

Chapter 1

Systems Security

COMPTIA SECURITY+ EXAM OBJECTIVES COVERED IN THIS CHAPTER INCLUDE THE FOLLOWING:

✓ **1.1 Differentiate among various systems security threats.**

- Privilege escalation
- Virus
- Worm
- Trojan
- Spyware
- Spam
- Adware
- Rootkits
- Botnets
- Logic bomb

✓ **1.2 Explain the security risks pertaining to system hardware and peripherals.**

- BIOS
- USB devices
- Cell phones
- Removable storage
- Network attached storage

✓ **1.3 Implement OS hardening practices and procedures to achieve workstation and server security.**

- Hotfixes
- Service packs
- Patches
- Patch management





- Group policies
- Security templates
- Configuration baselines

✓ **1.4 Carry out the appropriate procedures to establish application security.**

- ActiveX
- Java
- Scripting
- Browser
- Buffer overflows
- Cookies
- SMTP open relays
- Instant messaging
- P2P
- Input validation
- Cross-site scripting (XSS)

✓ **1.5 Implement security applications.**

- HIDS
- Personal software firewalls
- Antivirus
- Anti-spam
- Popup blockers

✓ **1.6 Explain the purpose and application of virtualization technology.**



The Security+ exam will test your basic IT security skills—those skills needed to effectively secure stand-alone and networked systems in a corporate environment. To pass the test and be effective in implementing security, you need to understand the basic concepts and terminology related to systems security as detailed in this chapter.

1.1 Differentiate among various systems security threats.

Knowing how to recognize and respond to a wide variety of security threats is an essential skill in today's networking environments. Security is not just about locking down the environment against threats, but also about detecting breaches and being prepared to respond to incidents. This section discusses several common threats that all environments must be prepared to face.



For more information on this topic, refer to Chapter 1 of the *CompTIA Security+ Study Guide, 4th Edition* (Sybex, November 2008).

Privilege escalation

Privilege escalation is the malicious event when a user is able to obtain privileges or capabilities beyond what they were assigned or which they are authorized to have or use. Privilege escalation can be performed by a normal user, an administrator, or an outside attacker.

Privilege escalation can be performed by exploiting administrative oversights or misconfiguration of the environment. It could also be performed by clever manipulation of the systems or even through hacks. Some hacks, such as GetAdmin, temporarily grant the current user account full administrative privileges, while other hacks, such as X.exe, create a new user account in the Administrators group with a known password. Privilege escalation hacks can also be performed by stealing the credentials of another user account (such as through password guessing, password cracking, or authentication packet sniffing).

The best defenses against privilege escalation include clearly defined job descriptions with privileges that are enforced and restricted on a detailed basis, strong password policies, and detailed auditing of the environment.

Virus

Viruses are just one example of malicious code, malicious software, or malware. *Malicious code* is any element of software that performs an unwanted or undesired function from the perspective of the legitimate user or owner of a computer system. Malicious code includes viruses, worms, Trojan horses, spyware, adware, rootkits, botnets, logic bombs, and sometimes even spam.

Viruses get their name from their biological counterparts. They are programs designed to spread from one system to another through self-replication and to perform any of a wide range of malicious activities. The malicious activities performed by viruses include data deletion, corruption, alteration, and theft. Some viruses replicate and spread so rapidly that they consume system and network resources, thus performing a type of denial-of-service (DoS) attack.

Most viruses need a host to latch onto. The host can be a file (as in the case of *common viruses*) or a sector of a hard drive. Viruses that attach themselves to the boot sector of a hard drive and thus are loaded in memory when the drive is activated are known as *boot sector viruses*. *Polymorphic viruses* have the ability to alter their own code in order to avoid detection by antivirus scanners. *Macro viruses* live within documents or e-mails and exploit the scripting capabilities of productivity software. *Stealth viruses* attempt to avoid detection by masking or hiding their activities. *Armored viruses* are designed to be difficult to detect and remove. *Retroviruses* specifically target antivirus systems to render them useless. *Phage viruses* modify or infect many aspects of a system so they can regenerate themselves from any remaining unremoved parts. A *companion virus* borrows the root filename of a common executable, and then gives itself the .com extension in an attempt to get itself launched rather than the intended application. *Multipart or multipartite viruses* perform multiple tasks and may infect a system in numerous ways.

The best countermeasure to viruses is an antivirus scanner that is updated regularly and which monitors all local storage devices, memory, and communication pathways for viral activities. Other countermeasures include avoiding downloading software from the Internet, not opening e-mail attachments, and avoiding the use of removable media from other environments.

Worm

Worms are self-contained applications that do not require a host to infect. Worms typically are focused on replication and distribution, rather than on direct damage and destruction. However, due to the expanding capabilities (although malicious) of viruses, worms are no longer an easily identifiable, distinct category of malicious code. Worms are designed to exploit a vulnerability in a system (operating system, protocol, service, or application) and then use that flaw to spread themselves to other systems with the same flaw. Worms may

be used to deposit viruses, Trojan horses, logic bombs, zombies/agents/bots for botnets, or they may perform direct virus-like maelstrom activities on their own.

Countermeasures for worms are the same as for viruses, with the addition of keeping systems patched.

Trojan

A *Trojan horse* is a form of malicious software that is disguised as something useful or legitimate. The most common forms of Trojan horses are games and screensavers, but any software can be made into a Trojan horse. The goal of a Trojan horse is to trick a user into installing it on their computer. This allows the malicious code portion of the Trojan horse to gain access to the otherwise secured environment. Some of the most common Trojan horses are tools that install DDoS zombies or remote control agents onto systems (see Chapter 2 for more information on denial-of-service).

Countermeasures for Trojan horses are the same as for viruses.

Spyware

Spyware is any form of malicious code or even business or commercial code that collects information about users without their direct knowledge or permission. Spyware can be fully malicious when it seeks to gain information to perform identity theft or credential hijacking. However, many advertising companies use less-malicious forms of spyware to gather demographics about potential customers. In either case, the user is often unaware that the spyware tool is present or that it is gathering information that is periodically transmitted back to some outside entity. Spyware can be deposited by viruses, worms, or Trojan horses, or it can be installed as extra elements from commercial, freeware, or shareware applications.

Countermeasures for spyware are the same as for viruses, with the addition of specific spyware-scanning tools.

Spam

Spam is any type of e-mail that is undesired and/or unsolicited. Think of spam as the digital equivalent of junk mail and door-to-door solicitations.



Some studies have shown that more than half of all e-mail now consists of spam.

Spam is a problem for numerous reasons:

- Some spam carries malicious code such as viruses, logic bombs, or Trojan horses.
- Some spam carries a social-engineering attack (also known as hoax e-mail).

- Unwanted e-mail wastes your time while you sort through it looking for legitimate messages.
- Spam wastes Internet resources: storage capacity, computing cycles, and throughput.

