: Application Metrics	18
Code Reviews	19
Vulnerability Scanning	20
Runtime Metrics	20
Summary	21
Article 8: Five Things to Know about SQL Injection Attacks	22
Potential SQL Injection Vulnerabilities	23
Probes Reveal Information About the Database	24
SQL Injection Attacks Can Manipulate Data and Database Structures	24
Use Bound Parameters and Stored Procedures to Query and Manipulate Data	25
Minimize Privileges	26
Summary	26
Article 9: Four Things to Know to Protect Your Network from Unmanaged Devices	27
What Are Unmanaged Devices?	27
What Security Threats Do Unmanaged Devices Pose?	28
What Functions Are Required in a NAC System to Protect Against Those Threats?	28
What Are the Limits of NAC Protections Against Unmanaged Devices?	29
Article 10: Security and Service-Oriented Architectures	30
New Entry Points into Applications	30
Limits of Network Security Measures	31
Need for Encryption	32
Challenges with Access Controls	32
Article 11: Multiple Layers of Database Security	33
Network and OS Security Specific to Database Servers	34
Database Listeners	34
OS Access	34
Who Can You Trust: Database Federation	35
Potential Vulnerabilities in Applications that Use Databases	35
Encrypting Data	36

Article 12: Integrating Security Functions: What Makes Sense?	37
Functional Similarities	38
Performance	38
Management and Organizational Issues	39
Summary	39
Article 13: The Rise of PDF Spam and How to Stop It.....	40
Spamming Methods	40
Controlling Plain Text Spam	41
Image Spam Sneaks by, for a While.....	41
PDF and Other Document Spam.....	42
Article 14: Elements of Effective Event Log Monitoring	43
Consolidated Management and Reporting.....	44
Comprehensive Monitoring and Reporting	44
Targeted Analysis and Reporting.....	45
Summary	45
Article 15: Ten Tips for Securing MySQL Databases.....	46
Run mysqld as an Ordinary User	46
Use Grant and Revoke to Limit Access.....	47
Review Granted Privileges Regularly.....	47
Use Strong Passwords.....	47
Validate All Input	47
Limit FILE Privilege to Database Administrators	48
Do Not Index Confidential Data.....	48
Patch the Database and OS	48
Limit Access to the Database Server to Trusted Devices	48
Isolate and Minimize Code that Interacts with the Database.....	48
Summary	49
Article 16: Establishing Organizational Security	50
Managing Risks	50
Sharing Responsibility for Risk	51
Use Security Metrics.....	51
Realistic and Enforceable Policies.....	52
Implement New Security Initiatives with Other Initiatives	52

Article 17: Payment Card Industry Data Standards and Data Loss Prevention.....	53
Preventing Data Loss Over the Network	54
Minimizing Application Exposure.....	54
Network-Based Data Loss Prevention	55
Database Encryption	55
Summary	56
Article 18: Web Developers' Guide to Avoiding Cross Site Scripting Attacks.....	57
Which Applications Are Vulnerable?.....	57
Preventing XSS.....	58
Allow-In Filtering	58
Keep-Out Filtering	59
Article 19: Evolving Social Engineering Elements of Phishing.....	60
Social Engineering: The Basics	60
Improving Context in Phishing Lures.....	61
Phone Phishing	62
Online Shopping Phishing	62
Summary	62
Article 20: Rootkit Countermeasures.....	63
Rootkit Countermeasure 1: Basic Blocking.....	63
Rootkit Countermeasure 2: Local Antivirus Defenses	64
Rootkit Countermeasure 3: Blocking Global Hooks	64
Rootkit Countermeasure 4: Trusted Computing Platform-Based Countermeasures	65
Summary	65
Article 21: Using Host-Based Intrusion Prevention Systems to Prevent Data Loss.....	66
Ways to Steal and Lose Data	66
Blocking Data Loss with Host-based IPS.....	68
Summary	68
Article 22: 5 Evaluation Criteria for Selecting a Data Loss Prevention Product.....	69
Policy Enforcement.....	69
Classification Accuracy	70
Reporting Features	70
Network and Host-based Implementations.....	71
Platforms Supported.....	71

Summary	71
Article 23: Human Factors in Improving Application Security.....	72
Human Factor Considerations in Application Design	72
Agreements Between Trusted Sites and Users	72
Visibility and Understandability	73
State and Security Status.....	75
Summary	75
Article 24: 5 Issues to Consider with Mobile Device Encryption	76
Policy Management	76
Scope of Encryption.....	77
Authentication Mechanisms.....	77
Supported Platforms.....	77
Compliance	78
Article 25: Basics of Database Auditing.....	79
What Is Auditing?.....	79
What to Audit in a Database?	79
Login and Logout Events.....	80
Changes to Data Structures.....	80
Privilege Changes	80
Database Errors.....	81
Modifying and Viewing Sensitive Information	81
How to Audit?.....	81
Article 26: 5 Security Considerations with Portals.....	82
Role of Identity Provider	82
Propagating Identities	83
Client-Side Security	84
Service Deployment and Patch Management	84
Portal Administration.....	84
Article 27: Role of Code Reviews in Application Security	85
Identifying Code to Review	86
Parts of a Code Review.....	86
Summary	87

Article 1: Issues with Identity Management and Access Control

Web-based applications are now taking advantage of service-oriented architectures, and with that comes increased challenges in managing identities and access controls across distributed architectures. For example, a user in Company A may place an order with Supplier B who performs a credit check with a credit bureau, Company C, and arranges shipment through Company D. It is quite conceivable that the entire process is automatic and involves no human intervention beyond deciding what to order. This raises several questions:

- What is the user at Company A allowed to do with Supplier B's order system?
- Is the user at Company A allowed the full credit extended to Company A or is there a limit to the charges that can be incurred?
- Did Company A grant permission to Supplier B to perform a credit check? If so, how does Company C verify this?
- The user at Company A will want to track the status of the order; how will Company D know to provide information to that user but not others?

All these questions center around knowing who is making a request for a service and the authorizations granted to that person—or in some cases, an agent application acting on behalf of a user. To realize the benefits of this idealized scenario and others like it, you must address both technical and organizational issues.

Technical Elements of Identity Management

Identity management is not a single isolated application or process but consists of several component services:

- Single sign-on (SSO)
- User provisioning
- Entitlement management
- Auditing and monitoring
- Policy management

SSO

SSO is an application that allows users to specify a single username and password set, or other credentials, to access multiple applications. The SSO system responds to the authentication requests from other applications on behalf of the user. The SSO system can store credentials for each application the user is allowed to access.



This is different from a password synchronization application, which is a program that maintains the same username/password pair for each application.

Most SSO systems integrate relatively easily with Web-based applications but legacy mainframe applications may require some custom coding.

User Provisioning

Users are constantly being added, removed, and updated in identity management systems. Unless the provisioning of user accounts is automated, the cost of user management can quickly get out of control. An automated user provisioning system is a crucial part of an identity management system and should include functions for

- Establishing a user's identity
- Creating user accounts on relevant applications
- Determining user privileges based on the user's role in the organization
- Supporting an approval process, such as ensuring manager's approvals, if necessary, are obtained before access is granted
- Removing access rights when a user changes roles or leaves the organization

The ability of the provisioning system to integrate with an organization directory can improve the cost effectiveness of the system. Ideally, the user provisioning system should integrate with the organization's Lightweight Directory Access Protocol (LDAP) or Active Directory (AD), using attributes of the user's identity in the directory to determine access rights.

Entitlement Management

When creating a user's accounts, one must also specify what the user is allowed to do. This should be a policy-based process based on the user's role within the organization. For example, a financial analyst in the audit department would likely need read access to data in the general ledger system but not insert, update, or delete privileges. An accounts receivable clerk, however, would need far more limited read access to the general ledger but would require the ability to add entries to the accounts receivable system. Clearly, information about the user's role in the organization plays a key part in entitlement management as well.

Auditing and Monitoring

With access controls a central element of information security, the ability to audit identity management events is crucial. Among the types of events you should be able to audit:

- Creating and removing identities
- Modifying privileges to an identity
- Changing SSO credentials
- Using identities

Monitoring is also important because it provides the ability to correlate events across security applications. For example, a host intrusion prevention system (IPS) might detect a change to a monitored file; systems administrators might want to know which users were logged on at the time the file was changed.

Policy Management

Policies provide the means to implement specific rules within an identity management system and can vary:

- Password policies
- Assigned privileges based on roles
- Time controlled access to applications
- Location controlled access to systems

The advantage of policy-based management is the consistency it provides. Rather than having to enforce the details of a particular policy for every identity to which it applies, a policy can be defined for a set of users and the identity management system can enforce it automatically for all members.

The technical elements of identity management provide the means to automate and more effectively apply security policies. A future article will examine how the organizational aspects of identity management can contribute to, or detract from, those benefits.

Article 2: Authentication's Role in Access Control

Authentication is the process of confirming the identity of a person or agent using a system. Now, of course, there is no way for a machine to verify someone's identity with 100% certainty, but you can use techniques that make you confident that persons or agents are who they say they are to a level appropriate for your needs. For example, to authenticate to a shopping Web site, a username and password is usually sufficient; to access a check-processing system within a bank will require significantly more for authentication. What is that "more" and how do you get it? That is the subject of this article.


The first section of this article will look at several authentication techniques with varying levels of complexity and difficulty in cracking. The second part describes some of the advantages and disadvantages of the techniques, along with pointers about which techniques are appropriate for different situations.

Authentication Techniques

Several types of authentication techniques are commonly used today:

- Passwords
- Security tokens
- Biometrics
- One-time passwords


These may be used separately or in combination, known as two-factor or multi-factor authentication.

 Multi-factor authentication is becoming more popular as financial services firms turn to it to improve online banking security. See for example, the Federal Financial Institution Examination Council's [Authentication in an Internet Banking Environment](#) and a critical review of its potential limitations in [The Failure of Two-Factor Authentication](#) by Bruce Schneier.

Passwords

Passwords are the most common and the weakest authentication mechanism. The problem with passwords is twofold. First, if they are memorable, they are probably easy to crack. Password-cracking tools are readily available online, so systems administrators can use these to check the relative strength of passwords—and they may be surprised by what they find. Password policies can be enforced that require passwords to be a minimum length, include a combination of alphanumeric and punctuation characters, and have a limited lifetime. The challenge with policies is to balance the strength of the password with the ability of the user to remember it; otherwise, the user might write down the password, effectively trading one security weakness for another.

The second problem with passwords is the number of distinct ones that are required for a single user. The proliferation of applications, Web sites, and operating systems (OSs) using passwords means users have to remember more passwords, which leads to written records, or using the same password repeatedly, which is yet another security weakness. Single sign-on (SSO) systems address this problem but may not work with all applications, especially legacy applications.

 Another option for reducing the risks associated with passwords is to use a password repository such as the open source tool, [Password Safe](#).

Security Tokens

A security token is a hardware device that generates a key that is required to access a system. For example, a user may try to access a virtual private network (VPN) and have to enter a code from the token. The token may update the key value relatively frequently (for example, every minute) if it is a time-synchronized type, or update the key value after each use if it is a one-time password type. In both cases, authentication is dependent upon possessing the token so that the appropriate key value can be entered.

Biometrics

Biometric authentication is based on physical characteristics of the user. This class of authentication techniques includes:

- Fingerprints
- Palm scans
- Hand geometry
- Retinal and iris scan
- Signature dynamics
- Voice prints
- Keyboard dynamics

In each case, a difficult-to-forge characteristic of the user is initially measured and the results stored in a secure repository. After that, when the person attempts to authenticate to the system, the characteristic is measured again and compared with the reference measurement. If the two match sufficiently, the user is authenticated.

One-Time Passwords

One-time passwords are one of the most secure forms of authentication because even if a password is intercepted, it can not be used again to access a system. One-time passwords are pseudo-randomly generated based on the previous password, the time, or a challenge-response posed to the user.

One-time passwords that are time synchronized are less susceptible to phishing attacks because even if the user is tricked into disclosing the next password, the attacker would have a limited time to exploit the password.

Pros and Cons: What Works and When

Each of the technologies described has benefits and drawbacks outlined in Table 1.

	Pros	Cons
Passwords	Inexpensive, well understood	Users often have to trade off between easy-to-remember or easy-to-crack passwords
Security tokens	Improved security over passwords; can be used to implement one-time passwords	Can be expensive in large deployments; additional overhead to manage lost or stolen tokens
Biometrics	Difficult to forge; good discrimination	Potential for attackers to intercept a biometric measurement in transit and reuse; some users may be resistant to biometric measures
One time password	Eliminates risk of re-uses once a password has been used	Non-time-synchronized passwords are subject to phishing attacks

Table 1: Pros and cons of authentication techniques.

Cost is of course a major factor in choosing an authentication mechanism. Passwords, although potentially weak, are still the least expensive. When passwords are used, policies should be enforced and password repositories employed. Security tokens and biometrics are more expensive and more secure than passwords and preferred, from a security perspective, when cost is not prohibitive. One-time passwords offer strong security without some of the risks of passwords and biometrics. Expect their use to grow.

Article 3: Unified Threat Management Basics

Unified threat management (UTM) is a consolidation of multiple security services into a single device. UTM is a natural evolution of products from firewall, intrusion prevention system (IPS), and content filter solution vendors. From the vendor perspective, the move to UTM products allows them to package and support a more comprehensive security offering that provides a key advantage to customers—easier management. From a security manager's perspective, UTM products consolidate essential security measures, which typically include a consolidated management function.

UTM Solutions: Basic Components

UTM products are network appliances and are therefore generally built on hardened operating systems (OSs) running optimized for the security services offered by the devices. The security services can include:

- Firewall
- Content filtering
- URL filtering
- Antivirus
- Anti-spam
- Intrusion prevention
- Encryption
- Virtual private networking (VPN)

Simply stringing together these components, though, is not enough to constitute a UTM device. UTMs also provide centralized management over multiple operations. Key management operations include:

- Patch management
- Updating antivirus signatures
- Access to real-time URL reputation filters
- Status reporting
- Policy definition

All UTMs at this point can be expected to provide most if not all of these services. A number of other services, however, are not always found in every UTM device.

Advanced UTM Features

UTM appliances can distinguish themselves based on both security and management features. For example, some UTMs provide support for virtualizing the device and making a single device appear to function as several. This is useful in UTMs, as in other servers, because it allows systems administrators to consolidate multiple devices in a single unit. In the case of UTMs, this means network administrators can segment network traffic and apply different policies to each segment while using a single appliance.

Another feature found in some appliances is the ability to differentiate quality of service (QoS) for different types of traffic. For example, real-time video conferencing of voice over IP (VoIP) traffic may be guaranteed higher throughput than email traffic while still subjecting both to appropriate scans.

Although there are many cases in which all peer-to-peer traffic may be blocked, some network administrators may want to allow some types, such as instant messaging, while blocking others, such as peer-to-peer file sharing. The ability to distinguish peer-to-peer traffic by client is a useful feature.

A feature that can be easily overlooked is the usefulness of the appliance's default configuration. As there are so many parameters and features that can be configured in a UTM, it is helpful to have multiple policy configurations predefined for a range of possible uses.

Another area that is easy to overlook is reporting, especially in today's compliance-conscious environment. UTMs should provide basic reporting, but more advanced features, especially the ability to customize reports with user-defined filter criteria and report layout selections, can be useful.

Article 4: Issues with Remote Access

The need to access enterprise applications from a wide variety of locations is a long-standing problem. Early solutions based on dial-in protocols met some of the common needs, but that solution was eclipsed by IPsec virtual private networks (VPNs). IPsec VPNs have their own advantages but a significant drawback is the need for client-side software. Employees on the road, business partners, and even customers who need to access network or application services from unmanaged devices do not benefit from IPsec VPNs. Remote access methods based on SSL take advantage of ubiquitous browsers that already support SSL. Browser-based access equates with easier maintenance and more flexibility for end users. Needless to say, there are still issues that network administrators face when implementing SSL VPNs.