The primary countermeasure against spam is an e-mail filter. An e-mail filter is a list of e-mail addresses, domain names, or IP addresses where spam is known to originate. If a message is received from one of the listed spam sources, the e-mail filter blocks or discards it. Some e-mail filters are becoming as sophisticated as antivirus scanners. These e-mail filters can examine the header, subject, and contents of a message to look for keywords or phrases that identify it as a known type of spam, and then take the appropriate actions to discard, quarantine, or block the message. In addition to client application or client-side spam filters, there are also enterprise spam tools. Some enterprise tools are actually stand-alone devices themselves, often called anti-spam appliances, while others are software additions to internal enterprise e-mail servers. The benefits of enterprise spam filtering is to reduce spam distribution internally by blocking and discarding unwanted messages before they waste storage space on e-mail servers or make their way to clients.

However, e-mail spam filters are problematic. Just because a message includes keywords that are typically found in spam doesn't mean that every message with those words is spam. Some legitimate, if not outright essential, messages include spam words. One method of addressing this issue is for the spam-filtering tool to place all suspected spam messages into a quarantine folder. Users can peruse this folder for misidentified messages and retrieve them.

Another important issue to address when managing spam is spoofed e-mail. A *spoofed* e-mail is a message that has a fake or falsified source address. When an e-mail server receives an e-mail message, it should perform a reverse lookup on the source address of the message. If the source address is fake or nonexistent, the message should be discarded.

Hoaxes

A *hoax* is an e-mail message that includes incorrect or misleading information. This is a written or static form of a social-engineering attack. Hoaxes are common and widespread because they expertly prey on human nature. If e-mail recipients aren't prepared for hoaxes, they can be easily caught up in them or persuaded by them. Hoaxes may inform you of intended court cases or legislation and encourage you to support one side or the other with a donation. They may warn you of a quickly spreading virus and provide details on how to sanitize your computer, such as deleting certain files or editing the Registry. Hoaxes also include chain letters that promise good fortune, bypassing of bad luck, or accumulation of wealth by passing the message on to others.

Although a hoax isn't the same as a virus in that it does not cause any direct damage, it often ends up causing nearly as much damage as a virus would. When ignorant users follow the instructions of a hoax—especially those that instruct readers to delete files or alter their system configuration via the Registry—the users usually end up damaging their operating system so severely as to require a reinstall or restoration from backup. Even if the damage isn't immediately or obviously severe, sometimes the instructions in a hoax

open up vulnerabilities so that real viruses, remote-control hacker tools, or other forms of malicious code can gain access.

Your primary weapon against hoaxes is education and awareness. E-mail users should be on the lookout for any message that promises the unlikely, seems too good to be true, or has dire warnings that require immediate action.

One method for improving the security of your organization when it comes to dealing with e-mail hoaxes as well as spam and other unwanted messages is to develop a response policy. A spam response policy should define the steps users should follow when they receive a message that might be a hoax, whether intentional or not. Some recommended response steps include the following:

1. Notify your network administrator that you have received a suspected hoax.
2. Check with your antivirus vendor for confirmation of malicious code–related issues.
3. Check with your antivirus vendor for e-mail hoax–related issues (most maintain a database of hoaxes).
4. Find at least three other reliable, public, trusted sources to corroborate any message, especially if it involves legal or monetary issues.
5. Don't forward any message to others if the message specifically directs you to do so. If there is a legitimate security issue spreading across the Internet, the security watchdogs will respond and inform the public appropriately.
6. Never follow the directions in an e-mail from an unknown or untrusted source. Always validate procedures from a trusted, reliable source (such as Microsoft, a software vendor, an antivirus vendor, or your ISP).

If you discover a hoax, especially one that isn't already cataloged in your antivirus vendor's hoax database, send the vendor a copy so it can inform others. Be sure to contact the vendor and ask how to submit examples of hoaxes; don't just forward the message.

E-mail

E-mail allows for fast, efficient communications across the Internet. There are more e-mail addresses than there are actual Internet users, because many people have multiple e-mail addresses, whether by chance or by choice. E-mail offers individuals and companies alike a means to communicate without paying any type of per-message fee (such as postage fees associated with snail mail) and allows messages to be delivered in seconds rather than days. However, these abilities of e-mail also make it ripe for exploitation by those with malicious or at least nonbenevolent intentions, such as spam or hoaxes.



E-mail relaying is another important issue. That topic is discussed in the "SMTP open relays" section later in this chapter.

Because e-mail is so widely used, it has become the most prevalent delivery vehicle for malicious code such as viruses, logic bombs, and Trojan horses. To combat this threat, you should deploy an antivirus scanner to scan e-mail content and attachments. You should even consider stripping or blocking e-mail attachments (especially those with known extensions of scripts or executables) as they enter your network (on an e-mail gateway, firewall, and so on). It is always the more secure option to scan, check, and if necessary, strip e-mail on SMTP servers before it reaches an end user's client system.

E-mail servers should also check for invalid, corrupted, or malformed messages. An e-mail message with a corrupted *MIME header* can cause an unprepared e-mail server to crash or freeze. Thus, attackers can use invalid e-mail formats as a method of waging a DoS attack against your e-mail systems. By keeping e-mail servers properly updated and deploying antivirus scanners and e-mail filters, you can avoid most of the problems and attacks associated with e-mail.

Adware

Adware is a variation on the idea of spyware (discussed earlier). Adware displays pop-up advertisements to users based on their activities, URLs visited, applications accessed, etc. Adware is used to target advertisements to prospective customers. Unfortunately, most adware products arrive on client systems without the knowledge or consent of the user. Thus, legitimate commercial products are often seen as intrusive and abusive adware.

Countermeasures for adware are the same as for spyware and viruses, with the addition of specific spyware/adware-scanning tools.

Rootkits

A *rootkit* is a special type of hacker tool that embeds itself deep within an operating system. The rootkit positions itself at the heart of an operating system (OS) where it can manipulate information seen by the OS. Often, a rootkit replaces the OS kernel or shims itself under the kernel, so that whatever information it feeds or hides from the OS, the OS thinks is normal and acceptable. This allows a rootkit to hide itself from detection, prevent its files from being viewed by file management tools, and prevent its active processes from being viewed by task management or process management tools. Thus, a rootkit is a type of invisibility shield. A rootkit can be used to hide other malicious tools and/or perform other functions. A rootkit or other tools hidden by a rootkit could capture keystrokes, steal credentials, watch URLs, take screen captures, record sounds via the microphone, track application use, or grant a remote hacker back door access or remote control over the compromised target system.

After a rootkit has infected a system, that system can no longer be trusted or considered secure. There are rootkits that are still undetectable and/or cannot be effectively removed. Thus, any rootkit-compromised system can never be trusted again. To use a silly analogy: If you are fighting an invisible army, how can you be sure that you have defeated all of them?

There are several rootkit detection tools, some of which are able to remove some rootkits. However, once you know a rootkit is on a system, the only truly secure response is

reconstitution. Reconstitution is the action of performing a low-level formatting operation on all storage devices on that system, reinstalling the OS and all applications from trusted original sources, and then restoring files from trusted rootkit-free backups. Obviously, the best protection against rootkits is defense rather than response.

Botnets

The term *botnet* is a shortened form of the phrase *robot network*. It is used to describe a massive deployment of malicious code onto numerous compromised systems that are all controlled by a hacker. A botnet is the culmination of traditional DoS attacks into a concept known as a distributed denial-of-service (DDoS) attack. A DDoS attack occurs when a hacker has deposited remote-controlled agents, zombies, or bots onto numerous secondary victims, and then uses the deployed bots as a single entity to attack a primary target.

Botnets are either directly or indirectly controlled by a hacker. Sometimes the hacker is labeled as a bot herder, a master, or even a handler. Direct control of a botnet occurs when the bot herder sends commands to each bot. Therefore, bots have a listening service on an open port waiting for the communication from the bot herder. Indirect control of a botnet can occur through any intermediary communication system, including IRC, IM, FTP, e-mail, Web, blogging, Twitter, and so on. When indirect control is used, the bots listen on an intermediate communication service for messages from the master hacker.

Botnets are possible because most computers around the world are accessible over the Internet, and many of those computers are not fully secure. A botnet creator writes his botnet code to exploit a common vulnerability in order to spread the botnet agent far and wide—often using the same techniques used by viruses, worms, and Trojan horses. Botnets are typically comprised of thousands (if not hundreds of thousands) of compromised secondary victims. The secondary victims are the hosts of the botnet agent itself and are not affected or damaged beyond the initial intrusion and planting of the botnet agent. The hackers want the secondary victims fully functional so when they launch their botnet attack against the primary victim, they can use all the resources of the secondary victims against the primary target.

A botnet can be used to perform any type of malicious activity. Although they are most often used to perform DoS flooding attacks, botnets can also be used to transmit spam; perform massively distributed parallel processing to crack passwords or encryption keys; perform phishing attacks; capture network packets; or perform any other conceivable activity.

The best defense against a botnet is to keep your systems hardened and to not become the host of a botnet agent (in other words, don't become a secondary victim). Also, most antivirus software and anti-spyware/adware tools include well-known botnet agents in their detection databases.

If you are the primary victim of a botnet attack, there is little you can do to stop the attack. Your responses are often limited to disconnecting from the Internet, contacting your ISP, and reporting the incident to law enforcement.

Logic bomb

A *logic bomb* is a form of malicious code that remains dormant until a triggering event occurs. The triggering event can be a specific time and date, the launching of a specific program, or the accessing of a specific URL (such as your online banking logon page). Logic bombs can perform any malicious function the programmer wishes, from causing system crashes to deleting data to altering configurations to stealing authentication credentials.

Countermeasures for logic bombs are the same as for viruses.

Exam Essentials

Privilege escalation Privilege escalation is the theft of privileges or access to resources that a user is not authorized to possess.

Viruses Viruses are programs that are designed to spread from one system to another through self-replication and to perform any of a wide range of malicious activities.

Worms Worms are designed to exploit a single flaw in a system (operating system, protocol, service, or application) and then use that hole to replicate itself to other systems with the same flaw.

Trojan horses A Trojan horse is a form of malicious software that is disguised as something useful or legitimate.

Spyware and adware Spyware gathers information about users and may employ that information to target advertisements or steal identities. Adware gathers information about users and uses it to direct advertisements to the user. Both spyware and adware are usually unwanted software that gathers information without authorization.

Spam Spam is undesired or unsolicited e-mail. It's a problem for numerous reasons. First, spam can be the carrier for malicious code such as viruses, logic bombs, and Trojan horses. Second, spam can be the carrier of a social-engineering attack (hoax e-mail). Third, unwanted e-mail wastes your time while you're sorting through it looking for legitimate messages. Fourth, spam wastes Internet resources such as storage capacity, computing cycles, and throughput.

E-mail filters An e-mail filter is a list of e-mail addresses, domain names, or IP addresses where spam is known to originate.

Spoofed e-mail A spoofed e-mail is a message that has a fake or falsified source address. When an e-mail server receives an e-mail message, it should perform a reverse lookup on the source address of the message.

Hoaxes A hoax is an e-mail message that includes incorrect or misleading information. This is a written or static form of a social-engineering attack. Your primary weapons against hoaxes are education and awareness. Notify your network administrator when you receive a suspected hoax.

Rootkit A rootkit is a type of malicious code that fools the OS into thinking that active processes and files don't exist. Rootkits render a compromised system completely untrustworthy.

Botnet A botnet is a network of robots or malicious software agents controlled by a hacker in order to launch massive attacks against targets.

Logic bombs A logic bomb is a form of malicious code that remains dormant until a triggering event occurs. The triggering event can be a specific time and date, the launching of a specific program, or the accessing of a specific URL (such as your online banking login page).

Malicious code countermeasures The best countermeasure to viruses and other malicious code is an antivirus scanner that is updated regularly and which monitors all local storage devices, memory, and communication pathways for viral activities. Other countermeasures include avoiding downloading software from the Internet, not opening e-mail attachments, and avoiding the use of removable media from other environments.

1.2 Explain the security risks pertaining to system hardware and peripherals.

System hardware and peripherals require physical access controls and protections in order to maintain the logical security imposed by software. Without access control over the facility and physical environment, otherwise secured systems can be quickly compromised. Physical protections are used to protect against physical attacks, whereas logical protections protect only against logical attacks. Without adequate layers of protection, security is nonexistent. This section discusses several issues that often lead to security compromise because they are overlooked or deemed nonserious threats.



For more information on this topic, refer to Chapter 1 of the *CompTIA Security+ Study Guide, 4th Edition* (Sybex, November 2008).

BIOS

BIOS (basic input/output system) is the basic low-end firmware or software embedded onto the hardware's EEPROM (electrically erasable programmable read-only memory). BIOS identifies and initiates the basic system hardware components, such as the hard drive, optical drive, video card, and so on, so that the bootstrapping process of loading an OS can begin. This essential system function is a target of hackers and other intruders because it may provide an avenue of attack that is not secured or monitored.

BIOS attacks, as well as CMOS and device firmware attacks, are becoming a common target of physical hackers as well as of malicious code. If hackers or malware can alter the BIOS, CMOS, or firmware of a system, they may be able to bypass security features or initiate otherwise prohibited activities.

Protection against BIOS attacks requires physical access control to all hardware of a sensitive or valuable nature. Additionally, strong malware protection such as current antivirus software is important.

USB devices

USB devices are ubiquitous these days. Nearly every worker who uses a computer possesses a USB storage device, and most portable devices (such as phones, music players, and still or video cameras) connect via USB. However, this convenience comes at a cost to security. There are at least two main issues. First, just about any USB device can be used to either bring malicious code into or leak sensitive, confidential, and/or proprietary data out of an otherwise secure environment. Second, most computers built within the last three to five years have the ability to boot off USB. This could allow a user to boot a computer to an alternate OS (such as a live Linux distribution), which fully bypasses any security the native OS would have imposed.