This article will look into several topics:

- Basics of SSL VPNs
- Risks posed by the use of SSL VPNs
- Strategies for addressing those risks

Let's begin with an overview of SSL VPN technologies.

How SSL VPNs Work

SSL VPNs are essentially Web proxies that act as a middle agent between a client and server. The proxy passes client requests to the server and sends the response, in encrypted form, back to the client. The process is relatively straightforward for Web pages, but applications, especially non-Web applications, can be more complex.

In some cases, SSL VPNs use small downloadable applications that listen for traffic on an application-specific port. When a packet is received on that port, it is encrypted and sent to the server-side VPN where it is decrypted and sent on to the application server.

Additional network extension functionality is available in SSL VPNs that can perform authentication checks, such as verifying IP addresses of client. With such functionality, administrators can define policies limiting which devices can connect to the network as well as which users.

Risks to SSL VPNs

One of the most prevalent risks to SSL VPNs stems from the lack of control over client devices. Consider the problems a user might find when using a public access computer, such as in a hotel business center:

- The PC is infected with viruses, worms, or other malware
- Keyloggers, video frame buffer capture, and other kinds of spyware may be operating on the device
- Information cached in Web browsers may be accessed by others once the user has disconnected from the remote network

These risks highlight the limits of SSL VPNs: although SSL is sufficient to encrypt data transmitted between the client and server, it cannot guarantee the security of the information once it is decrypted on the client. Clearly, additional measures are required.

Mitigating SSL VPN Risks

Significant risks associated with SSL VPNs stem from lack of control over the device; however, there are limited measures that can be taken to minimize risks:

- Remote device scanning
- Spyware suppression
- Virtual encrypted environment
- One-time passwords
- Cache purging

Each of these measures offers some protection, but using them in combination provides much more comprehensive security.

Remote Device Scanning

Before establishing a VPN connection with a client, the VPN server may download a small, transient client program that can scan the client device to ensure that the device meets certain policy requirements:

- Up-to-date antivirus software
- Personal firewall
- No executing spyware

If the policies are not met, no connection is established.

Spyware Suppression

Spyware often takes advantage of OS support for hooks, which are functions that allow programs to respond to specific events. For example, Windows hooks allow programs to detect when a keystroke is typed and which key is typed. By checking for programs that make use of hooks, the remote scanning programs can mitigate the threat of spyware.

A potential problem with this measure is that some legitimate programs make use of spyware-like techniques. SSL VPN clients should be flexible enough to allow a user to control the level of spyware suppression. For example, a user may be allowed to leave suspicious programs running as long as keystroke logging is blocked and a virtual encrypted environment is used.

Virtual Encrypted Environment

Another method for protecting information on unmanaged devices is to encrypt application data that resides on the client device. By using a “virtual encrypted environment,” data that is stored in a browser cache is encrypted to prevent spyware from exploiting the information. With this approach, even if spyware is left running on the client device, it cannot gather any useful data.

One-Time Passwords

Continuing with the “belts and suspenders” strategy, another useful measure is one-time passwords. The obvious advantage is that even if spyware can capture a password, the password is of no use in establishing a second connection to the application.

Cache Purging

Finally, once a user is finished and the VPN connection is terminated, all user data about the session should be purged. This purge includes data in the browser cache, cookies, and navigation history.

Essential Security

Securing remote access is essential for many organizations. Although rich-client IPSec methods have been used in the past, lighter-weight SSL VPNs are more popular today. These solutions work well but have limitations. Fortunately, techniques such as spyware suppression, remote scanning, virtual encrypted environments, one-time passwords, and cache purging can mitigate some of the risks that remain with current remote access technologies.


Article 5: Combating Botnets: What Can Be Done?

Recent monitoring of botnet activity indicates a sudden and still unexplained surge. According to a recent posting at ShadowServer.org, the number of identified botnets quadrupled in 3 weeks. This activity cannot be explained by the rapid growth of one or two botnets or the introduction of new botnets. The wide distribution of growth may be indicative of a newly discovered vulnerability, but no conclusive results can be drawn from the available data. So what can be done?

This article will look at some ready-to-deploy defensive measures as well as potentially useful techniques for combating botnets including:

- Improved client device security
- Network monitoring techniques
- Changes in ISP market regulations

As with most security problems, a combination of countermeasures is likely to provide the most comprehensive and effective outcome.

 ShadowServer.org is a volunteer group of security professionals who monitor and report on malware, botnet, and electronic fraud. Their goal is to raise awareness of compromised servers, the spread of malware, and other malicious activities. ShadowServer.org works with security agencies to meet its objectives; it is not a cyber-vigilante group.

Improved Client Device Security

Advocating improved client device security is like telling people to eat their vegetables and get exercise. Many people already know they should be doing this but do not manage to get it done. One solution to the problem is to make client device security easier. Microsoft is trying to make inroads in this arena with improved security on Vista. However, Vista requires significantly more hardware than previous versions of Windows, so adoption rates are tied to hardware upgrade cycles. In addition, Vista is still vulnerable to attacks and additional countermeasures are required. Even if well-managed IT departments deployed Vista, hardened the configuration, and controlled changes to devices, bot herders could still find plenty of vulnerable devices among home users who are constantly connected to the Internet via broadband services.

In spite of the obvious limitations of improved client device security as a way to deal with botnets, it is still part of the solution, thus, it is worth outlining the basic steps you should follow:

- Install and keep antivirus software and personal firewalls updated
- Patch the operating system (OS), browsers, and other applications
- Turn off the device when not in use
- Use site integrity checking services such as McAfee Site Advisor or Exploit Prevention Labs LinkScanner

The fact that all these measures are available today yet the size and threat of botnets is still growing means these measures are far from sufficient to deal with the problem. Botnets must be countered at the network level as well as at the device level.

Network Monitoring Techniques

Researchers have developed metrics for assessing botnet activity. Three metrics, in particular, have proven useful in tracking botnet activity:


- Relationship
- Response
- Synchronization

The relationship in a centrally managed botnet can be detected by tracking communications from the central server, controlled by the bot herder, to the bots. It is unclear, from published reports, how well this metric works with peer-to-peer-based botnets.

The second metric, response, should work equally well in botnets regardless of their control structure. This metric captures the different response times between bots and humans. Typically, bots responded consistently according to their programming, whereas human response times vary by person and by task.

Bots also exhibit synchronized activities. They will launch Denial of Service (DoS) attacks at the same time or generate the same kinds of spam messages simultaneously.

These activities are used in combination to detect bots because each one individually can lead to false positives. For example, bot programmers could randomly vary response times to avoid detection by response-time metrics. Online, massive multi-player games can also exhibit synchronization or relationship patterns similar to botnets.

 For more information about botnet detection metrics, see "[A Proposal of Metrics for Botnet Detection Based on Its Cooperative Behavior](#)" by Mitsuaki Akiyama, et. al.

Changes in ISP Market and Regulations

ISPs have many advantages when it comes to monitoring and controlling botnets. The problem is that as businesses, costs and revenues drive these organizations. Monitoring and shutting down botnets can incur costs without generating revenues. Basic market forces are not enough to move ISPs, as a rule, into doing enough to combat botnets; more is needed.

One method is to raise public awareness about botnets and promote ISPs that actively take on the responsibility to control botnets. ShadowServer.Org does so with its “Hall of Fame,” which lists ISPs and other service providers that actively work to control botnets.

 See <http://www.shadowserver.org/wiki/pmwiki.php?n=Involve.HallOfFame> for ShadowServer.Org’s current list of Hall of Famers.

The cost of botnets is a negative externality, something like pollution in industrialized societies. We all pay a cost because of botnets but that cost is not reflected in the cost of ISP services. Until ISPs have financial incentive to control botnets, many will not do so. This is a classic situation in which regulation may have to be used to correct a deficiency in market forces.

Regulations need not be onerous. For example, ISPs could be required to:

- Monitor traffic using techniques described earlier
- Establish honeypots as surveillance devices
- Contribute to third-party monitoring organizations
- Establish procedures for rapidly responding to evidence of botnet activity

Botnets are growing in number rapidly and becoming more sophisticated. Centralized command and control structures are giving way to peer-to-peer models that are more difficult to shut down. As botnets move from centralized architectures, so should countermeasures. Active monitoring techniques can help detect botnet activity and coordinated responses from ISPs can then shut them down. Controlling botnets at the client and server level is no longer sufficient (if it ever was). Botnets are a network-scale problem and will have to be addressed at that level.

Article 6: Phishing Techniques and How to Protect Against Them

Phishing is a form of spam that attempts to trick readers into disclosing valuable personal information or installing malicious software on their computers. Phishing combines both technical and social engineering aspects in attacks and, not surprisingly, so do effective countermeasures.

Phishing Techniques

When one receives a phishing message, you can expect one or more of the following techniques to be used:

- Deception
- Trojan horses
- Spyware

Deception in Multiple Forms

Deception is common to all phishing attacks. It begins with harvesting email accounts from Web sources, generating likely email addresses, or collecting them from sites that ask visitors for email addresses, purportedly for legitimate purposes. Deception is also part of the email delivery process.

Many spam messages are generated by botnets, networks of compromised computers controlled by the spammer. These bots may deploy their own simple mail servers or send them through throw-away email accounts on legitimate email services.

The next kind of deception commonly used is obfuscation techniques. Anti-spam filters are quite effective at filtering out spam (it is not uncommon to have tens or even hundreds of spam messages accumulate in a bulk email folder in a single day). In response, phishers and spammers have developed techniques to avoid detection:

- Using images instead of plain text so that text classification filters will not detect the content of the message
- Including nonsense text strings in messages to throw off statistical classifiers that look for common patterns in phishing and spam messages
- Varying word patterns to avoid detection by statistical classifiers

Of course, the ultimate goal is to trick the reader, so phishers' primary tactic is to deceive the reader once the email has landed in that person's inbox. There are several techniques for doing so:

- Trying to raise fear in the reader by claiming their account information may have been compromised and needs to be updated.
- Using an image to hide the real URL of the spoofed site. When a user clicks on what appears to be a legitimate site, the user is taken to a bogus site for harvesting personal information.
- Masquerading as a social networking site, such as Linked In or Facebook, and asking readers to click a link to confirm information. These are not financial services sites, so readers may be less suspicious of phishing messages.

The final deception comes when the user has been lured to an illegitimate Web site. These typically look similar to the real sites they purport to be. At this point, the phishing victim may enter personally identifying information in a form. Other common attacks are to download Trojan horses or spyware to the victims' computers.

Phishing Trojan Horses

Trojan horses are programs that appear to be one thing, such as a screen saver, but are actually malicious programs. Trojans may be used to install botnet components, spyware, or other software that benefits the attacker. Again, the user is deceived; however, unlike with other forms of deception, phishing Trojan horses don't necessitate that the user do anything other than open an email or attachment and then suffer the consequences.

Spyware Attacks

Spyware is software that monitors user activities and collects information. Keyloggers, for example, intercept messages at the operating system (OS) level and collect copies of keys typed. These are saved in files and sent to a centralized server where the attacker can harvest them and scan for personal information.


Another kind of spyware is the video-frame grabber. These programs make copies of the contents of the screen at a set time interval. This type of attack is more useful than keyloggers because it captures displayed as well as typed information; however, video images are more difficult to analyze than streams of typed text.

Don't Be a Phishing Victim

The combination of deceptive messages, Trojan horses, and spyware is proving to be an effective and lucrative basis for phishing techniques. There are, however, a number of things you can do to reduce the risks of succumbing to a phishing attack.

The best way to avoid phishing attacks is to use a number of countermeasures. The following list highlights the most effective techniques:

- Use antivirus software to scan email. These programs will detect worms, Trojan horses, spyware, and other malicious software.
- Run anti-spyware checks on your computer regularly in case a piece of spyware made it past the antivirus scans.
- Use email filtering software. Most large email services provide email filtering software as do email clients such as Mozilla Thunderbird and Qualcomm's Eudora.
- Use anti-phishing features of the latest versions of Mozilla Firefox and Microsoft IE.

 Beware, a recent study shows that most users ignore the visual cues provided by browsers and legitimate Web sites; don't fall victim to this problem. See [The Emperors New Security Indicators](#) for more information about this pitfall.

- Do not click directly on links embedded in emails; cut and paste them or re-type them into the browser. Doing so can reduce the chance of visiting a Web site with a hidden URL embedded in the message.
- Use a personal firewall and block unused ports. This technique can stop some malware-generated traffic but is not effective when attackers use HTTP to transmit information.
- Validate senders of email messages.
- Familiarize yourself with phishing techniques and attack methods. See McAfee's white paper [Understanding Phishing and Pharming](#), the [Anti-Phishing Working Group's Resources](#), and the [MillerSmiles.co.uk Phishing Archive](#) for examples of phishing messages.
- Understand the email policy of the business you work with. Does your bank ever request account information via email? Does your ISP ever prompt you to update your records by sending an email? Many legitimate businesses are avoiding email-initiated exchanges because of phishing.

Phishing depends on deception and frequently employs technical means to collect personal information. The problem continues in spite of many efforts to control it. Understanding basic techniques to avoid becoming a victim is one of the best ways to reduce the financial incentives of phishing.

Article 7: Measuring Security: Application Metrics


Applications, especially Web-based ones, can provide entry points to your network for attackers. Keeping applications secure is a multi-faceted challenge, and one aspect that can easily be overlooked is security metrics. A commonly heard phrase is that you can't manage what you can't measure. This applies as well to application security as to other areas.

The purpose of this article is to describe security metrics that can be easily integrated into existing application development life cycles and maintenance models. Although there are more comprehensive security modeling techniques, the focus here is on ease of implementation and getting the greatest benefit while minimizing costs.

Application security metrics can be roughly divided into three categories:

- Code reviews
- Vulnerability scanning
- Runtime metrics

Together these areas provide a broad overview of application security, offer opportunities to define metrics that sufficiently measure application security, and allow designers, developers, and application administrators to identify and correct many common vulnerabilities.

 The proposed metrics presented are based on the valuable work done by application security specialists. Especially noteworthy are [A Metrics Framework to Drive Application Security Improvement](#) by Elizabeth Nichols and Gunnar Peterson; [Information Security: Why the Future Belongs to the Quants](#) by Daniel Geer, Jr., Kevin Soo Hoo, and Andrew Jaquith; and the [Open Web Application Security Project's Top Ten Vulnerabilities](#).

Code Reviews

Code reviews are a common practice in software development but they often focus on adherence to coding standards, such as naming conventions, error handling, and logging practices. Reviews that pay particular attention to code quality are especially useful for improving security. For example, a simple check on array reference boundaries or validation of input can avoid a potential vulnerability.

With regard to code reviews and security, several metrics can be used:

- Number of times input is not validated
- Number of times array references are not validated
- Number of procedures without error trapping
- Number of transaction types logged
- Number of privileges required to perform a database transaction
- Number of accounts with elevated privileges

These metrics can identify vulnerable areas of code, developers with poor coding habits, and problems with over-privileged accounts.

Code reviews should be incorporated in the software development life cycle. Rather than waiting until complete systems are ready for final quality control, security code reviews should be done iteratively during software development. This has two benefits. First, by finding errors early, there is less chance that the same error will be repeated later in another module. Second, reviewers will have less code to review at once and may be able to delve more deeply into the logic of the code. Once code is written and sufficiently complete for integrated testing, vulnerability testing techniques may be used.