To protect against USB threats, the only real option is to fully disallow use of all USB devices and lock down all USB ports. Otherwise, allowing the use of USB typically leaves your organization's system vulnerable to these threats.

Cell phones

Cell phones are an ever increasing security risk as they become more and more capable of interacting with the Internet as well as corporate networks. Cell phones often support memory cards, thus they could be used to smuggle malicious code in or confidential data out of organizations. Cell phones often contain sensitive data such as contacts, text messages, e-mail, and possibly notes and documents. The loss or theft of a cell phone could mean the compromise of personal and/or corporate secrets.

As cell phones become PDAs or even ultra-portable personal computing devices (miniature computers that are almost desktop replacements), they are becoming the target of hackers and malicious code. It is important to keep nonessential information off of portable devices, run a firewall and antivirus product (if available), and keep the system locked and/or encrypted (if possible).

Many cell phones also support USB connection to perform synchronization of communications and contacts with desktop and/or notebook computers as well as the transfer of files, documents, music, video, and so on. As discussed in the previous section, USB devices pose risks to malicious code and data theft.

Additionally, cell phones are not immune to eavesdropping. With the right type of sophisticated equipment, most cell phone conversations can be tapped into—not to mention the fact that anyone within 15 feet can hear you talking. Be careful what you discuss over a cell phone, especially when you are in a public place.



Please see Chapter 2, where additional cell phone issues of blue jacking and bluesnarfing are discussed.

Removable storage

Removable media drives, and removable storage in general, are considered both a convenience and a security vulnerability. The ability to add and remove storage media to a computer system makes it more versatile. However, it also makes it vulnerable to data theft and malicious code planting.

Removable media include the electronic, logical, or digital storage mechanisms listed in the following sections as well as printed materials. Any time media is no longer needed, it should be properly destroyed to prevent disclosure of sensitive and confidential information to unauthorized entities. For example, failing to destroy printouts or burned CDs may provide dumpster-diving attackers with treasures.

Tape is a form of removable media commonly used for backup purposes. It's a form of sequential storage, so data elements are written and read in sequential order rather than semi-randomly as with hard drives. Tape media often support larger storage capacities than most removable media, excluding hard drives. This makes them suited for backup operations.

The topic of CD-Rs (recordable compact disks) includes the wide range of optical media that can be written to. This includes CDR, CD-RW, DVD-R, and DVD-RW (plus numerous other variants). Writable CDs and DVDs are often inappropriate for network backups due to their size (a maximum of 650MB for CD-R/RW and 4GB or more for DVD-R/RW), but they're useful for personal (home) or client-level backups. However, the data on a CD isn't protected and thus is vulnerable to unauthorized access if you don't maintain physical control over the media.

Hard drives are usually thought of as the permanent internal storage devices of a computer. This is true, but hard drives are also available in removable formats. These include hard drives that are plugged into the case or attached by SCSI, eSATA, USB, or IEEE 1394 (FireWire) connections with their own external power-supply connections.

Diskettes, or floppies, are removable media that can store only a small amount of data (about 1.4MB). However, even though they're small, they represent a significant security threat to a protected environment if they get into the wrong hands.

A *flashcard*, or memory card, is a form of storage that uses EEPROM or NVRAM memory chips in a small form-factor case. Flashcards often use USB connectors or are themselves inserted into devices such as MP3 players and digital cameras. Some flashcards are almost as small as a quarter and are therefore easy to conceal.

Smartcards can be used for a wide variety of purposes. They can be used as an authentication factor (specifically, it is an example of a Type 2 authentication factor commonly known as “something you have”). When used as such, the smartcard hosts a memory chip that stores a password, PIN, certificate, private key, or digital signature. The authentication system uses this stored data item to verify a user’s identity. Smartcards are used as an authentication mechanism by networks, portable computers, PDAs, satellite phones, PKI devices, and more. A smartcard can even function as a credit card (like the American Express Blue card).

A smartcard can also be used as a storage device. Most smartcards have a very limited amount of storage, but sometimes being able to move a few kilobytes of data is all someone needs to steal something of great value. Account numbers, credit card numbers, or a user’s private key are all small items that can be very valuable.

Network attached storage

Network attached storage (NAS) is a storage system connected directly to a LAN in order to provide network file storage without the need for a dedicated file server. A NAS device is basically a hard drive storage bay that has just enough intelligence to share the drive with the network. NAS is used to quickly and easily expand the storage capacity of an organization without significant overhead expense of additional server hardware. However, NAS is not necessarily a secure option. Not all NAS solutions offer strong security such as user authentication or authorization.

If a rogue user gains access to a NAS, they may be able to access any or all of the organization’s confidential, private, and proprietary information. Additionally, such access may allow the rogue user to corrupt the data or plant malicious code.

Exam Essentials

BIOS BIOS (basic input/output system) is the basic low-end firmware or software embedded onto the hardware’s EEPROM used to initialize bootstrapping.

USB devices A USB device is any device that connects to a system via USB. USB devices may pose a security risk via information leakage, planting of malicious code, or allowing booting to alternate operating systems.

Removable media or storage Removable media include the electronic, logical, or digital storage mechanisms listed here as well as printed materials. Any time media is no longer needed, it should be properly destroyed to prevent disclosure of sensitive and confidential information to unauthorized entities.

1.3 Implement OS hardening practices and procedures to achieve workstation and server security.

It's important to realize that a key element in securing a system is to reduce its attack surface. The *attack surface* is the area that is exposed to untrusted networks or entities and that is vulnerable to attack. If a system is hosting numerous services and protocols, its attack surface is larger than that of a system running only essential services and protocols.



For more information on this topic, refer to Chapter 1 of the *CompTIA Security+ Study Guide, 4th Edition* (Sybex, November 2008).

It's tempting to install every service, component, application, and protocol available to you on every computer system you deploy. However, this temptation is in direct violation of a security best practice stating that you should have each system host only those services and protocols that are absolutely essential to its mission-critical operations.

The real issue is that software isn't trusted. Software (services, applications, components, and protocols) is written by people and therefore, in all likelihood, it isn't perfect. But even if software were without bugs, errors, oversights, mistakes, and so on, it would still represent a security risk. Software that is working as expected can often still be exploited by a malicious entity. Therefore, every instance of software deployed onto a computer system represents a collection of additional vulnerability points that may be exposed to external, untrusted, and possibly malicious entities.

From this perspective, you should understand that all nonessential software elements should be removed from a system before it's deployed on a network, especially if that network has Internet connectivity. But how do you know what is essential and what isn't? Here is a basic methodology:

1. Plan the purpose of the system.
2. Identify the services, applications, and protocols needed to support that purpose. Make sure these are installed on the system.
3. Identify the services, applications, and protocols that are already present on the system. Remove all that aren't needed.