Vulnerability Scanning

Vulnerability scanning uses tools to automate the processes of finding vulnerabilities in applications. Vulnerability scanners have been widely used to detect potential security problems in networks and device configurations but they can also be used for applications. They can be used, for example, to detect

- Missing patches on server operating systems (OSs)
- Misconfigured servers
- Missing patches in databases, application servers, and other software components

Application vulnerability scanners target specific vulnerabilities in Web applications. Corresponding metrics include:

- Number of vulnerable scripts
- Number of vulnerable Web services
- Number of paths to access administration interface
- Number of accounts compromised with password crackers
- Number of vulnerabilities found with database vulnerability scanner

The final set of metrics comes into play by monitoring deployed applications.

Runtime Metrics

The goal of collecting runtime metrics is to detect indications of vulnerabilities missed in previous code reviews and vulnerability scans. These metrics include:

- Number of times broken authentication is detected
- Number of unused accounts for some period of time (for example, 30, 60, and 90 days)
- Number of errors logged
- Time required to patch a vulnerability

In the case of broken authentication, one can correlate user authentication events in the application with database or Web service access. When a database query or transaction is executed without a corresponding record of a legitimate authentication to the application, one can reasonably assume the authentication mechanism is broken.

Unused accounts may become targets of attack by hackers with username generators and password crackers. Such accounts should be disabled or deleted completely.

The number of errors logged is a direct measure of software quality, and in some cases, usability. Ideally, the number of errors would decrease over time as the software is improved and users become more familiar with the operational aspects of the application. The time required to patch a vulnerability is essentially the time the application, and related data sources and services, are exposed to attack through that vulnerability.

Summary

Application security can be measured during development, testing, and deployment. The metrics described in this article are not comprehensive but they do cover a wide range of security issues and when incorporated into development methodologies and management procedures, can provide the raw data needed to understand where vulnerabilities lie.

Article 8: Five Things to Know about SQL Injection Attacks

Databases are the “crown jewels” of many companies as well as obvious targets for attack. One of the methods used to get at a database is through an application interface, especially one available on the Web. Unless you develop your applications carefully, you can put databases at risk of data loss or even data corruption through SQL injection attacks.

SQL injection attacks are a form of non-validated input vulnerability targeted to database applications. The goal of such attacks is to access or change information in a database using a weakness in a database application.



Note that it is the application—written in Java, PHP, and so on—that is the problem, not the Oracle, SQL Server, MySQL, etc. database. (Other kinds of vulnerabilities can be found in database servers).

The following list highlights five things to know about SQL injection attacks as you develop database applications:

- Attackers can quickly determine whether an application is vulnerable.
- Probes can reveal a great deal of information about the database server, the data model, and the application.
- SQL injection attacks can include both data manipulations, such as querying for extra data, and structural manipulations, such as dropping a table.
- The use of bound parameters and stored procedures to manipulate data significantly reduces the risk of a successful SQL injection attack.
- Minimize privileges to any database account accessible to users to reduce the damage that can be done if a SQL injection attack occurs.

Here are the details on each.


Potential SQL Injection Vulnerabilities

SQL, the query language used by most relational databases, makes it easy for attackers to determine whether an application is vulnerable to SQL injection attacks. Consider a simple Web form prompting a user for a username and password. Behind the scenes, there could be a statement such as:

```
SELECT
    Full_name
FROM
    User_info
WHERE
    username = $user
AND
    password = $password
```

where \$user and \$password are variables holding the text typed by the user. The developer intends to get the user's name for display purposes and validate that the person logging in is a registered user.


The problem here is that anything the user types will be passed directly to a database operation without any checks. If someone were to enter an invalid punctuation character, such as a quotation mark, the database would generate an error that would propagate through the application and eventually display information to the user. The attacker would then know that invalid characters are not removed from the input string and chances are that there is little or no validation of input. Therefore, the database is potentially vulnerable to a SQL injection attack.

 Avoid this problem by validating all input. Accept only characters that make sense for the information sought. Passwords and usernames are limited to alphanumeric characters and a small number of punctuation marks; make sure nothing else is passed through to the SQL processor. Semi-colons are used as command delimiters in some databases, so any text following a semicolon should be discarded.

Probes Reveal Information About the Database

Let's assume there is no validation on input strings and attackers start to append commands to the end of the SQL statement in the application. For example, if the application is written so that multiple SQL statements can be executed, the attacker could input strings such as `'; SHOW TABLES'` or `'; SELECT table_name FROM user_tables'` to see a list of tables in MySQL and Oracle, respectively.


It might take some time for the attacker to find the right combination of legitimate and malicious input to get the sought-after information, but once the right syntax is determined, a great deal of information can be discovered. If the database connection depends on a user account with access to the data dictionary, the attackers can get a list of tables, columns in tables, referential integrity constraints, statistics on volumes of data, and other details that would make further attacks possible.

 Avoid this problem by validating all user input as described earlier.

SQL Injection Attacks Can Manipulate Data and Database Structures

If an attacker can execute arbitrary commands by appending them to application SQL statements, data can be queried and data structures manipulated. A simple example that allows more information than intended to be returned is adding an OR clause such as `'OR 1 = 1'` to a SQL statement. Doing so causes the WHERE clause to evaluate to true in all cases returning all data for all rows in the compromised table or view.

Data definition commands could also be executed in some cases. These could include dropping tables or constraints between tables, adding columns to tables, or changing indexes. The consequences range from decreased performance to the loss of functionality.

 Avoid this problem by validating input and limiting privileges to data definition commands.


Use Bound Parameters and Stored Procedures to Query and Manipulate Data

A common vulnerability in the examples used so far is that the application is building a string that is passed to the SQL processor as a single unit. The entire string is assumed to be a command or set of commands. Instead of coding at the level of strings, it is better to code at the level of statements.

Take the username and password example. Rather than concatenate SQL commands to user input into a single string, a better model is to construct a prepared statement that is preprocessed by the SQL optimizer. This step takes the legitimate SQL and builds an internal representation that the database server uses to fetch or manipulate the data. Place holders, known as bind variables, are used in the statements to store the input variables, which are passed like parameters.


The difference is that the parameters are only evaluated in the context of the bind variable. Thus, if extra text is added after a reasonable value (for example, “pass1w0rd; SHOW TABLES”), the full text is treated as the value; it is not parsed as a command to be executed by the SQL command processor.

Even better is to encapsulate these statements in stored procedures. Doing so helps with maintainability. If underlying table structures change, only the procedures need to be updated; there is no need to hunt through code looking for multiple uses of the table. In addition, privileges may be restricted to the stored procedures. There is no need to grant a user insert, update, delete, or select privileges to tables. Grant those privileges to the stored procedures and grant the privilege to run the procedure to the user. In this way, you trust the procedures to manipulate your data but only trust your users to run trusted procedures.

 Bound parameters and preprocessed SQL statements and stored procedures can reduce the risk of vulnerabilities related to SQL syntax and minimize the number of agents trusted to manipulate the database.

Minimize Privileges

If an attacker does manage to successfully construct a SQL injection command, the command will do the attacker no good if the account has insufficient privileges to execute the command. Users without select privilege to the data dictionary cannot see the data structure of the application. Without proper resource privileges, attackers cannot create new tables, drop existing ones, retrieve source code of stored procedures, or edit and recompile stored procedures.

 Limit privileges to any account used by a database application. Ideally, reduce privileges to the bare minimum and add back only those that are definitely needed for the application to function.

Summary

The risk of successful SQL injection attacks can be significantly reduced by simply validating input, using bound parameters and stored procedures, and limiting privileges of database accounts.

Article 9: Four Things to Know to Protect Your Network from Unmanaged Devices

Network services are commonly accessible well beyond the firewalls that used to demarcate internal networks from the Internet. Customers, business partners, remote employees, and others are using Web applications to review their accounts, enter orders, check calendars and emails, and collaborate with colleagues. Sometimes remote users are working with a company-issued laptop running the latest antivirus software and up-to-date operating system (OS) patches and the system has been expertly configured to reduce the risks of known vulnerabilities. These devices are in your control; they are one of your many managed devices. However, in today's world, we do not have the luxury of working just with devices we can control and secure.

It is important to know four basic things about unmanaged devices and how to reduce the risk inherent in their use on your network:

- What are unmanaged devices?
- What security threats do unmanaged devices pose?
- What functions are required in a network access control (NAC) system to protect against those threats?
- What are the limits of NAC protections against unmanaged devices?

What Are Unmanaged Devices?

Unmanaged devices are hardware clients that have access to your network but are not subject to your security policies and procedures. Some prototypical scenarios involving unmanaged devices include:

- The employees checking in from home are using the same computers their teenagers use to download music and, unknown to these employees, these computers host peer-to-peer file-sharing systems, malware, Trojan horses and botnet agents.
- Road warriors working for business partners checking in from coffee shops around the country using weak or no encryption on their wireless device.
- A busy executive quickly downloading a spreadsheet with sales projections to a kiosk in a hotel business center before rushing off to catch a plane with no time to make sure copies are not accidentally left behind.

The bottom line is that unmanaged devices may be as secure as your managed devices or they may be malware-infested, poorly configured sieves through which valuable information is lost or conduits for gaining access to your IT infrastructure.

What Security Threats Do Unmanaged Devices Pose?

Unmanaged devices pose a number of potential threats, including the spread of malware and information loss. An unmanaged device that is not properly protected with antivirus software can house viruses, worms, Trojan horses, and botnets that can potentially spread to managed devices through access to a company's network.

Perhaps of greater concern is the potential loss of information. Spyware, such as keyloggers and video frame grabbers, can intercept usernames, passwords, and images of confidential documents. Unsuspecting users may not clear browser caches and might unintentionally leave information available to the next user who decides to use the browser's back button. The low-tech printer is also a problem. Consider how many times a document has been left by a printer because someone forgets to pick it up or has been left in a print queue because of a problem with the printer.

What Functions Are Required in a NAC System to Protect Against Those Threats?

To protect against the spread of malware or the loss of information, several basic functions are required:

- Encrypting data sent to the unmanaged device
- Disabling spyware during session initialization
- Blocking access to peripherals, especially printers
- Clearing browser caches during session termination
- Scanning traffic between the unmanaged device and the network for malware
- Blocking split tunneling (that is, sending information from the secured network connection over a second network connection on the unmanaged device)
- Providing functionality on demand so that browser-based, downloadable agents are required

In addition, network administrators may establish a policy on the minimal configuration of an unmanaged device with regards to security. For example, the policy might dictate that any unmanaged device must have antivirus software and a personal firewall running.

What Are the Limits of NAC Protections Against Unmanaged Devices?

Unmanaged device protections have to work around several limitations. For starters, functionality must be provided on demand; no software may be permanently installed on unmanaged devices. The modules must also work without requiring administrative privileges.

Similarly, on-demand security for unmanaged devices must leave the client in the same state it was found. Agent software and any configuration changes must be rolled back when the session is terminated. On-demand security cannot change the long-term security posture of an unmanaged device, so there may be a point where the only secure option is to block some functionality, such as printing.

The growing use of unmanaged devices to access company networks has created the need for on-demand security within NACs. Existing on-demand technologies can significantly reduce the threat of malware and information loss from such devices.

Article 10: Security and Service-Oriented Architectures

A popular approach to Web software development is to assemble complex functions and workflows from a set of smaller, focused services. For example, rather than build an order-processing system from scratch, a vendor may assemble several core services from an inventory management system, a billing system, a shipping system, and a Web portal for managing the presentation layer of the application. When orders are processed, information is passed back and forth among the several systems. This kind of system design is becoming more appealing to architects and project sponsors because it reuses functionality, increases consistency among operations, and reduces maintenance. It also, however, introduces additional security considerations.

This article will examine some of the factors you will want to keep in mind as you develop and deploy service-oriented architectures (SOAs), including:

- New entry points into applications
- Limits of network-based security measures
- Need for encryption
- Challenges with access controls

Fortunately, the security concerns related to these topics have been addressed by Web services standards such as WS-Security. Pointers to additional resources are provided throughout the article.

New Entry Points into Applications

Access to enterprise applications has changed radically over the years, from dumb terminals hardwired to mainframes to software interfaces accessible to anyone with an Internet connection. In the past, physical security could contribute to logical security; if you couldn't get to a terminal, you couldn't get to the application. We don't have that benefit in SOAs.

Take the example of Google maps (<http://www.google.com/enterprise/maps/>). Google provides an application programming interface (API) that allows developers to incorporate maps, driving directions, and traffic overlays in applications. The API is essentially an entry point into Google's mapping application and database. Anyone with a freely provided API key can start using the service. No one needs to know where Google data centers are located, have physical access to Google's internal network, or even run proprietary software. How does this kind of design affect an enterprise's application security?

When developing Web services, you should include in design considerations:

- What functions will be publicly accessible?—Information about Web services can be listed in directory services such a Universal Description, Discovery, and Integration (UDDI) registry. These registries can be queried using a standard protocol.

 See <http://www.oasis-open.org/committees/uddi-spec/doc/bps.htm> for best practices on using UDDI.


- How will Web services authenticate themselves to underlying services?—For example, a Web service may need to look up information in a database. What will the service be allowed to query? Will it return information if provided a valid customer ID? What if an attacker probes with a list of generated IDs, how will the Web service respond?
- What auditing is required for the Web service?—Will every invocation be logged or will only certain events, such as invalid authentication credentials, be logged?

It is also important to understand that some network security measures that have worked well with client/server applications do not apply to Web services.

Limits of Network Security Measures


Firewall ports have long been used to control access to internal networks. If a database server communicated with clients over a particular port, that port could be opened on the firewall. Traffic could be monitored knowing that any packets coming in through that port were destined for the database server. Web services communication typically uses HTTP and so comes in over the same port as Web server traffic, thus, it cannot be blocked at the lower levels of the network stack.

Application firewalls are proxies that analyze application-level, as opposed to packet-level, traffic. These firewalls are configured with rules based on application communications (for example, a Web service call). They provide protection against attacks based on cookie manipulation, brute force attacks, session attacks, manipulation of request flow attacks, and others.

 For more information about application firewalls, see the at Web Application Security Consortium's application firewall evaluation criteria at <http://www.webappsec.org/projects/wafec/>.

Need for Encryption


Sensitive data exchanged between Web services should be encrypted. This is especially the case if you allow business partners, customers, and others outside of your organization's security perimeter to access internal Web services. If sensitive information is sent outside the internal network, the only way to ensure it is not copied or tampered with is to encrypt it.

 The WS-Security standard includes encryption; the IBM Developerworks Forms is a good source of information on this topic, especially <http://www.ibm.com/developerworks/library/specification/ws-secure/>.

Challenges with Access Controls

One of the most challenging aspects of SOAs is controlling access to resources and information based on a user's identity. Web services are proxies acting on behalf of a user, but how can the Web service verify the identity of the user? In the case of public services, such as Google's map service, it doesn't matter (much) who makes the call. Retrieving medical records or financial information is a different story.

Even if you have a good handle on identity management and access controls within your organization, you should consider how business partners may use your services. Do you trust the authentication and authorization systems used by business partners? If your business partner A trusts one of its business partners, B, do you trust B's users? If you trust your business partner enough to use their authentication system, how do you exchange information about users and privileges and keep it up to date? These are the kinds of problems one encounters in federated identity management.

 The Liberty Alliance (<http://www.projectliberty.org/>) is one organization that has addressed the problem of standards in this area. The Security Assertions Markup Language (SAML) and related standards from OASIS (http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security) also address problems in identity management.

SOAs offer several advantages from both business and technical perspectives. Making the move to SOAs requires a number of changes, including addressing new sets of security challenges. Fortunately, fundamental components are in place for dealing with these.

Article 11: Multiple Layers of Database Security

Databases are prime targets for attackers. Whether the attackers are searching for identity information, proprietary project details, product design information, or company financial data, there is a good chance they will find it in a database. Obviously, you want to secure databases as much as possible, but what is less obvious is how many layers of database security are required to maintain even the most basic levels of security.

This article will examine the major aspects of database security including:

- Network and operating system (OS) security specific to database servers
- Protecting databases themselves
- Potential vulnerabilities in applications that use databases
- Encrypting data

Each of these aspects entails potential vulnerabilities that can be leveraged by attackers to compromise the confidentiality and integrity of data in a database.

Before delving into specific security issues, let's just list some of the major operations of a modern database system. First, we have a listener. This is a service that listens for and responds to requests on a network port. It is essentially the doorway into the database. The listener passes off requests to a query processor that actually determines what data or operation is requested and translates the query into low-level commands that can be processed (for example, look up a record or update a row). Privileges are verified to ensure that the user making the request has been granted permission to carry out the requested operation. The actual execution of the operation either retrieves data from or updates data in the database. The database itself depends on OS services, such as the file system, to store the lowest levels of data and data structures.



Actually this is a simplification. Oracle, for example, supports a method for using "raw" disk storage that can bypass the usual file system.

If all is copasetic, the operation is performed and results are returned to the caller. So the question is, Where can things go wrong from a security perspective?

Network and OS Security Specific to Database Servers

From a network administrator's perspective, a database is just another service running on one of many servers. This is yet another simplification that glosses over many details, but it is a useful view from a security perspective because it helps focus on the network-oriented and OS-specific vulnerabilities of a database. Two of the most important relate to the listener and the OS.

Database Listeners

As noted, database listeners are the gateway to the database. Several steps should be taken to minimize the chances that someone could compromise the database by exploiting the listener:

- Change the default port of the listener—It's just too easy for attackers to probe for a response from the default port
- Make sure the listener password, if one is used, is set to a strong password
- Patch the listener—Attackers use *fuzzers* (programs designed to probe for vulnerabilities) to test listeners and detect unauthorized means of gaining access to the database
- If possible, restrict access to a limited set of servers (for example, an application server) that have legitimate access to the database; IP addresses can be used to verify the source of queries
- Enable security policies that restrict access to the database listener

Although the listener is like the doorway to the database, the OS is the surrounding environment and it too should be used to protect the database.

OS Access

Databases depend on OS services, especially on file system management. Data is typically stored in files. These files and their parent directories should have access controls in place to prevent tampering. The following list highlights things to check at the OS level:

- Access controls on all database files and directories
- Logging of changes to control files
- The number of users with root or administrative access to the server
- Whether the OS patches are up to date

Some databases provide a means to link multiple databases, known as database federation, and this too needs to be considered when planning database security.

Who Can You Trust: Database Federation

Many of us are used to sharing network drives. We can store files on a network server and access them from our PC as if they were drives right on our machine. The same general concept holds among some databases that allow access to one database from another. This is especially useful when two databases have slightly overlapping domains.

For example, an insurance company may have one database for managing policies and claims and another for customer marketing. Occasionally, it may be useful for a marketing application to have access to claims history. The application could simply connect to both databases and pull information from both, but if the data from the two systems needs to be merged, it's far more efficient to have the databases do it. (After all, that is what they are designed to do). This combination of multiple databases for a single operation requires database federation.


Federation entails trust between two databases and their users. When database federation is used, one should carefully establish policies governing:

- What operations are allowed from the federated database (for example, read only, read and write)
- How users are authenticated from remote databases
- What data is made accessible to federated queries


Also consider chains of trust. For example, if Database A allows Database B to query its data, and Database B allows Database C to query its data, should Database C have access to Database A's data? Depending on how access controls on federated tables and views are structured, you could run into an unintentional trust relationship.

Potential Vulnerabilities in Applications that Use Databases

Applications that use database are like database listeners: they are doorways to data. SQL injection attacks are a well-known threat to databases. In these attacks, data entry features are compromised by specially designed data that take advantage of poor application design to gain unintended access to databases.

 Key aspects of SQL injection attacks have been described in an earlier article "5 Things to Know About SQL Injection Attacks" available at the Realtime Messaging and Web Security Community digital library at http://www.realtime-websecurity.com/digital_library.asp.

To minimize the risk of a compromised application or user account, access controls should be used to restrict access to tables, views, stored procedures, metadata, and other database information. All enterprise-level relational database applications provide basic, course-grained access controls. Oracle databases also support fine-grained access controls that limit access not just to tables and views but to specific rows of data using a functionality known as virtual private databases. These access control features are essential to ensuring the confidentiality and integrity of data.

 Also assess the level of exposure through Web services that may access databases. For more on this topic see the article entitled "Securing Service Oriented Architectures" at http://www.realtime-websecurity.com/digital_library.asp.

Encrypting Data

If the database listener is secured, the OS is used to protect database files, databases are federated according to established policy, the potential for SQL injection attacks has been addressed, and access controls within the database are in place, there is still the potential for a data breach. At some point, data will have to leave the confines of the database server. It may be as a backup tape or as an export file to another database. Once data is written to a device that is outside the control of the database management system, all the access controls in that system are useless. To ensure that data outside of a database is protected, one must encrypt the data before exporting it. Encryption is essentially the only means to protect data once it is written to a device outside of your control.

Databases are essential to managing vast volumes of data in today's enterprise. Keeping the data secure requires attention to multiple levels, from the network and OS to the database management system and database applications.

Article 12: Integrating Security Functions: What Makes Sense?

It can sometimes seem that the constantly evolving nature of information security leads to the creation of new countermeasures, all of which need specialized planning, deployment, monitoring, and management. The list of common network, server, and client device countermeasures includes:

- Firewalls
- Intrusion detection/prevention
- Content filters
- Antivirus/anti-spyware
- Network access controls
- Policy enforcement
- Identity management and access controls

Add to this the list of newer security devices, such as application firewalls, and you can see the appeal of security applications that integrate multiple functions. But does the drive for a more controlled management environment outweigh any disadvantages of consolidating security functions?

To answer that question, consider three aspects of integrated security applications and how they fit in your business:

- Functional similarity of countermeasures
- Performance implications
- Management and organizational issues

Functional Similarities

Anyone can throw different ingredients together and call it a recipe, but is the product edible? That is the first kind of question you should ask with respect to security integration. Many threats are similar enough that it makes sense to deploy a single, comprehensive method for dealing with them. Take malware and spyware, for example.

When we talk about antivirus today, we are really talking about antivirus, anti-worm, anti-Trojan horse, anti-botnet, and anti-rootkit protection. Security professionals distinguish these programs by their methods of propagation and their function; however, the way we detect them is similar. When scanning an email attachment, there is no need to have separate systems scan the same content, one for a virus that might carry a payload that deletes files and a piece of spyware that installs a keylogger on an infected device.


When countermeasures use similar technologies and examine the same data streams, integration is logical. Even when the technologies vary, it is often practical to combine operations into a single application; this is the case even in antivirus detection, where signature-based and behavior-based analysis are used together. From the perspective of a network manager, fewer devices means fewer assets to manage—consolidation is a good thing. There is a point, though, when the disadvantages of combining functions into a single application become more pronounced.

Performance

Security appliances have to scale to the volume of traffic on the network and to the level of analysis dictated by policies. In small and even some mid-sized businesses, network traffic may be light enough that a single appliance can handle anti-malware, content filtering, URL filtering, and firewall services. Large organizations will not consolidate all those functions into a single appliance; in fact, they will likely require multiple appliances each running a dedicated function distributed throughout the network.

Consider current and anticipated traffic volumes when considering consolidated security applications. You need to keep in mind the increasing popularity of Web-based applications. Rich client-side applications can offload some processing from centralized servers and reduce network traffic, but other disadvantages, especially compatibility and restricted administration privileges on client devices, limit the appeal of this approach.

Rich Internet applications are getting closer to client/server levels of functionality and with this can come increased Web traffic. AJAX, a popular application framework, reduces the traffic between browsers and Web servers relative to traditional HTML programming. This benefit may be offset by the increasing complexity of Web applications that use mashups to combine content from multiple sites, bringing even more text, images, and multimedia to the client.

 One of the easiest-to-use mashup builders is iGoogle (<http://www.google.com/ig>). In minutes, you can build Web pages pulling in rich content from a variety of sources using Google gadgets.

Even when a solution makes technical sense, it may not fit well with the way an organization works.

Management and Organizational Issues

How a business structures its IT department can influence how well a consolidated security appliance will fit in. For example, if a network infrastructure group is responsible for firewalls, routers, and intrusion prevention systems (IPSs), they would be a logical choice for managing anti-malware scanning. If another group is responsible for ensuring compliance with approved-use policies for email, Web, and other Internet services, they would be the logical choice to manage content-filtering and URL filtering functions. Introducing an appliance that accomplishes both functions would raise a number of organizational problems:

- Who will pay for the appliance if both groups have independent budgets?
- Who is responsible for monitoring hardware and applying common patches?
- Who configures and monitors shared functions?
- Who establishes policies?

None of these questions are so complex that they could not be worked out, but in some organizations, especially large ones, separate appliances might make sense.

Summary

Consolidating security functions makes a lot of sense. You can take advantage of high-performance hardware to enable several countermeasures on a single appliance. Vendors provide comprehensive reporting and monitoring systems so that management is easier than if the operations were spread over several devices. However, the advantages need to be balanced with concerns about performance and organizational boundaries. The question you should ask is not “Does integrating security functions makes sense?” but rather “What is the optimal integration of security functions for our organization?”

Article 13: The Rise of PDF Spam and How to Stop It

It seems that just several months ago image spam was increasing rapidly. That was understandable, given how well spam blockers were working with text-based spam. Image spam was an attempt to avoid detection using existing techniques and it lasted for a while, but now image spam can be effectively blocked. Spammers have moved on to their next method *de jour*, PDF spam.

In addition to being another example of the ongoing cat-and-mouse pattern prevalent in information security, the rapid rise and fall of different spamming techniques tells us something about the kinds of anti-spam techniques we should be building. Quick fix responses, like blocking images in email messages, will work for some time, but they are relatively brittle—they break easily. What we should be aiming for are more robust methods that degrade gracefully; that is, they may not work 100% of the time but are capable enough to withstand simple avoidance techniques.

This article will examine different methods for delivering spam and ways to counter them. The article will then discuss the fundamental issue with spam blocking, content analysis, and see why when we veer from addressing fundamentals for a quick fix solution, we end up with brittle, easily avoided detection techniques.

Spamming Methods

Spammers are concerned with getting their words before a reader. How the messages are delivered and packaged is irrelevant, so long as they avoid detection. The history of spam has seen several ways of packaging messages, including:

- Plain text
- Image files
- Document files
- PDF files

These different packaging methods have advantages and disadvantages for both spammers and spam filter developers.

Controlling Plain Text Spam

When spammers use plain text, they have to work to slip by spam blockers. When an email message was received, a spam blocker could analyze the text looking for patterns commonly seen in known spam and apply a few heuristic rules to check for other clues that a message may be spam. The result was a very high-quality classification scheme that worked well to block spam.

Today, crafting good spam blockers takes some work, but we now understand how to effectively use pattern recognition techniques to identify spam. Spammers knew this and responded with obfuscation techniques, like adding garbage text at the end of the message to throw off the detection algorithms. That worked for a while and then spam blockers adapted. This is a good example of graceful degradation. The spam blockers examined many characteristics of spam to classify messages so that changes in one or two of these characteristics may have tricked them in some cases, but not all.

Image Spam Sneaks by, for a While

Image files became spammers' best friends. These files could get the message in front of a reader without risking detection by text-based classification methods. The developers of spam blockers responded by incorporating optical character recognition (OCR) techniques, which would analyze the image, map the contents to text, and then pass that text to the classification program.

OCR techniques are not nearly as good as humans at recognizing characters, though, so spammers could obscure their messages just enough to throw off the OCR program while leaving a message recognizable to humans. (The same techniques are used to prevent bulk registrations at Web sites when users are asked to type in a set of characters that are displayed in an obscured way somewhere on the page).

Image files are easily identified and email clients can be configured to block all images or at least not display them. The volume of image spam has dropped significantly over the past several months and appears to have been replaced with spam packaged in documents, especially PDF files.

PDF and Other Document Spam

Spammers seek the path of least resistance. They know plain text won't make it past filters and images are being blocked wholesale. They also know that legitimate documents are frequently attached to messages, so there can be no bulk blocking of those. Combining these realities with the efficiency of creating PDF files makes that file format an ideal package for getting spam through. Judging from examples, spammers have not gone out of their way to improve the quality of their spam, they are just putting it into a PDF format.

The long-term solution to the problem of constantly shifting packaging methods is to focus on content, not file formats. This, however, has significant implications for the time and computing resources required to detect spam. At a minimum:

- Email plain-text content is analyzed by spam filters
- Text is extracted from PDF, Word, and other types of attachments and analyzed by spam filters
- Images are either wholesale blocked or OCR techniques are used to extract text

This kind of analysis could slow email delivery if sufficient resources are not in place. In spite of the drawbacks, content filtering, not packaging-based detection, is the best way to combat spam. The technologies exist to do this today and are used to protect against another problem, the loss of intellectual property. Although businesses and other organizations are working to keep spam out, they also need to work to keep their intellectual property and confidential information in. The same technologies will work well with both problems.

Article 14: Elements of Effective Event Log Monitoring

Keeping track of events on an organization's network is challenging: there are many kinds of hardware and software, configurations vary among clients and servers, the state of infrastructure is in flux as assets are rolled out and retired, and user access control requirements seem to be in a perpetual state of flux. Yet all these different types of events can have implications for information security. Consider the following examples:

- A wireless device has just connected to the network. Is it a registered device, a mistaken attempt by an unauthorized consultant to use the network, or a rogue device used by an attacker?
- A new set of antivirus updates has been pushed out to clients. Were all updates successful?
- A server that is usually on at all times is no longer on the network. Is this an expected outage, a hardware failure, or another kind of event?
- A new user account was added to a Web server and then added to a privileged group. Has this been approved?

The breadth and frequency of events in even small networks can generate more data than one could reasonably expect to analyze manually. Systems administrators and network managers have two options: they can either automate some parts of the collection and analysis or limit the amount of data that is analyzed.

Why not limit the amount of information analyzed? After all, monitoring programs are notorious for generating huge volumes of data. Yes, one could reasonably not record information about successful login attempts or successful file access operations. Even with the exclusion of events that should be tracked only in the most secure environments, the volume of data generated is still significant. Not monitoring some devices may be a reasonable method in some cases, but in general, it is more prudent to consider any device as an entry point for an attack and therefore the device should be monitored.

Assuming that event monitoring is generating more data than can be manually analyzed, what should one look for in tools for managing events? There are three factors to consider; event monitoring should be

- Consolidated
- Comprehensive with respect to collecting data from devices and applications
- Targeted with respect to reporting and analysis

At first, the second and third items may seem contradictory, but they are not. We want as much useful data as we can collect, but once we have it, we want to organize and filter it effectively so that we can find information relevant to maintaining operations and detecting security problems.

Consolidated Management and Reporting

Networks and applications are highly distributed and this, ironically, increases the need for consolidated reporting for a number of reasons. First, there is the problem of heterogeneous operating systems (OSs). Networks may have hundreds of thousands of Windows clients running Windows XP and Vista in various versions. The same network can have Windows, Linux, and UNIX servers often with differing versions and distributions. Analyzing data from a Windows event file is different from analyzing the same kinds of data from syslog on a UNIX system. Consolidation can help mask some of the differences and provide a single point of access to information from an array of devices.