Often, you won't know if a specific service that appears on a system by default is needed. Thus, a trial-and-error test is required. If software elements aren't clearly essential, disable them one by one and test the capabilities of the system. If the system performs as you expect, the software probably isn't needed. If the system doesn't perform as expected, then the software will need to be reenabled. This process is known as application and system hardening.

You may discover that some services and protocols offer features and capabilities that aren't necessary to the essential functions of your system. If so, find means to disable or restrict those characteristics. This may include restricting ports or reconfiguring services through a management console.

The essential services on a system are usually easy to identify—they generally have recognizable names that correspond to the function of the server. However, you must determine which services are essential on your specific system. Services that are essential on a web server may not be essential on a file server or an e-mail server. Some examples of possible essential services include the following:

- File sharing
- E-mail
- Web
- File Transfer Protocol (FTP)
- Telnet
- Remote access
- Network News Transfer Protocol (NNTP)
- Domain Name Service (DNS)
- Dynamic Host Configuration Protocol (DHCP)

Nonessential services are more difficult to identify. Just because a service doesn't have the same name as an essential function of your server doesn't mean that it isn't used by the underlying operating system or as a support service. It's extremely important to test and verify whether any service is being depended on by an essential service. However, several services are common candidates for nonessential services that you may want to locate and disable first (assuming you follow the testing method described earlier). These may include the following:

- NetBIOS
- Unix RPC
- Network File System (NFS)
- X services
- R services
- Trivial File Transfer Protocol (TFTP)
- NetMeeting
- Instant messaging
- Remote-control software
- Simple Network Management Protocol (SNMP)

Operating system hardening is the process of reducing vulnerabilities, managing risk, and improving the security provided by or for an operating system. This is usually accomplished by taking advantage of an operating system's native security features and supplementing them with add-on applications, such as firewalls, antivirus software, and malicious-code scanners.

Hardening an operating system includes protecting the system from both intentional directed attacks and unintentional or accidental damage. This can include implementing security countermeasures as well as fault-tolerant solutions for both hardware and software. Some of the actions that are often included in a system hardening procedure include the following:

- Deploy the latest version of the operating system.
- Apply any service packs or updates to the operating system.
- Update the versions of all device drivers.
- Verify that all remote-management or remote-connectivity solutions that are active are secure. Avoid FTP, Telnet, and other clear text or weak authentication protocols.
- Disable all unnecessary services, protocols, and applications.
- Remove or securely configure SNMP.
- Synchronize time zones and clocks across the network with an Internet time server.
- Configure event viewer log settings to maximize capture and storage of audit events.
- Rename default accounts.
- Enforce strong passwords on all accounts.
- Force password changes on a periodic basis.
- Restrict access to administrative groups and accounts.
- Hide the last-logged-on user's account name.
- Enforce account lockout.
- Configure a legal warning message that's displayed at logon.
- If file sharing is used, force the use of secure sharing protocols or use virtual private networks (VPNs).
- Use a security and vulnerability scanner against the system.
- Scan for open ports.
- Disable *Internet Control Message Protocol* (ICMP) functionality on publicly accessible systems.
- Consider disabling NetBIOS.
- Configure auditing.
- Configure backups.

The file system in use on a system greatly affects the security offered by that system. A file system that incorporates security, such as access control and auditing, is a more secure choice than a file system without incorporated security. One great example of a secured file system is the Microsoft New Technology File System (NTFS). NTFS was first deployed under Windows NT, but it's now found in Windows 2000, Windows XP, Windows Server 2003, and Windows Vista. It offers file- and folder-level access permissions and auditing capabilities. Examples of file systems that do *not* include security are FAT (file allocation table) and FAT32.

Workstations are the computer systems that people use to interact with a network. Workstations are also called *clients*, *terminals*, or *end-user computers*. Access to workstations should be restricted to authorized personnel. One method to accomplish this is to use strong authentication, such as two-factor authentication with a smartcard and a password or PIN.

Servers are the computer systems on a network that support and maintain the network. Servers provide services or share resources with the network. They require greater physical and logical security protections than workstations because they represent a concentration of assets, value, and capabilities. End users should be restricted from physically accessing servers, and they should have no reason to log on directly to a server—they should interact with servers over a network through their workstations.

Hotfixes

A *hotfix* is often a single-issue update (however, there are some multi-issue hotfixes) that corrects a single problem. Hotfixes aren't as thoroughly tested as other updates—they're quickly designed and released to deal with immediate issues and problems. You should install them if you're experiencing the problem they're designed to correct or you're threatened by the vulnerability they're designed to address.

Service packs

Service packs are collections of hotfixes and other previously unreleased updates and features as a single entity. They're thoroughly tested and generally should be applied to all systems once they're made available. Service packs are cumulative, so you only need to apply the most recent service pack to keep your systems current.

Patches

A *patch* is an update that corrects programming flaws that cause security vulnerabilities. Patches are single-issue utilities that are more thoroughly tested than hotfixes.

Patch management

Security is always a moving target. A system that is secure today may be vulnerable tomorrow. New methods of attacks, new attack tools, new viruses, new weaknesses, accidents in your environment, and much more can cause new risks, threats, and vulnerabilities at any time. Staying vigilant against new security issues is essential in today's business environment. One method to stay as secure as possible is to install updates from vendors.

Vendor *updates* to operating systems, applications, services, protocols, device drivers, and any other software are the absolute best way to protect your environment from known attacks and vulnerabilities. Not all vendor updates are security related, but any error, bug, or flaw that can be exploited to result in damaged data, disclosure of information, or obstructed access to resources should be addressed.

The best procedure to keep your systems updated is through a good patch management system that includes the following steps:

1. Watch vendor websites for information about updates.
2. Sign up for newsletters, discussion groups, or notifications.
3. Download all updates as they're made available.
4. Test all updates on nonproduction systems.
5. Document changes to your test systems and plan the implementation on production systems.
6. Back up production systems before implementing updates.
7. Implement updates on production systems.
8. Evaluate the effect of the updates on the production systems.
9. If negative effects are discovered, roll back the update.

Patch management can be implemented via a manual process, or an intelligent software tool can be used to automate this essential activity. Although security is not just patch management, security management requires that patches and updates are properly installed.

Group policies

A group policy is a mechanism of Microsoft Active Directory domain networks that is used to distribute and apply configuration settings to servers and clients. Group policies, commonly known as group policy objects or GPOs, are a collection of configuration settings that are applied to systems upon bootup and/or upon user login. There are settings for function, environment, network, performance, auditing, and security. There are settings for the computer and operating system as well as settings only for specific user accounts.

GPOs are a common mechanism used to manage and maintain the configuration and security of a large Windows-based network environment. New security requirements can be easily distributed throughout the network using GPO.

Security templates

A security template is a set of security settings that can be mechanically applied to a computer to establish a specific configuration. Security templates can be used to establish baselines or bring a system up to compliance with a security policy. Security templates can be custom designed for workstations and server function/task/purpose. Security templates are a generic concept; however, there are specific security templates that can be applied via Windows's group policy system.

Security templates can be built by hand or by extracting settings from a preconfigured master. Once a security template exists, it can be used to configure a new or existing machine (by applying the template to the target either manually or through a GPO), or it can be used to compare the current configuration to the desired configuration. This latter process is known as *security template analysis* and often results in a report detailing the gaps in compliance.

Configuration baselines

One mechanism often used to help maintain a hardened system is to use a security baseline. A *security baseline* is a standardized minimal level of security that all systems in an organization must comply with. This lowest common denominator establishes a firm and reliable security structure on which to build trust and assurance. The security baseline is defined by the organization's security policy. It may include requirements of specific hardware components, operating system versions, service packs, patches/upgrades, configuration settings, add-on applications, service settings, and more.

The basic procedure for establishing a security baseline or hardening a system is as follows:

1. Remove unneeded components, such as protocols, applications, services, and hardware (including device drivers).
2. Update and patch the operating system and all installed applications, services, and protocols.
3. Configure all installed software as securely as possible.
4. Impose restrictions on information distribution for the system, its active services, and its hosted resources.

Documentation is an important aspect of establishing a security baseline and implementing security in an environment. Every aspect of a system, from design to implementation, tuning, and securing, should be documented. Failing to have sufficient documentation is often the primary cause of difficulty in locking down or securing a server. Without proper documentation, all the details about the operating system, hardware configuration, applications, services, updates, patches, configuration, and so on must be discovered before security improvements can be implemented. With proper documentation, a security professional can quickly add to the existing security without having to reexamine the entire environment.

Creating or defining a baseline requires that you examine three key areas of an environment: the operating system, the network, and the applications. The following sections examine issues related to security baseline establishment for these areas.

Exam Essentials

Hotfixes, service packs, patches, and updates Hotfixes, service packs, patches, and updates are improvements to distributed products produced by vendors to improve or fix issues. To maintain security, all hardware and software should be kept current with vendor-released patches.

Security risks of nonessential software Nonessential software increases the attack surface of your systems. Removing every element of software that isn't required will improve the security of a system.

Security baselines A security baseline is a standardized minimal level of security that all systems in an organization must comply with. This lowest common denominator establishes a firm and reliable security structure upon which to build trust and assurance. The security baseline is defined by the organization's security policy.

Documentation Documentation is an important aspect of establishing a security baseline and implementing security in an environment. Every aspect of a system, from design through implementation, tuning, and securing, should be documented. Failing to have sufficient documentation is often the primary cause of difficulty in locking down or securing a server.

Operating system hardening Operating system hardening is the process of reducing vulnerabilities, managing risk, and improving the security provided by or for an operating system. This is usually accomplished by taking advantage of the native security features of an operating system and supplementing them with add-on applications, such as firewalls, antivirus software, and malicious-code scanners.

File system security The file system in use on a system greatly affects the security offered by that system. A file system that incorporates security, such as access control and auditing, is a more secure choice than a file system without incorporated security.

1.4 Carry out the appropriate procedures to establish application security.

Application hardening is the task of imposing security on required applications and services. This usually involves tuning and configuring the native security features of the installed software and installing supportive security applications as needed. When you're developing new applications in house, it's also important to include security design, implementation, and integration throughout the development process.



For more information on this topic, refer to Chapter 1 of the *CompTIA Security+ Study Guide, 4th Edition* (Sybex, November 2008).

Application hardening is often seen as a subelement of operating system hardening. In fact, many of the same steps and procedures used to lock down an operating system are used to harden an application or service. In addition to the general notion of disabling any unneeded protocols and services, you should also disable any unneeded features, functions, or capabilities of a service or protocol based on the server's role and the capabilities your organization needs.

ActiveX

ActiveX is a mobile code technology developed by Microsoft. ActiveX controls or components are stand-alone programs that can be attached to or embedded in web documents to perform a wide range of functions. ActiveX components are executed with the same security privileges as the current user. However, the ActiveX component is saved to the hard drive and can be accessed at a later time, even after you've left the website where it was obtained. Thus, ActiveX can be a significant security issue. It's recommended that ActiveX components not be downloaded or executed except from websites you're sure you can trust. In many cases, web browsers can be configured to always allow, always block, or prompt the user each time an ActiveX component is presented by a website.

One method that helps reduce the number of malicious elements of mobile code downloaded and executed on a system is a mechanism known as applet signing. A *signed applet* is a piece of mobile code that has been digitally signed using the creator's or owner's certificate. A signed applet clearly indicates to the user the source of the applet. Web browsers can be configured to accept only signed applets or to prompt whenever any applet is offered by a website. It's important to understand that a signed applet only proves the applet's identity or source; it provides no guarantee as to the reliability or quality of the applet. Just because you know where an applet comes from doesn't mean that the applet isn't malicious code or that it won't cause a problem with your computer system.

Java

Java is a programming language created by Sun Microsystems specifically to operate in the distributed environment of the Internet. Java is a write-once, run-anywhere solution that allows a programmer to write an applet once and have it run on any platform. There is a requirement that a Java Virtual Machine (JVM) be installed on every system, but since there is a JVM for every commercial/public OS, this is usually not an issue.

Java is designed to operate within a restricted memory space and execution container known as a sandbox. This limits the functions of Java, which makes it harder to craft malicious code in Java. Java applets are not saved to the hard drive and do not fully take on the access privileges of the user. Thus, Java is designed as a more secure mobile code system. However, there are malicious Java applets and many of them perform social-engineering-based attacks rather than direct code-based attacks.

Java should only be allowed to execute if it is from a known trusted source.

Scripting

Scripting is a type of programming where the lines of the program or script remain in their original human readable form and which does not need to be precompiled to execute. Instead, the OS or a scripting interpreter (such as Perl) will perform JIT (just in time) compiling as needed. Scripting is a powerful and flexible tool used by a wide number of tech-savvy administrators. However, scripting can also be employed for malicious purposes, so it is always a good idea to inspect and/or scan all scripts for malicious code before use.

JavaScript is a scripting programming language that can be embedded directly into the HTML of a web page. It's executed by the web browser and can be used to perform a wide range of functions, both benign and malicious. Unlike its namesake Java, JavaScript doesn't run in a restricted security zone; rather, it has nearly unrestricted access to all system resources. Due to its power and unpredictability, it's generally recommended that you disable the download and execution of JavaScript except from websites you're sure you can trust.

Common Gateway Interface (CGI) is a mechanism developed early in the life of the Web to allow a web browser to submit information from a user back to a web server for processing by a server-side script or application. Although CGI scripting isn't used as widely as JavaScript or ActiveX today, it's still a default-enabled feature of both web servers and web browsers.

CGI has numerous vulnerabilities, the most prevalent of which is the ability of a user to submit data that results in the script's failure, a buffer overflow attack, or the performance of an unauthorized or unwanted activity (at least from the perspective of the web server owner).

Another issue with CGI is that all directories on the web server that host CGI scripts must be configured to allow web visitors to execute files. If the web server's folder hierarchy isn't properly protected, this could enable a user to upload their own scripts and then execute them.