Second, a single event can trigger the logging of multiple event messages. For example, an application may log an event that an operation failed. At nearly the same time, the database may create an event indicating a data table could not be extended to accommodate more data. The OS on the database server may also log information about low disk space. Having all this information accessible together can help reduce the time required to diagnose and repair these problems.

Consolidated reporting also facilitates the creation and use of template reports and analysis tools. For example, an event log monitor that collects data from all database servers could use the same compliance template to report on all servers.

Comprehensive Monitoring and Reporting

Efficient reporting and event log management is also comprehensive. Any device that must be managed or could be used as an entry point in an attack should have its event logs monitored.

Managed devices are ones that connect to your network and are controlled by the organization. Unmanaged devices, in contrast, belong to customers, business partners, consultants, and others who have access to some IT resources within the organization. Monitoring the events on managed devices will have the greatest benefit for service desk staff and budget. Again, consolidated reporting can provide more information and help service desk staff diagnose and repair problems more quickly than would be possible without that information.

Monitoring is also important from a security perspective. Malware has grown in complexity and can include more than a single virus or worm. Some ensemble malware includes multiple payloads, multiple methods of compromising hosts, and multiple ways of spreading. A laptop may appear relatively unimportant compared with directory servers, database servers, and core applications, but it can also be an entry point of a multi-vector piece of malware that can spread well beyond the initial laptop infection. Monitoring can help to ensure that countermeasures are up to date and, if a system is compromised, detect the attack. At the very least, event logs may be of use in forensic analysis following an attack.

Targeted Analysis and Reporting

The final piece of effective event log monitoring is knowing what to look for. Many events in a network are mundane. The problem is knowing how to distinguish something mundane from something of interest. This depends in part on the context. For example, agencies collecting sensitive or private information may want to monitor anytime that sensitive data is accessed, even if it is not changed. In most cases, successfully accessing a file or database record is insignificant. Some events are likely to be of interest to most organizations. These include events such as:

- Failed login attempts to servers and applications
- Failed attempts to access files
- Failed attempts to perform operations within an application because of insufficient privilege
- Change of access control privileges on a user account

Others can be added to this list, depending on the specific uses of servers. It is especially important to monitor database activity. Databases house the “crown jewels” of many businesses and government agencies. They are likely targets, have many legitimate users, and as the number of Web applications and Web services grow, more and more databases have some exposure to extra-organizational attacks.

Summary

Event log management contributes to both infrastructure management and security. The most effective event management includes consolidating information in a single repository that supports a single point of access to log information, comprehensive data collection from all devices and important applications, and the ability to target and analyze the most relevant types of events across the full range of devices on a network.

Article 15: Ten Tips for Securing MySQL Databases

Relational databases are prime targets for attacks because the databases can house the ‘crown jewels’ of a company, agency, or other organization. MySQL, the popular open source database with well-established vendor support, is popular with small Web-based applications but has also developed into a platform for enterprise applications. Database designers and developers have to keep security in mind while creating applications that meet functional requirements. Making sure the database is designed and deployed following these 10 tips will help reduce the risks of known threats and vulnerabilities. 10 tips for securing MySQL databases:

1. Run mysqld as an ordinary user.
2. Use grant and revoke to limit access
3. Review granted privileges regularly
4. Use strong passwords
5. Validate all input
6. Limit FILE privilege to database administrators
7. Do not index confidential data
8. Patch the database and operating system (OS)
9. Limit access to the database server to trusted devices
10. Isolate and minimize code that interacts with the database

Run mysqld as an Ordinary User

The MySQL daemon, mysqld, should not be run as root. Mysqld has FILE privilege, which allows it to write and read files on the host to any directory accessible to the account running the daemon.

By running mysqld as an ordinary user, one effectively limits the potential directories to which the process can write. If the process is compromised, for example by using a buffer overflow attack, and the attacker is able to write a file to the host, at least the damage will be limited to a subset of files and directories. Had the root account been used, any directory and file on the host would be vulnerable.

Use Grant and Revoke to Limit Access

Privileges can be granted at several levels. Granting a privilege at the database level applies to all objects in a database; table level grants apply to all columns in a table; and column level grants apply to a single column in a table. Table and column level privileges can be used to finely tune who is granted access to particular pieces of data. These are especially useful when combined with data modeling techniques to limit operations that users are allowed to execute.

For example, when a database application must allow a number of different users to write to a table, such as an order table, rather than granting full insert and update privilege to the order table, users may be granted privileges to write to a pending order table which is then read and processed by a trusted procedure that writes transaction data to the order table. In this way, the functional requirement of allowing an arbitrary user to write data to the database does not expose other data to tampering.

Column level privileges are especially useful to limit access to sensitive data. The classic example is the salary column in an employee table. All employees may have access to read most data in the table but the salary column can be limited to just members of a manager group.

Review Granted Privileges Regularly

Database administrators should regularly review group membership and privileges granted to ensure no one has elevated privileges. Just as attackers who breach OS security may seek to create accounts with elevated privileges, the same principles hold with database security. A less malicious scenario is that employees or contractors may leave and their accounts are not removed. Regular reviews of users and granted privileges can help correct this.

Use Strong Passwords

Password crackers are readily available on the Internet. Sometimes they are presented as administrator tools for password recovery but regardless of how they are described, these programs can be used to perform dictionary attacks and use other techniques to recover weak passwords. Strong passwords are not words found in the dictionary or simple combinations of words and numbers.

Validate All Input

Never allow data input directly by a user to be passed to a dynamic SQL string. SQL injection attacks can be defeated by properly validating input. This includes verifying numeric parameters are actually numbers, that length of input is reasonable, and that strings are escaped to avoid the potential for changing the meaning of a dynamic SQL statement.

Limit FILE Privilege to Database Administrators

As noted earlier, the FILE privilege allows users to write files to the host file system. Generally, writing files should only be required when exporting data, which is usually done by database administrators. By limiting access to this privilege, if a database user account is compromised, there is a reduced risk of the attacker being able to write to the host file system.

Do Not Index Confidential Data

Recent discoveries by database security researchers indicate that it may be possible to recover confidential data from indexes used to improve performance. By writing data to a database and analyzing the resulting changes, attackers may be able to discover the underlying data used to create an index. This could expose confidential information, such as Social Security numbers and credit card numbers. Rather than index confidential data directly, calculate a hash value using MD5 or other hash function and store that value in a column that is indexed.

Patch the Database and OS

It is important to keep the database and OS patched. Vulnerabilities continue to be discovered and there is no reason to assume that any version of MySQL is “secure enough.”

To reduce risks from database and OS vulnerabilities, follow a patch management process that includes applying the patch to a test server that is as closely configured to the production server as possible. Run a series of regression tests on the test server to ensure that the patch did not break any functions within the database application.

Limit Access to the Database Server to Trusted Devices

Database servers should be protected behind DMZ and accessible only to trusted application servers. The database should not be directly accessible from the Internet. Connects to the database should be managed by a proxy application that performs preliminary input validation and limits the types of queries and commands sent to the database server.

Isolate and Minimize Code that Interacts with the Database

As much as possible, isolate application code that interacts with the database to a set of procedures that are called throughout the application. This allows for easier maintenance and can help improve security. Suppose, for example, that a vulnerability is found in a particular type of query and a workaround is proscribed that requires an extra validation check on the command. Rather than search through all application code looking for instances where the command is used, developers could just update the database procedures.

Summary

Databases, and in particular MySQL, are widely used in Web applications. Following a few simple rules about security, these applications can reduce the risks from a number of threats.

Article 16: Establishing Organizational Security

Introducing security awareness to organizations is challenging. Executives have a broad range of concerns competing for their attention, managers are focused on operational efficiency and the particular metrics on which they are measured, users have too little time and expertise to understand the technical details of information security threats. Add to that list the limited budgets and growing list of demands on IT departments, and you have a 30,000-foot view of what many security professionals face every day.

We know from experience that a pervasive awareness of security and its relation to operations is an important element of managing cyber security risks. How do you get there? Every organization is different, but several themes seem to recur when we study organizations that have had success in this area:

- A focus on risk, not just compliance and certifications
- A shared responsibility for security
- The use of composite metrics
- The use of realistic and enforceable metrics
- Introducing security initiatives with other organization- or department-level initiatives

Managing Risks

It is easy to react to security threats and attacks. IT news sources and industry blogs provide fresh information almost daily on new threats, vulnerabilities, and examples of breaches. Of course, you need to respond to emerging threats and address attacks as they occur. You should not, however, manage security as if that were the extent of the problem. Good managers do not foster customer loyalty by responding to complaints, which are signs that something has failed, but by working to minimize those complaints. Information security management functions similarly.

Information security practices are implemented to support business operations. The real goal is not to, for example, prevent a virus infection but to prevent disruption to operations. This becomes clear when you ask, what if a virus infection occurs, what should be done? The infected machines should be isolated to prevent any adverse effect on operations and the virus should be eliminated from infected machines. How many resources are applied to the problem? That depends on the impact. You would not drop all other tasks in IT to clean up a malware infection; there are too many other important operations that need support. The answer of how one responds is based on the risk associated with the problem.

Using risk management practices as the basis for making security decisions has a number of benefits. First, risk is a common currency between business and technology. Business executives may not care about the details of a polymorphic virus but they do want to know the impact if one works its way into a network. By formulating security plans in terms of risk, you are better able to align security with business objectives—and supporting business objectives is the ultimate goal of information security. Focusing on risk management also helps avoid falling into the trap of thinking being in compliance or maintaining a particular certification ensures security. Meeting these standards is a baseline and provides some assurance that sound practices are in use, but they do not guarantee security practices are aligned with business objectives.

Sharing Responsibility for Risk

Security cannot be an add-on feature or service provided by a security department. Security risks permeate an organization and its operations and so too must responsibility for it.

Line of business (LOB) managers are held responsible for other risks, such as market risks, customer dissatisfaction, account churning, and so on. Cyber security should be added to that list. This does not mean that LOB managers should become security experts but that they should understand risks to their operations from cyber threats and work with security staff to mitigate those risks. They should also be held responsible for ensuring security policies are followed and for providing feedback on the impact of security measures on operations.

When working with business partners, one necessarily needs to share responsibility for information security. This must be genuinely shared; one cannot depend on contract clauses that say partners will abide by best practices and preserve the integrity and confidentiality of information. These contractual agreements are necessary but not sufficient. Partners should work together at the implementation level to ensure practices are sufficient.

Use Security Metrics

Security metrics are something of a tricky topic. Some will argue that you do not need metrics to manage security (“malware is bad this year, it will be worse next year, what more do we need to know?”) but you cannot manage risk without metrics. You face multiple threats and you must be able to prioritize threats to business objectives. You also cannot just assume you will implement a broad spectrum of security measures and therefore cover all threats.

The budget probably does not exist for this and even if it did, the right combination of measures for one organization may not be the right mix for another. For example, one company may mitigate the greatest risk by focusing on perimeter defenses and intrusion prevention while another company might be better served by focusing on database auditing and encryption.

Given the need for metrics, the next question is what metrics? Checklists work well to ensure that basic policies are being enforced (for example, how many client devices are up to date on antivirus patches) but do not necessarily give a good indication of how well risks are mitigated. Composite metrics combine several metrics related to high-level measures, like resiliency and immunity, which measure how well a network can recover from an attack or resist an attack.

Realistic and Enforceable Policies

Security policies are constrained by operational requirements. From a security perspective, you might argue for strict controls on where customer data may be copied. Blocking all sales staff from downloading customer information to PDAs or smartphones may be too disruptive but providing encryption on these devices could be an acceptable alternative. Finding the right balance between security and functionality is also served by taking a risk-centric approach because threats, impact of threats, and costs of countermeasures are all taken into consideration. You should also keep in mind that policies that are in place but not enforced will undermine credibility in the policy process and security objectives.

Implement New Security Initiatives with Other Initiatives

When an organization takes on a new, broad initiative, such as a new focus on customer satisfaction or a server consolidation operation, it is an ideal opportunity to implement new security initiatives. You can often argue that security is required to realize objectives of the initiative. For example, a customer satisfaction initiative may include Web applications that provide customers with more access to their account information and such a system clearly requires robust security measures from the start.

New initiatives also provide an opportunity to introduce risk management practices. When planning the implementation of the initiative, consider the risks to implementation and ways to mitigate those risks. This is the ideal opportunity to introduce security measures and procedures that may eventually apply beyond the original scope.

Infusing an understanding for the need for information security into an organization is challenging but can be done. Risk management practices apply to both business and technical aspects of an organization and provide the foundation for several other ways to promote security awareness.

 Reference: Institute for Information Infrastructure Protection, [Embedding Information Security Risk Management into the Extended Enterprise: An Executive Workshop](#).

Article 17: Payment Card Industry Data Standards and Data Loss Prevention

The Payment Card Industry Data Security Standard (PCI DSS) defines requirements for preserving the confidentiality of credit card data. Given the widespread distribution of credit card information, this regulation is broadly applicable. There is much debate about best practices for implementation and which policies and techniques are sufficient and which are lacking. The finer-grained details about protecting data will continue to be debated and implementations will vary from one business to another, but several technologies and procedures will likely comprise the answer to PCI DSS compliance.

Before delving into those technologies and procedures, it is worth noting that before PCI DSS there were a number of other privacy regulations, ranging from the Health Insurance Portability and Accountability Act (HIPAA) in the United States, the European Union (EU) privacy directives, and a number of state regulations. IT managers and security professionals cannot be reduced to responding to each of these individually as they arise. A comprehensive data loss prevention program that addresses the fundamental problem of data confidentiality will address the requirements of individual regulations. Thus, rather than run down individual regulation's checklists, it is better to formulate and implement a broad data protection program that will address most if not all existing regulations and lay a foundation for addressing possible future regulations as well.

Taking PCI DSS as a starting point, the following are the basic requirements to protect confidential data:

- Network security measures
- Database and application security measures
- Data encryption
- Vulnerability management procedures
- Identity management and access controls
- Network, application, and database monitoring
- Data security policies
- Training and education

Volumes can, and have been, written on these topics; let's focus on a few that are especially relevant to the problem of data loss prevention: preventing data loss over the network, preventing data loss from hosts, and encrypting data at rest.

Preventing Data Loss Over the Network

Keeping credit card data secure while in transit over the network is a multi-faceted problem. Obviously, the data in transit as part of an online transaction process should be encrypted, but two other avenues of attack must be considered: attacks directed against an application and the transmission of confidential data outside standard operating procedures.

Minimizing Application Exposure

Applications and databases are prime targets for information theft. The process of securing networks is well understood and there are a number of technologies and practices that can be combined to effectively and significantly reduce the risk of data loss. Complex business applications, especially those dependent on relational database back-ends, present fundamentally different challenges and are more difficult to address. The solutions fall into three broad categories.

First, application vulnerabilities are often the result of poor coding practices. Applications that do not perform boundary checks are subject to buffer overflow errors; failing to validate inputs can lead to a number of different types of injection attacks. These vulnerabilities require changes to code to correct, or at the very least, external monitors, such as application firewalls, to detect malicious input before it reaches the application.

Second, application workflows should be optimized to minimize the number of times confidential data is transmitted. The fewest possible number of applications should have access to that data as well. By reducing the exposure of this data, you reduce the opportunities that attackers can exploit to steal the data.

Third, host-based data loss prevention (DLP) tools are designed to monitor storage and data manipulation operations on host devices. For example, a host-based DLP tool could block copying of a file containing credit card numbers to a USB flash drive. Host-based detection is especially important when confidential information can be copied to mobile devices.

If an attacker were able to access confidential information from an application, the next line of defense would be to prevent the data from being sent over the network.