To protect against CGI vulnerabilities, either don't use it or properly lock down the web folders appropriately. Also, thoroughly inspect all CGI scripts for malicious code before use. When possible, write your own scripts rather than use someone else's.

Browser

A browser (or web browser) is the client software used to interact with the Web. Browsers are usually graphical in nature, but there are several popular text-only or command-line browsers. Browsers grant users access to the vast repositories of information and resources distributed around the globe via the Internet's World Wide Web. Unfortunately, browsers bring security risks to their host computers.

A browser can be used to access malicious websites, which can result in information leakage, identity theft, or downloading of malicious code. It is important to practice safe surfing habits by being cautious and not downloading content from sources you don't fully trust.

Browsers should be kept current with patches and updates from the vendor. The OS should be hardened, and both a firewall and antivirus software should be running and kept updated. Block access to all known risky or malicious sites. Only accept downloadable code from known trusted and signed sources.

Buffer overflows

Software exploitation attacks are directed toward known flaws, bugs, errors, oversights, or normal functions of the operating system, protocols, services, or installed applications. One of the most common forms of software exploitation is a buffer overflow attack.

A buffer overflow attack occurs when an attacker submits data to a process that is larger than the input variable is able to contain. Unless the program is properly coded to handle excess input, the extra data is dropped into the system's execution stack and may execute as a fully privileged operation. Buffer overflow attacks can result in system crashes, corrupted data, user privilege escalation, or just about anything a hacker can think of. The only countermeasures to buffer overflow attacks are to patch the software when issues are discovered and to properly code software to perform input validation checks before accepting input for processing.

Once a weakness is discovered in software, a hacker often writes an exploit or attack tool. These tools are easily accessible on the Internet. They allow anyone to grab the tool and point it at a victim to perform the attack, even when the attacker has no knowledge of how to perform the attack.

A *buffer overflow* occurs when a program receives input that is larger than it was designed to accept or process. The extra data received by the program is shunted over onto the CPU without any security restrictions; it's then allowed to execute (assuming it's a valid command, script, system call, and so on) with system-level privileges. There are many possible results of a buffer overflow, including a program crash, a system freeze or crash, opening a port, disabling a service, creating a user account, elevating the privileges of an existing user account, accessing a website, or executing a utility. Clever attackers can do just about anything they wish if they can execute a command or script with unrestricted access to a system.

Sometimes a buffer overflow attack can be labeled as a form of DoS attack, since a buffer overflow occurs when a system receives more data than it can handle (a bit like a flooding attack). This is especially true when the buffer overflow event results in a system no longer being able to process legitimate data or requests.

Poor programming quality controls and not including input validation checks in software lead to buffer overflow attacks. Unfortunately, there is little commercial software that isn't vulnerable to buffer overflow attacks; web server software is attacked most frequently. Fortunately, buffer overflow vulnerabilities are often easily patched with vendor updates.

Cookies

A *cookie* is a tracking mechanism developed for web servers to monitor and respond to a user's serial viewing of multiple web pages. A cookie is often used to maintain an

e-commerce shopping cart, focus product placement, or track your visiting habits. However, the benign purposes of cookies have been subverted by malevolent entities. Now cookies are a common means of violating your privacy by gathering information about your identity, logon credentials, surfing habits, work habits, and much more. A cookie can be easily exploited against a web browser to gather sufficient information about a user to allow the attacker to impersonate the victim online. It's generally recommended that you block third-party cookies from everyone and first-party cookies from all but the most trusted sites. Trusted sites are usually those entities that protect your identity by not including such details in a cookie. Instead, these sites only place a session ID in the cookie and thus keep all of your personal information in a backside database. Without allowing trusted first-party cookies (a.k.a. session cookies), functions such as e-commerce shopping carts, online banking, and even posting to discussion forums would be disabled.

SMTP open relays

E-mail is the most widely used communication vehicle on the Internet. However, it has also become one of the primary delivery mechanisms for malicious code and social-engineering attacks. Understanding e-mail security issues is essential to the Security+ exam.

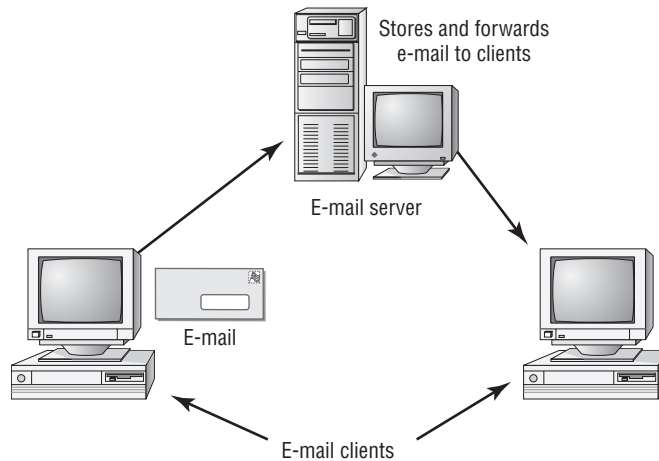
Internet-based e-mail relies primarily on a single protocol: *Simple Mail Transport Protocol (SMTP)*. As you can see in Figure 1.1, SMTP has proven itself over more than 20 years as a reliable e-mail delivery system. However, it has one significant flaw: its nearly complete lack of security. SMTP doesn't offer encryption for transmitted messages. Thus, any and all e-mail can be snooped and examined. As e-mail becomes the basis for business communications, cleartext communication is no longer a viable option. Fortunately, SMTP supports add-on capabilities that bring encryption and other security services to e-mail. SMTP operates over TCP port 25.

Other protocols are involved with a complete e-mail solution. *Post Office Protocol (POP3)* and *Internet Message Access Protocol (IMAP)* are used to pull e-mail from an e-mail server down to a client, but they aren't involved in moving e-mail across the Internet. POP3 operates over TCP port 110, and IMAP operates over TCP port 143. Because e-mail is natively insecure, several encryption options have been developed to add security to e-mail used over the Internet. Two of the most common solutions are Secure Multipurpose Internet Mail Extensions (S/MIME) and Pretty Good Privacy (PGP).

S/MIME is an Internet standard for encrypting and digitally signing e-mail. It uses RSA (an asymmetric encryption scheme) to encrypt and protect e-mail.

PGP is used to encrypt and digitally sign e-mail messages. It uses RSA or Diffie-Hellman asymmetric cryptography solutions.

SMTP relay is the feature or function of an SMTP or e-mail server when it receives an e-mail from a client or other SMTP server and then forwards it on to another SMTP server. The forwarding action is similar to normal network packet routing, in that the purpose is to transmit the e-mail message to its destination. E-mail relaying is an essential part of the success of Internet e-mail, but it can be abused.