Network-Based Data Loss Prevention

Network-based data loss prevention applications analyze TCP traffic searching for patterns of data. The patterns are defined in policies. When a stream of data matches a known pattern, the transmission can be blocked or logged for later review. One of the challenges with this type of detection is not disrupting normal business operations with false positives. This is addressed in two ways.

First, the patterns defined in policies must be sufficiently precise to reduce the numbers of false positives. In DLP systems that provide for training by example, accuracy can be improved by providing more representative data sets.

Second, the hosts and applications that process protected credit card information should be segmented from other areas of the network. The rules that are applied outside that segment can be broader and risk false positives without causing too much disruption to operations.

The final piece of the data loss prevention puzzle with respect to protecting credit card information is encryption of stored data.

Database Encryption

The major database vendors are now offering built-in encryption options. There are a number of factors to keep in mind when implementing database encryption.

First, there will be additional CPU demands to encrypt and decrypt data. Hardware accelerators can offload some of the processing, but this may introduce compatibility issues with specific versions of database software.

Second, one must decide when to encrypt and decrypt data. Certainly credit card information should be encrypted when it is on disk but data can move through complex workflows in which some stages require some data decrypted but not others. For example, if a series of services is called with and purchase transaction data is passed between the services, it is likely that few will require a fully decrypted credit card number; some might need only the final four digits of the card and the rest may not need any credit card data. In this case, when should the credit card data be decrypted? Probably the simplest option to implement is decrypting the data as it is read from the database, but this exposes the clear-text data through the entire workflow. Decrypting only in the services that need the data could be more secure but it would require more complex key management because the service would need access to the decryption key.

A third consideration is the amount of data to decrypt in an operation. At the lowest level, the database may decrypt at the page level but a finer-grained option is to decrypt at the row or even column level. Again, there are tradeoffs between lowering the complexity of the application and risking greater exposure of the data.

Summary

DLP tools clearly have a role in the PCI DSS. Network-based and host-based DLP complement each other to address data leaks over the network and through mobile storage devices. By segmenting applications and hosts that process the bulk of credit card information, the DLP policies can be crafted to reduce the incidence of false positives and false negatives. Database encryption also provides additional data loss protection but because it involves issues of application design and architecture, it is one of the more challenging technologies to implement.

Satisfying the requirements of data confidentiality regulations such as PCI DSS requires a multi-layer approach that includes DLP technologies. And even within DLP, a combination of complementary techniques will further reduce risks.

Article 18: Web Developers' Guide to Avoiding Cross Site Scripting Attacks

Web developers have a wide array of tools for creating the dynamic, feature-rich applications that many of us have come to expect. With the additional functionality of dynamic Web sites come potential vulnerabilities. One common problem with dynamic sites is cross site scripting, also known as XSS. (CSS is a logical abbreviation for cross site scripting but that is already commonly associated with Cascading Style Sheets).

XSS is a form of injection attack. Injection attacks take advantage of a user's ability to input data to an application to insert executable code or URLs into an application or Web site. The fundamental problem with all injection attacks is that inputs can include meta-characters which disrupt the normal execution. In the case of XSS, an attacker may post a message in a forum that includes malicious JavaScript that is executed when the message is read or a link is clicked. Another well-known form of injection attack, SQL Injections, disrupt the expected query processing in an application by including code that changes what the database query returns.

Which Applications Are Vulnerable?

Any application that accepts user input is potentially vulnerable to an injection attack. In the case of XSS, any application that accepts user input and allows for HTML, JavaScript, ActiveX, or other scripting code can be compromised if countermeasures are not employed. Attackers also target popular Web site tools like PhpNuke. Tools like PhpNuke that do not analyze input and remove potentially malicious code leave applications vulnerable.

 For PhpNuke XSS examples, see <http://www.securiteam.com/unixfocus/5HP042KE0S.html> and <http://phpnuke.org/modules.php?name=News&file=article&sid=4603>.

There is something of a myth or at least misunderstanding about XSS and Secure Sockets Layer (SSL) that sometimes shows up in XSS discussions. XSS vulnerabilities are not eliminated by using SSL and a site that does not use SSL is not necessarily vulnerable to XSS attacks. SSL is used to establish a confidential communication channel between a client and a server. The content of messages sent over the channel, whether it is secure or not, and how the application processes that content determines whether an injection attack is successful.

Preventing XSS

The key to preventing XSS attacks, and injection attacks in general, is by sanitizing user input. There are two general philosophies here. One approach defines rules for what is allowed in and another defines rules for what is not allowed at all. Both have their advantages and disadvantages.


Allow-In Filtering

Allow-in filtering uses formatting rules to define acceptable input. For example, the following regular expression can be used to validate a data input in mm/dd/yyyy format:

```
^\d{1,2}\/\d{1,2}\/\d{4}$
```

Any input with invalid characters, such as < or >, would be rejected.

Allow-in filtering provides the greatest degree of control over what is allowed for user input. This approach is most appropriate with database applications where the expected information is highly structured. For example, names, addresses, credit card numbers, and account numbers can all be reasonably well validated with regular expressions.

 Fortunately, a large library of regular expression (more than 1770 at the time of this writing) is available at <http://regexlib.com/Default.aspx>. The library includes patterns for dates, times, addresses, URLs, numbers, strings, phone numbers, and other string patterns.

This technique also helps reduce the chance of SQL injection attacks. Search forms of Web applications typically allow users to type in data that is then passed to a SQL query. Carefully formed strings can modify the intended purpose of a query. For example, if a user is searching for order information and they enter order number '1234,' the query may execute as something like:

```
SELECT
    customer_id, order_id, order_date, order_amount
FROM
    Orders
Where
    Order_id = 1234
```

A search form that did not sanitize input could be forced to return all rows in the order table with input such as '1234 OR 1 = 1,' which would result in the following query:


```
SELECT
    customer_id, order_id, order_date, order_amount
FROM
    Orders
Where
    Order_id = 1234 OR 1=1
```

Since the extra condition is true in all cases, all rows would be returned. Fortunately, the same regular expression that one would use to verify order numbers would work equally well to prevent XSS attacks. In most cases, though, the most likely targets of XSS attacks are free-form text fields that cannot be validated with regular expressions. In these cases, the keep-out filter is a better option.

Keep-Out Filtering

The goal of keep-out filtering is to prevent injected scripts from successfully executing. This can be done in one of two ways. The first method removes all characters defined in a list of disallowed characters, such as '<' and '>'. This can work so long as the developer thinks of all possible meta-characters that can be used in an injection attack.

The second method, allowing only known good characters, such as numbers, alphabetic characters, and punctuation marks is preferred to the first. There is less chance of a developer missing something that allows for a successful attack.

 See the CERT reference [How to Remove Meta-characters From User-Supplied Data in CGI Scripts](#), for an example of how this is done in Perl and C. Similar techniques can be used in other programming languages as well.

XSS is an injection attack that can be thwarted by careful sanitizing of all user input—especially that inserted into free-form text entries.

Article 19: Evolving Social Engineering Elements of Phishing

Phishing is a well-known and established fact of life for just about anyone with an email account. Phishers have been trying to lure us with scare tactics, like messages that our bank accounts may have been compromised, with appeals to greed, have you collected your millions from the Nigerian account holding your share?, and just about any other way the phishers can come up with to trick us. But we are getting better at recognizing these well-worn scams. Not to be outdone, phishers are changing their tactics. A key to the change is crafting new and more subtle social engineering techniques.

This article will briefly describe social engineering and then look at three ways phishers have adapted to an increasingly sophisticated pool of potential victims. The three alternative techniques each tackle the phishers' problem from a different perspective:

- With more information—Improving context in phishing lures
- With alternative technologies—Casting lures with phone phishing
- With new vectors—Phishing through online shops

Of course, in time, these techniques too will give way to improved or alternative versions, but as of today, awareness of these techniques is far less than that of traditional email-based phishing scams.

Social Engineering: The Basics

Social engineering, in the context of phishing, is a means of deceiving a victim by taking advantage of a cognitive bias in the victim, which hinders their coming to a rational decision about a topic. These attacks are generally based on a combination of presumed trust and another factor such as fear or greed.

Take a simple example. A bank, it happens to be your bank, sends an email with the bank's logo saying there has been some unusual activity in your account, please click the link in the message to reset your password. Looking at this objectively, if the bank is one of the larger financial institutions, phishers will have a relatively high rate of success in matching victims with their banks even if they send the messages at random. The fact that a message correctly identifies your bank should not add to your sense of trust about the message (it could be just chance), but it does. Also, the bank has probably notified customers on a number of occasions that it does not use email to communicate with customers and that it or its representatives would never ask you to disclose your password. Still, the concern over the possible breach in our account counts for more in our reasoning process than all the messages sent from the bank.


The combined sense of trust and urgency created by the phishing lure is often enough to get people to fall victim to scams. Growing public awareness about these scams can reduce the effectiveness of these techniques. It does not mean, however, that phishing is no longer effective; it just means that phishers have to change the way they instill a sense of trust and urgency in their victims.

Improving Context in Phishing Lures

Launching phishing attacks with large businesses—such as PayPal, eBay, Bank of America, Wells Fargo Bank, and others—is relatively easy. There is a fairly high probability that a significant number of recipients of the messages will actually be customers of the target company. That is no longer enough to ensure that victims will click through to the phishing site that actually collects information, though. The potential victim needs more to be convinced and there are a number of ways to add contextual information.

One approach is to narrow the distribution of an attack and use smaller, more local businesses in lures. For example, a phisher might craft a message purportedly from a regional credit union and target email addresses harvested from social networking sites that include geographic information in member profiles.

Another technique is to mine public data sources for personal information. A study of mother maiden name analysis by researchers at the University of Indiana documented just how easily one could find mother's maiden names. The researchers there were able to deduce the mother's maiden names of more than 4,000,000 Texans.

 For more information about mother maiden name attacks, see Virgil Griffith and Markus Jakobsson "Messin with Texas: Deriving Mother's Maiden Names Using Public Records" at www.informatics.indiana.edu/markus/papers/mmn.pdf.

Semi-public records can also be used as a source of information. For example, in a recent attack on Monster.com, an attacker posing as legitimate employers was able to steal information about 1.6 million Monster.com customers. The attack used a Trojan and (probably) stolen credentials of employers with access to the customer resumes. This kind of detailed information could possibly be used to craft highly targeted phishing scams.

Finally, social networking sites—such as MySpace, Facebook and LinkedIn—provide self-declared personal information that could be harvested for phishing scams. Another way to circumvent the growing awareness to email-based phishing scams is to shift to a different attack platform.

Phone Phishing


Phishers don't care how they lure you; just that they do. And, the means of luring must be low cost. Phishers are not going to launch a direct mail campaign using the postal service because the cost is prohibitive. Low cost Voice over IP (VoIP) technologies are just the key for opening another method of attack.

In phone phishing, the attacker can use a spam message to prompt a call or may directly contact the victim through an automated system. In the first case, the victims receive an email message indicating a problem with a bank account, credit card, and so on and ask the victim to call a phone number. This message doesn't ask the victim to click a link, which many email users are now more careful about, so the message appears more legitimate. If the recipient calls, an interactive voice response system will prompt for information such as credit card numbers and PINs.

Other phone phishers may just call victims directly, deliver the lure verbally, and prompt for personal information. If the attacker is able to spoof caller ID and make it appear that the call is from a legitimate business, chances of converting the recipient into a victim increase. Phone phishing moves the scam offline but other phishing attacks take advantage of other online opportunities.

Online Shopping Phishing

Unwitting shoppers can become the victim of phishers who create fake shopping sites and lure would-be customers to disclose names, addresses, and credit card numbers. One way to do so is to establish a shop and advertise through shopping aggregators such as Froogle or Yahoo! Shopping. Shoppers have become accustomed to having sellers rated on sites like PriceGrabber.com and BizRate.com. If a phisher were not able to spoof a quality rating on sites, he could resort to luring in victims by offering the lowest price. Many online services provide APIs, so a phisher could query other sellers for the price of popular products and under-sell those legitimate vendors in an attempt to catch the low-cost shopper.

 For more information about online shopping phishing, see "Case Study: Phishing on Froogle" by Filippo Menczer in *Phishing and Countermeasures: Understanding the Increasing Problem of Electronic Identity Theft* by Markus Jakobsson and Steven Myers (Wiley, 2007).

Summary

Phishers are changing their tactics to stay ahead of public awareness of their scams. A key to their continued success is improving their social engineering attacks. That, in turn, has led to a number of developments in phishing, including the use of more contextual information in phishing lures; the shift to alternative means of delivery, such as phone phishing; and the use of non-email-based lures, like online shopping bots and aggregators.

Article 20: Rootkit Countermeasures

Rootkits, one of the most insidious software-based threats, is by some definitions not even a type of malware. Yet rootkits can enable attacks that can hide malicious software, change the behavior of operating system (OS) components and utilities, mask changes to configurations, and incorporate themselves so deeply into a system that drastic measures are required to get rid of them. Clearly, it is better to prevent a rootkit infection than to try to eliminate one.

This article describes four levels of countermeasures that can reduce the risk of rootkit infection and its impact:

- Basic blocking defenses
- Local antivirus defenses
- Global hook blocking
- Trusted computing platform (TCP) countermeasures

These four levels proceed from early countermeasure prevention to contain a rootkit infection if it should occur.

Rootkit Countermeasure 1: Basic Blocking

The first layer of a multi-layer defense against rootkits begins with personal firewalls and content filtering. Personal firewalls can help prevent rootkit and malware infection by blocking accesses to IP ports on a device. This helps in cases in which infection is done through well-defined ports, such as ftp's port 21, or through randomly selected unassigned ports (above 1023). For example, a blended threat piece of malware may contain multiple programs that each attempt to communicate with a command and control server through a different means, such as chat, IM, ftp, and so on. By blocking outgoing communications on those ports, the malware cannot download additional code or instructions.

Of course, even with firewalls, most users will want port 80 open for HTTP traffic as well as other commonly used ports, such as those for email and instant messaging. In these cases, you do not want to block traffic but want to ensure that malicious content is not being transported over those ports. Network-based content-filtering appliances can block malware, spyware, rootkits, and other unwanted content before it reaches a computer.

The essential purpose of the first layer of defense is to prevent rootkits from even reaching a device; if they do reach the device, the remaining layers can make the difference between infection and maintaining an uncompromised device.

Rootkit Countermeasure 2: Local Antivirus Defenses

If a blended threat piece of malware carrying a rootkit reaches a PC, the best chances of discovering this threat lies with locally installed antivirus software. Although content filtering at the network level can detect and block a number of threats, the most sophisticated malware is polymorphic, so it changes patterns with each infection. Content filtering based on pattern matching will not detect polymorphic viruses and blended threats.

Antivirus software, however, uses pattern detection (also known as signature-based detection) and behavioral analysis. Polymorphic viruses avoid detection by using several techniques that change the underlying code in a piece of malware without changing its function; they are often encrypted as well. To execute their payload, though, these polymorphic viruses have to decrypt the malicious code and execute. Antivirus systems can simulate the execution of a potential threat in a sandbox and observe the behavior of the program. If the program carries out operations indicative of a virus, the program is blocked. At this second level of defense, a rootkit and its related virus or worm transport mechanism is blocked as it reaches a device but before it can execute and potentially infect the device.

Rootkit Countermeasure 3: Blocking Global Hooks

If a rootkit makes it past network and local antivirus defenses, the next line of defense is placed within the functioning of the OS. For the purpose of this article, we will focus on the basics of Microsoft Windows architecture; the principles apply to other OSs although the details will differ.

The Windows OS keeps track of operations within the OS with a queue of events. When a user types a key or clicks the mouse within a window, an event is registered in the queue. This makes programming much more flexible. For example, a programmer may design a multi-window application that can use the event queue to detect when a users shifts from one window to the other and respond accordingly. Programmers can code different actions to occur in response to different events. For example, when a key is typed, a program can record that key in a file as is done with keyloggers. This ability to tie an action to an event is called a hook.

Rootkits often make use of global hooks; that is, hooks that can execute regardless of which program triggered the event. A rootkit might use a file system event related to returning a list of currently running processes to run a program that removes the name of malicious programs from the list before returning it to the calling program. Host intrusion prevention systems (IPS) that block the implementation of global hooks can prevent rootkits from operating properly and thus provide the third layer of defense.

Rootkit Countermeasure 4: Trusted Computing Platform-Based Countermeasures

The last level of defense against rootkits is to block the use of a computer that has been successfully infected with a rootkit. Rootkits tamper with the OS, so you need a method outside of the OS to detect infection.

TCP is a hardware specification and protocol for detecting tampering with a system and preventing the execution of OSs that have been compromised. TCP computing does require specialized hardware, so this level of defense is not commonly available on deployed systems; however, we are likely to see more adoption of TCP in the future.

Summary

Rootkits are especially difficult to detect and remove because they modify the very tools and services we use to detect tampering. Preventing rootkit infection is preferable to having to remove a rootkit. A four-layer defense, including basic blocking, antivirus, global hook blocking, and the use of TCP, can reduce the risk of rootkit infection.

Article 21: Using Host-Based Intrusion Prevention Systems to Prevent Data Loss

Data is a valuable asset and some people out there want to get their hands on your data. At least that is the working assumption prudent IT professionals take. There are just too many stories in the news about data losses due to hackers, disgruntled employees, unintentional mistakes, and malware-infected devices to not take the threat seriously. Most organizations, of course, have multiple layers of security measures in place that reduce the risk of data loss. For example, firewalls, antivirus, anti-spyware, content filtering, URL blocking, and network-based intrusion prevention systems (IPS) can all block potential methods of attack or reduce the chances of successfully capturing and transmitting data. Another countermeasure that complements those already mentioned is host-based IPS. This article will examine some of the ways data loss occurs and describe how host-based intrusion protection can mitigate the risk of data loss.

Ways to Steal and Lose Data

Information managers have to deal with a wide array of methods for losing data that range from technically sophisticated attacks to simple mistakes:

- Hacking breaches of network and host security
- Social engineering attacks
- Theft with mobile storage devices
- Theft by disgruntled employees
- Lost mobile and storage devices


Hacking breaches are probably the first type thought of when considering data loss. In the case of TJX, a major retailer, credit card data about 40 million customers was stolen by hacking the company's wireless network. Hardening networks and servers by using strong passwords on routers and servers, running only necessary services on servers, and patching all devices and other measures, reduces the chances a hacker will be able to break in and successfully steal information.

One way attackers have gotten around technical barriers, such as strong passwords and access controls, is through social engineering attacks. The attacker tricks a person into disclosing a piece of information that gets them closer to the data. This could be done by masquerading as a service desk employee and asking for a password or with a phishing lure that leads a person to a Web site where confidential information is solicited.

A cardinal rule of information security is to limit physical access to devices. This is even more important today with the availability of high-capacity mobile storage devices. Flash drives can easily hold a directory's worth of data, iPods with 30GB and 60GB hard drives rival the size of PC disk drives just a couple of years ago, and perhaps the more worrisome are low-cost external hard drives that can hold a data mart's worth of data but only cost several hundred dollars. The intellectual property of a company, the customer database, and other valued assets can literally walk out the doors in someone's pocket or backpack.

The problem of mobile storage devices is especially acute when combined with the problem of disgruntled employees. Administrative assistants to executives may have access to confidential proprietary plans. Database administrators are responsible for maintaining essential operational data that ranges from inventory levels and customer orders to marketing data and HR information. This data could be stolen and, in small amounts, emailed to outsiders. (Content filters might be able to catch an email with confidential information, unless the thief is knowledgeable enough to encrypt the data first.) Larger amounts could be copied to an iPod or external drive and carried out. Network content filters and intrusion prevention devices would never catch these.

Another way data is lost is by accident. Over the past 2 years a number of stories have made headlines when laptops with private data are stolen or when backup tapes are lost. These have become so common, in fact, that they are no longer necessarily newsworthy.

 The Privacy Clearinghouse keeps a chronological list of data breaches and tracks incidents that do not necessarily make major headlines. The list is available at <http://www.privacyrights.org/ar/ChronDataBreaches.htm>.

Technical solutions will not help with some kinds of data losses, especially accidental loss of laptops or backup tapes. They may be of some help with social engineering, especially in the case of phishing attacks. When it comes to protecting against network hacking, disgruntled employees, and theft with mobile devices, host-based intrusion prevention can add another important layer to a multi-layered defense.

Blocking Data Loss with Host-based IPS

A host-based IPS can help mitigate the risk of data loss by blocking critical steps in the process of stealing data. For example, a host-IPS can:

- Detect changes to the system registry that are indicative of a rootkit installation. Rootkits are not technically malware but are used by other malicious programs to hide their presence.
- Prevent the installation of global hooks in the Windows message queue. These hooks could be used to capture keystrokes and other low-level events that could provide attackers with usernames and passwords.
- Block access to USB flash drives and other mobile devices so that data cannot be improperly copied from the host.
- Log unusual activities, such as file access outside of normal business hours.

A key difference between network-based and host-based IPS is that the latter runs an agent on the host and thus can exercise greater control, such as blocking access to USB devices. Another difference is that the host-IPS can be configured specifically for a particular device. A file server, for example, may be governed by different access policies than a database server. Host-based IPS also provide protection even if a device is not connected to a network, for example, when laptops are taken off site.

Summary

Data loss is a significant problem in information security and in many cases no single measure will sufficiently reduce the risk of this threat. A host-based IPS provides defense close to the data. If other defenses (such as firewalls and content filters) have failed and intruders have the potential to access data on a server, host-based IPS can provide a final or near-final block.

Article 22: 5 Evaluation Criteria for Selecting a Data Loss Prevention Product

Businesses are growingly concerned about preventing data loss. High-profile cases that involve major retailers and credit card industry companies to government agencies and academic institutions have suffered well-publicized breaches at the hands of outside attackers. Insider threats are a potential source of costly data loss as well. The security market has responded to the need for monitoring and controlling the flow of information with an array of data loss prevention (DLP) products. The range of options is good news for companies with DLP requirements, but how should one choose a DLP solution?

There is certainly no single best solution for all companies, but we can frame a set of features that customers should consider, in the context of their own requirements and constraints, when making a DLP purchasing decision. The key topics to address are:

- Policy enforcement
- Classification accuracy
- Reporting features
- Network- and host-based implementation
- Platforms supported

In general, all of these factors will be important considerations, but their relative importance will vary from one customer to the next.

Policy Enforcement

The first step in a DLP program is formulating a data classification system. Not all information is equally valuable or in need of the same level of protection. A data classification scheme can be as simple as labeling all data as either confidential, company, or public, with strict rules on confidential data, company data made readily available to employees and partners, and public data available to all.

With a data classification policy in place, the next consideration is how to enforce it using a DLP tool. When evaluating DLP solutions, consider:

- How easily one can map from the written data classification policy to defined rules within the system. How easily can policies be updated?
- What solutions are available to control access and changes to policies?
- Are templates available to help get you started?
- What reports are available to demonstrate to auditors the company is in compliance with regulations?

Classification Accuracy

A DLP solution is only as good as its ability to distinguish among different data classification categories. It must correctly identify confidential data without having classification rules so broad that they incorrectly categorize non-confidential data as confidential.

To measure the classification accuracy for each category, create a test data set with items in that category and items not in that category. For example, to test the classification accuracy of a DLP solution on confidential data, create a test set with confidential examples as well as some public and company samples. Have the DLP solution categorize the test cases and measure the following:

- True positives—The number of items in the category (for example, confidential test items) that were correctly categorized
- True negatives—The number of items not in the category (for example, public test items) that were correctly categorized
- False positives—The number of items not in the category that were classified as being in the category (for example, a public item classified as confidential)
- False negative—The number of items in the category that were classified as not being in the category (for example, a confidential item classified as public)

Ideally, true positive and true negative rates would be 100% and there would be no instances of false positives or false negatives. In practice, there will be tradeoffs. Increasing the number of true positives may require broadening classification rules so that false positives also increase. Similarly, tightening rules to reduce false negatives can lead to lower true positives. This is clearly a balancing act between competing priorities. Your requirements will help determine the optimal balance for your situation.

Reporting Features

A DLP solution should have sufficient reporting capabilities to allow administrators to easily assess the operational state of the system. This includes reporting on the rates of detection, the policies that are triggered when a data transmission is blocked, the user who is using the data when transmission is blocked, and so on.

Reporting for auditors is a key consideration. Companies should be able to prove that DLP measures are in place, how long they have been in place, what policies are applied, who has changed policies and when, and to what devices and data the policies apply.

Network and Host-based Implementations

Data loss tools may be network or host based. For optimal protection, both types should be used. Host-based prevention will be especially important when mobile devices are widely used in a company. Unless a laptop is connected to the network, the network-based solution will not provide protection. At the same time, a network DLP system will not detect confidential data copied from a PC to a locally connected USB flash drive.

When evaluating products, be sure to measure the classification accuracy of both host-based agents and network filters. They may use different technologies or be implemented with different code bases, so do not assume that if one component works well, the other will work just as well.

Platforms Supported

Heterogeneous environments are the norm. DLP tools for Windows-specific platforms may be insufficient to cover the breadth of data protection requirements in your environment. Non-Windows-based systems should be protected as well. Many non-Windows devices are servers, so network-based solutions may be sufficient here.

Also, network-based DLP solutions may run on Windows hosts and still protect data on non-Windows devices. The point is to make sure that data from Linux, UNIX, and other platforms are protected.

Summary

Organizations have a wide and growing array of options in the DLP market. Choosing the best DLP solution can be challenging, but by using the five evaluation criteria described and conducting in-house evaluations using your own data classification scheme and your own data, the top options should become readily clear.

Article 23: Human Factors in Improving Application Security

There is something of a disconnect between users and developers when it comes to application security. Developers and designers tend to think about the security of a system, that is, how to protect the integrity of the system, ensure confidentiality, and keep systems up and running in spite of threats from attackers and vulnerabilities in our own applications. Users do not usually think in those terms; for them, the questions are about trust. Is this Web site really my bank's site? Can I trust this online retailer not to sell or lose my credit card data? As application developers, we are faced with the challenge of bridging this divide. We must answer the question: how do we not only make this application secure but also make it appear trustworthy to non-technical users?

Human Factor Considerations in Application Design

Conveying a sense of credibility and trustworthiness to users is a wide open research area, but we can examine a few high-level considerations that developers and designers can consider:

- Agreements between trusted sites and users
- Visibility and understandability
- State and security status

Agreements Between Trusted Sites and Users

Some users have come to associate the locked padlock icon with a trustworthy site. Conventional wisdom, as well as research, points out that many users will not understand the details of SSL certificates and cryptography behind the padlock. Some studies have further shown that many users do not really get the basic idea of confidential communication represented by the icon; they know it has something to do with trustworthiness but the details are unclear.

The advantage of the padlock icon is that it is perhaps the most well-know image related to browser security. The icon represents something of an agreement between a client and a server about how data is handled, even if the user is not quite sure of the details. This model has been extended by some companies, such as Bank of America, which has customers select an image that is displayed as part of the authentication process. This represents an agreement in the form of a shared secret that may help users be more confident that they are working with a trusted site.

Applications that incorporate low-tech components such as this may improve the security of transactions with customers. For example, customers may be less likely to be lured by phishing sites that cannot correctly display their selected image. A non-technical agreement, such as the image, also takes advantage of another consideration in application security: visibility.

Visibility and Understandability

Visibility of security information allows users to distinguish trusted from non-trusted sites quickly. Take for example, the new extended validation (EV) certificates. These certificates turn the address bar to green in Internet Explorer (IE) 7 (and in Firefox 1.5 and 2.0 with an add-on from VeriSign). The address bar also contains the name of the organization registered as owner of the site.

The value here is that users can presumably quickly distinguish a real site, such as Paypal.com, from a look-a-like phishing site, like Paypa1.com (using the number 1 instead of the letter “L”). One of the requirements for an EV certificate is that the URL not be similar enough to another site to cause confusion, so it is unlikely that another company could acquire an EV certificate for Paypa1.com (with the number 1).

Users can, of course, click the padlock icon in the address bar and view certificate details to find the name of the organization (see Figure 1).



Figure 1: A conventional SSL certificate displaying organization information.

Most users do not know how to find this information, and if they did, may not understand its importance. This is where the principle of understandability comes in. Security information should be mapped to more easily understood metaphors, such as green bars and padlocks instead of being left in technical details.

State and Security Status

The green bar display of an EV certificate is one way to show the security status of a site. Similar indications can be added to displays to denote sensitive operations, such as:

- Entering credit card data for a transaction
- Updating customer account information
- Confirming a request made by email

The security status could be indicated with the use of icons, for example, to show the sensitive nature of the operation, improve user's awareness about issues around online security, and build a sense of trustworthiness with the site.

Summary

Helping users distinguish trustworthy from untrustworthy sites cannot be done with hidden security measures. EV certificates, for example, use visual cues to indicate the owners of the site have undergone a more rigid validation process than other sites and that a site that displays a green bar in the browser is likely the legitimate business it claims to be. As we develop a better understanding of human factors and their role in improving security, we may be able to apply similar visual cues. In the meantime, some experimentation is in order.

Article 24: 5 Issues to Consider with Mobile Device Encryption

There is no shortage of headlines about lost or stolen laptops containing confidential data. There is also no shortage of concern about preventing data loss. When it comes to protecting information on mobile devices, encryption is a fundamental component of the solution. Mobile devices—such as laptops, smartphones, PDAs, and removable storage media—enable much more flexible and adaptive work arrangements than if we were forced to keep data tied to desktop devices. For many, these devices just would not be as useful if they were restricted to not storing intellectual property, confidential business data, or customer information. Yet, this is exactly the data that requires the greatest protection.

Protecting mobile devices requires a multi-pronged strategy, including user awareness, clear policies about the use and storage of mobile devices, restrictions on installing or updating software, and of course data encryption. Encrypting a file is simple. Managing encryption across an enterprise's collection of laptops, smartphones, and PDAs is not. Five topic areas should be considered when planning to implement mobile device encryption:

- Policy management
- Scope of encryption
- Authentication mechanisms
- Supported platforms
- Compliance issues

Each of these areas entails consideration of advantages and disadvantages of various options and finding the optimal mix to support an enterprise's specific requirements.

Policy Management

The first step to establishing mobile encryption is formulating a policy to govern the encryption. This includes components, such as:

- Defining the encryption algorithm and key length or some equivalent definition of minimal encryption strength.
- Determining the type of information that is to be encrypted. This may be as simple as encrypting all data on mobile devices or it may be more selective based on data classifications.
- Establishing reporting and auditing guidelines.
- Developing a plan for key management and recovery.
- Setting guidelines on how to respond to multiple failed attempts to access a device. For example, should the device be locked and unlocked after a certain time, unlocked only by a systems administrator, or should the data on the device be erased?

The policy frames the options for several implementation details, such as the scope of encryption.

Scope of Encryption

In many cases, encrypting all data on a device may be the most secure and easiest to manage option. However, for older, lower performance devices, this may put an unacceptable load on the CPU. In that case, selective file encryption may be the best option.

Removable devices, especially USB flash drives, should be included in consideration of mobile device encryption. Some USB flash drives support encryption natively, but if not, host-based data loss prevention (DLP) applications can be used to ensure that sensitive information is not copied to removable media in a way that violates security policy.

Authentication Mechanisms

Encryption protects data from unauthorized access, but if an attacker can break device authentication, the effectiveness of encryption is essentially lost. A device worth encrypting is a device that is worth protecting with at least strong passwords. Minimal password strength can be defined and enforced through policies.

Additionally, biometric devices, such as a fingerprint scanner, may be used to authenticate users. These are readily available for laptops and may become more popular on other mobile devices in the near future.

In the enterprise environment, one should also consider how device authentication will integrate with enterprise identity management and access controls. For example, if a single sign on (SSO) solution or Lightweight Directory Access Protocol (LDAP) directory is used with non-mobile infrastructure, users may expect the same level of integration with mobile devices.

Supported Platforms

Heterogeneous environments often require IT to support multiple applications for mobile device encryption. Laptops may run Windows XP or Windows Vista, Mac OS X, and increasingly Linux. Mobile phones may run Windows Mobile, Symbian OS, or a customized version of the Mac OS X for the iPhone. Soon, we can expect to see greater use of Linux on low-cost mobile devices.

Linux, the open source OS, is becoming especially popular in mass-market, low-cost devices like the Asus Eee. Google is rumored to be working on a version of Linux for its anticipated Google Phone. Just as the Blackberry entered many organizations through grassroots demand rather than IT-driven strategy, we can expect low-cost Linux devices to enter the device mix over the next year.

IT departments will have to balance a number of factors when choosing the best mix of encryption support for a heterogeneous environment. A company may decide to use a single product line that works across platforms in order to ensure that all devices can be managed from a single console and with a single policy enforcement mechanism. Another organization may decide to use OS-specific options, such as Microsoft Vista's BitLocker and Encrypting File System (EFS), when available while deploying a third-party encryption tool for other devices. This would make sense in cases in which the cost savings outweigh the additional management and service desk overhead.

Compliance

The fifth issue to consider with mobile device encryption is compliance. Compliance requirements will heavily influence and frame policy formulation. Having a policy, of course, is not enough. The policy has to be enforced and organizations need to prove that their policies are enforced. If a laptop is stolen, how can we be sure that all the data was encrypted?

Reporting and audit support are essential features when it comes to managing mobile device encryption across the enterprise. Using multiple encryption methods (for example, Windows Vista BitLocker for some and third-party applications for others) can make this more difficult.

There is clearly a need for mobile device encryption, and hardware and software vendors have responded with a variety of options. Keeping in mind the five issues outlined here will help to narrow the pool of candidates to the most appropriate for an organization's particular needs.

Article 25: Basics of Database Auditing

Database security is gaining more attention—and justifiably so. Concerns about compliance, privacy protection, and data loss will naturally lead to measures to secure databases and the data they hold. As with other areas of information security, a defense-in-depth strategy can significantly reduce risks associated with databases. A key element of this strategy is database auditing.

Database auditing can be divided into two subtasks: security assessment auditing and information access auditing. A security assessment audit entails vulnerability scanning, code reviews, and an analysis of database management policies and procedures. These are typically done at regular intervals and in conjunction with broader IT operations audits. Information access auditing is more of a monitoring operation and will be the subject of this article. For the remainder of the article, when reference is made to database auditing, it means information access auditing.

What Is Auditing?

Database auditing is the process of recording information about significant events during the operation of a database. The definition of “significant event” will vary by type of database, applications using the database, and the type of information stored there. In all cases, database auditing requires that a reliable set of details is maintained and is not tampered with. This information is used to understand the normal operations within a system, assess trends in database use, and most importantly, detect unusual or unauthorized activity within the database.

What to Audit in a Database?

Given the goal to detect unauthorized activities, database auditing should include events that could provide information about changes to the database or about the user or agent that made those changes. Some of the most commonly logged events in audit trails include:

- Login and logout events
- Changes to data structures
- Privilege changes
- Database errors
- Views and modifications of confidential data

Login and Logout Events

Do regulations or other requirements dictate that all user access to a database be recorded? The answer depends on the situation. For example, a database of personal financial records may warrant detailed logs of every time a user accessed the database. A product catalog queried to generate dynamic Web pages for an e-commerce site is unlikely to need such level of detailed tracking. When dealing with sensitive data such as financial, health care, and intellectual property, recording all login and logout events is probably warranted. In other cases, logging only failed attempts may be sufficient. Multiple failed attempts on a single account or a large number of failed attempts over a large group of users could indicate an attempted attack. Login and logout events are infrequent enough that the performance hit by auditing these are minimal.



It should be noted that many multi-tiered applications use a pool of database connections. Connections are created and kept open for use by multiple users to minimize the time required to fetch data from a database. A database log may show a login event when the connection is created and then an event when the connection is destroyed, perhaps hours later. In the meantime, dozens of different application users may have used that one connection. To track which application user made use of that connection would require additional audit detail available only at the application level.

Changes to Data Structures

Relatively few users should have the ability to changes tables, views, indexes, and other data structures in a database. In test and production environments, database administrators are often the only users with those privileges. Development environments are often more open to changes with data modelers and some developers are given the rights to create and change data structures.

Data structure changes should be relatively infrequent in production environments. When they do occur, they should happen within normal maintenance windows. Changes outside these times may indicate a security breach; for example, an insider may be stealing data and using a temporary table to store the data before exporting or downloading it to another server.

Privilege Changes

Privileges are the right to perform some operation on a database object. The most common are privileges to insert, update, delete, and select from tables and select from views. In some well-designed databases, users are granted the privilege to execute a trusted procedure that performs the insert, update, delete, or select on their behalf. This approach can provide an additional level of security by restricting user access to data structures.

There are also privileges to perform operations within the database, such as to create tables, export data, drop low-level storage structures, and create links to other databases. Anytime privileges are changed, there should be an audit trail. Again, these events should be relatively infrequent compared with the number of times a table or view is queried. This level of detail is useful in cases in which an attacker has to elevate privileges or grant additional select privileges to access data of interest.

Database Errors

Recording information about database errors is essential in production databases. Not only does this information help diagnose operational problems, it can indicate probes of the database. Attackers that use SQL injection attacks often have to experiment with the syntax of their input to get a proper string. This experimenting will generate errors that, as with failed login attempts and elevated privileges, can be a sign of malicious activity.

Modifying and Viewing Sensitive Information

In highly regulated areas, such as healthcare, knowing who is looking at and changing data can be a requirement. Auditing every select statement can put a significant load on a database—remember, every audit record has to be written to an audit table or audit file and I/O operations are some of the most time consuming in database applications.

When requirements are more flexible, you might want to balance the additional overhead of logging every select operation with the value that audit information provides. It may be sufficient to log inserts, update, and delete operations only. In the cases of large batch inserts, such as during loads of a data warehouse, you might want to disable logging of insert operations and depend on the metadata about data loading and data provenance stored in the data warehouse to trace the date, time, and account used to load the data.

How to Audit?

How to audit is the most database-specific question one must answer. The major database vendors and open source projects vary in their support for auditing. DB2, for example, provides administrators with commands to define the types of auditing they would like; Oracle allows custom-written code to execute on defined system events, such as a login, or data manipulation events, such as an insert or delete event. Optimizing database auditing operations requires a detailed understanding of the tools available in a database and the best practices that have been developed for that database. Database auditing is an important part of defense-in-depth practices for database security, but both the scope of auditing and the implementation method will vary by type of application and database used.

Article 26: 5 Security Considerations with Portals

Portals are multi-tiered applications that provide a framework for unifying an array of applications and services. The flexibility they bring for developers and users also carries a number of security considerations centered on topics such as identity management, client-side security, and application management. In this article, we will examine five topics that should be addressed when developing and deploying portal-based applications:

- The role of the identity provider
- Propagating identities
- Client-side security
- Service deployment and patch management
- Portal administration

These closely related topics apply to other Web applications as well but the broad application of portals and the breadth of services potentially provided in a portal make these critical to the success of a portal deployment.

Role of Identity Provider

Portals are distributed systems with at least three tiers: presentation-level services, application services, and data services. Users interact with presentation-level services of the portal. One of the advantages of a portal framework—such as IBM WebSphere Portal or BEA WebLogic Portal—is a customizable interface. Functional interface components, called *portlets*, can be configured differently for different users depending on their needs. A department manager, for example, may log in to a portal to find a dashboard with key performance measurements for her department while a sales person may log in to the same portal and find a contact manager, calendar, document management system, and other sales support tools. Personalized configurations such as this are based on the identity of the user, and identity information must be provided by some part of the portal framework.

Identities, groups, and role information should be consolidated in a single or a small number of identity providers. These identity providers may be Lightweight Directory Access Protocol (LDAP) servers, Active Directory (AD) databases, or other enterprise-scale directories. In the simplest case, all applications within the portal use the same identity provider. For example, the portal interface uses the identity provider to authenticate users and configure their personalized settings, Web services use identity credentials to provide access to data and services, and database applications use identities to determine what data is accessible to a particular user. In more complex portals, multiple identity management systems may be used. This raises a number of questions that designers and developers must keep in mind:

- Are identities duplicated across identity providers and, if so, what attributes and roles are managed by each?
- How are multiple providers federated?
- Who controls the management of identities within a single provider?
- Are there common governance procedures across all identity providers?

When multiple identity providers are used, portal designers may need to manage multiple sets of credentials to simulate single sign-on (SSO) services. Services, such as the IBM WebSphere Credentials Vault, can be useful for this purpose.

Propagating Identities

Once a user has authenticated to a portal, the user's credentials may need to be passed to other services used by the portal. Consider an example of a portal providing a business intelligence dashboard to a department manager. The dashboard includes a graph showing sales results for the past month by region, and the data is retrieved from a data warehouse. The portlet displaying the results makes a database query through a BI reporting tool service and requires a parameter specifying the identity of the user.

If this were a monolithic system, the portlet could pass in a parameter with the department number and return the results for that department. However, this BI service is used by multiple applications and so makes no assumptions about the trustworthiness of the calling program. The BI service requires cryptographically signed credentials that it uses to determine the identity of the user and her role within the organization. These credentials may include authorizations defined in Security Assertion Markup Language (SAML), an XML standard for securely exchanging role and authorization information between systems. Based on that role, the database returns data relevant to her department.

The use of more service-oriented architectures introduces great flexibility into the application development process but removes the idea of a single, unified authorization context. For this reason, identities and authorizations must be propagated through service calls.

Client-Side Security

From the perspective of portal applications, no assumptions should be made about any code that runs on the client. For example, if inputs are validated by JavaScript, which is executed on the client and then passed to the server, the validation results should not be trusted. An attacker could easily view the page source, copy the code removing the validation segments, and then make calls to server-side services. In portal applications and Web applications, all validation should be performed on the server.

Service Deployment and Patch Management

Patching applications is a routine part of systems management and helps to maintain overall security. Patching in portal applications is just as important but can be more difficult to track. The problem arises because of the distributed nature of the Web services and application services that are aggregated within a portal.

Portal applications often pass data from one service to another. For example, in the presentation layer, one portlet may pass data to another portlet to coordinate the display of data. A portlet showing tabular data from a database query may send data to a graphing portlet to display the same data graphically. In another case, one portlet may send data to another portlet that is used to query another database. This creates the potential for a SQL injection attack.

The receiving portlet should validate the input (on the server) before sending it to the database. If the validation is lacking and later implemented in a way that requires a change in the portlet interface, patching the vulnerability will break the inter-portlet communication. Dependencies such as this must be considered when deploying and patching portlets and services within a portal framework.

Portal Administration

The distributed nature of portals provides for mechanisms for using a variety of applications within a single framework. This also means that portals become dependent on multiple source systems that might be administered by different staff and perhaps even in different companies or under different policies. As portals are designed, clear lines of responsibility should be established along with procedures for communicating changes in administration policy and procedures to reduce the chances a vulnerability falls through the management cracks (for example, “I thought you validated the input for that call”).

Portal frameworks enable the rapid development of composite applications. Their distributed nature, however, imposes some additional security considerations not found in more monolithic applications.

Article 27: Role of Code Reviews in Application Security

Software developers have to juggle numerous requirements and constraints. Applications have to meet functional specifications, maintain sufficient levels of usability, and meet performance expectations. They also have to be reliable, scalable, and, of course, secure. For decades, software engineers have created and improved development methodologies and life cycle management processes to corral these sometimes competing demands and meet as many of them as possible while delivering applications on schedule and on budget. Code reviews are an increasingly important part of the software management life cycle.

A code review is a process in which application code is studied for potential bugs and vulnerabilities. The developer of the code likely participates but at least one person not involved in the development of the code is required to bring a fresh set of eyes to the process. A security code review specifically focuses on finding potential vulnerabilities related to topics such as

- Authentication
- Authorization
- Session management
- Encryption
- Input validation
- Error recovery
- Logging

Code reviews can occur at a fixed point in time, for example, if one is using a waterfall development methodology; at regular intervals in the development cycle if using a spiral methodology; or more frequently but at less well-defined cycles if using agile methodologies. Different methodologies work well for different teams under different circumstances and we will not argue for one over the other here. We will instead address the questions that apply across methodologies:

- What code should be reviewed?
- What should be done during a code review?

Identifying Code to Review

In an ideal world, all code would be subject to review. Actually, in an ideal world, we would not need code reviews, but most development teams are already under tight deadlines with existing demands and the often present problem of feature creep. One can identify the most likely code to benefit from review and have the greatest return with respect to security. The following types of code should be reviewed:

- User input code, which should be reviewed for potential injection attacks
- SQL queries that could be used for SQL injection attacks
- Any code that listens on a network port
- Code that is used in authentication and authorization operations
- Code that processes confidential or high-value data
- Code that has been used in previous exploits

Fortunately, once code has been reviewed once, subsequent reviews often take less time. The flip-side of the coin is that the initial code reviews can be time consuming. Once the segments of an application that will be reviewed have been identified, the review itself can commence.

Parts of a Code Review

The main parts of a code review include:

- Collecting source code and libraries
- Compiling and building the application
- Conducting automated vulnerability scanning
- Performing a methodical, manual review of code

The first steps seem obvious, but it is important to properly scope a code review. It is not reasonable to expect to be able to review all components in a large, multi-tiered application in one code review. Applications depend on libraries for which source code is not available. Therefore, the first steps focus on selecting a subset of an application and understanding what code can be manually and statically reviewed and which will have to be subject only to runtime vulnerability scanning.

The next step is to compile the code. This serves two purposes. First, it shows the code is minimally ready for review. If code will not compile, it is not ready to test. Second, it provides an executable that can be use with runtime vulnerability scanning tools.

Many tools are available for static analysis. These work with source code and look for common errors, such as out of bounds array references and memory leaks. Technically, one could run static analysis tools on code that does not compile. The problem is that compiler errors will need to be fixed any way and in the process, new bugs could be introduced; thus, you end up having to re-run the static analysis.

Dynamic, or runtime, analysis should also be done. In this process, programs execute target code and probe for vulnerabilities. One common type of test is to generate out-of-range input sets to determine whether buffer overflows or other exploitable exceptions occur. For example, if an input field accepts a character string up to 10 characters, a test program might pass it a 20- or 100-character length string. How the tested code responds can indicate potential vulnerabilities.

The last, and most time-consuming, part of the security code review is the manual review. An experienced developer with knowledge of code exploits and common errors should perform this step. Errors in arithmetic operation (for example, division by 0), exposed or short cryptography keys, non-standard cryptography algorithms, and non-validated input are some of the problems to look for.

Summary

Security code reviews provide an opportunity to step back and examine an application for potential vulnerabilities. It is not a time to optimize code or find other bugs (although that may happen). By focusing on the most likely target areas, code reviewers can maximize the potential benefits of a code review conducted under less-than-ideal time constraints.