FIGURE 1.1 E-mail connections between clients and a server

E-mail relaying abuse often leads to DoS attacks or spam. Spam (specifically, spoofed spam) is often sent by rogue solicitors who find e-mail servers on the Internet that aren't properly configured to authenticate clients and servers before accepting e-mail. When an SMTP server fails to authenticate clients and servers before accepting e-mail, it's known as an *open relay*.

When you're deploying and securing an SMTP server, it's important to disable the open relay feature. However, even after an SMTP server has been secured, a clever attacker may be able to reenabling relaying using a Trojan horse, buffer overflow, or remote access attack against the system. Thus, it's important to regularly check the performance logs and configuration settings of e-mail servers. One important item to look for is the presence of a universal acceptance configuration that would allow for the unrestricted and unverified relaying of e-mail. For example, on Unix systems, the e-mail server will have a list of domains that are authorized to submit e-mail. If this file includes a universal include coding statement, which is usually represented by a dot and an asterisk (.*), delete that entry and reboot the system to force the change to take effect.

Instant messaging

Instant messaging (IM) is a mechanism that allows for real-time text-based chat between two users located anywhere on the Internet. Some IM utilities allow for file transfer, multimedia, voice and video conferencing, and more. However, unlike many Internet information services, such as the Web, FTP, and e-mail, IM is a peer-to-peer service. There is no need for a centralized controlling server. This makes IM easy for end users to deploy and use, but it's difficult to manage from a corporate perspective. IM is insecure. It has numerous vulnerabilities, such as susceptibility to packet sniffing, it lacks true native security capabilities, and it provides no protection for privacy.

Instant messaging offers little in the way of security or privacy. Many IM clients are susceptible to malicious code deposit or infection through their file transfer capabilities. Also, IM users are often subject to numerous forms of social-engineering attacks, such as impersonation or convincing a victim to reveal information that should remain confidential (such as passwords).

Most IM clients don't use encryption when transmitting messages. Thus, most communications are subject to *packet sniffing* and *eavesdropping* attacks.

IM clients do not provide any direct protection for user privacy. The fact that IM clients perform communications in the clear and do not offer encryption or security services makes all private and confidential transmissions subject to packet sniffing and eavesdropping attacks.

P2P

P2P (peer to peer) is a file-sharing system that allows for the decentralized distribution of files over the Internet. P2P can be used to distribute legitimate free, open-source, or public content. However, P2P is often used to distribute illegal content such as pirated music, software, and movies. There are many varieties of P2P mechanisms; one of the more prevalent systems is known as BitTorrent.

Because P2P is a file distribution system, common precautions should be taken. First, try not to download files from untrusted sources. Second, scan every file with an antivirus scanner before use. Third, don't access or distribute content that is not free and legal to distribute. Even with these precautions, P2P can be a bandwidth hog and there is always the possibility that the P2P software can involve your system in distributing illegal content even without your direct permission. Because of these issues, organizations often ban P2P as a standard policy.

Input validation

Input validation is an aspect of defensive programming intended to ward off a wide range of input-focused attacks, such as buffer overflows and fuzzing. A fuzzing attack occurs when an attacker sends a variety of types and sizes of content to input points to see if they can trigger an abnormal response. Input validation checks each and every input received before it is allowed to be processed. The check could be a length, character type, language type, domain, or even timing check to prevent unknown, unwanted, or unexpected content to make it to the core program.

Cross-site scripting (XSS)

Cross-site scripting (XSS) is a form of malicious code injection attack where an attacker is able to compromise a web server and inject their own malicious code into the content sent to other visitors. Hackers have discovered numerous and ingenious methods to inject their own malicious code into websites via CGI scripts, web server software vulnerabilities, SQL

injection attacks, frame exploitation, DNS redirects, cookie hijacks, and many other forms of attack. A successful XSS attack could result in identity theft, credential theft, data theft, financial losses, or planting of remote control software on visiting clients.

Defenses against XSS include maintaining a patched web server, using firewalls, and auditing for suspicious activity. As a web user, you can defend against XSS by keeping your system patched, running antivirus software, and avoiding nonmainstream websites.

Exam Essentials

ActiveX ActiveX is a mobile code framework from Microsoft. ActiveX controls take on the privileges of the user. Only allow signed applets from sites you trust.

Java Java is a mobile code language from Sun Microsystems. Java was designed as a secure solution for the distributed Internet. Java runs in a sandbox.

JavaScript JavaScript is a scripting programming language that can be imbedded directly into the HTML of a web page. JavaScript, unlike its namesake Java, doesn't run in a restricted security zone; rather, it has nearly unrestricted access to all system resources. Due to its power and unpredictability, it's generally recommended that you disable the download and execution of JavaScript except from those websites you're sure you can trust.

Signed applets A signed applet is a piece of mobile code that has been digitally signed using the creator's or owner's certificate. A signed applet clearly indicates to the user the source of the applet.

Buffer overflows Buffer overflows occur due to a lack of secure defensive programming. The exploitation of a buffer overflow can result in a system crash or arbitrary code execution. A buffer overflow occurs when a program receives input that is larger than it was designed to accept or process. The extra data received by the program is shunted over onto the CPU without any security restrictions; it's then allowed to execute. Results of buffer overflows can include a program crash, a system freeze or crash, opening a port, disabling a service, creating a user account, elevating the privileges of an existing user account, accessing a website, or executing a utility.

Cookies A cookie is a tracking mechanism developed for web servers to monitor and respond to a user's serial viewing of multiple web pages. A cookie may allow identity theft.

SMTP open relay An SMTP open relay can be abused to send spam, hoaxes, or malicious attachments.

SMTP Simple Mail Transport Protocol (SMTP) moves e-mail messages across the Internet from sender to recipient. It doesn't include native encryption. It operates over TCP port 25.

POP3 and IMAP Post Office Protocol (POP3) and Internet Mail Access Protocol (IMAP) are used to pull e-mail from an e-mail server down to a client. POP3 operates over TCP port 110. IMAP operates over TCP port 143.

XSS Cross-site scripting (XSS) is a form of malicious code injection attack where an attacker is able to compromise a web server and inject their own malicious code into the content sent to other visitors.

Application hardening Application hardening is the task of imposing security on required applications and services. This usually involves tuning and configuring the native security features of the installed software and installing supportive security applications as needed. When you're developing new applications in house, it's also important to include security design, implementation, and integration throughout the development process.

1.5 Implement security applications.

Imposing security is often a long and complex process. One important part of this process is the installation or implementation of security applications. A security application is software designed specifically to perform a set of security functions. The following sections discuss some common security applications.



For more information on this topic, refer to Chapter 1 of the *CompTIA Security+ Study Guide, 4th Edition* (Sybex, November 2008).

HIDS

A host-based intrusion detection system (HIDS) is a security application designed to monitor or watch the local computer system for malicious or anomalous activity. (IDS is discussed in detail in Chapter 2.) HIDS is different from a network-based IDS in that it is only able to monitor or oversee the activities within a single computer, rather than monitoring the activity across an entire network. An HIDS can be installed on a client or a server system. Because it is an active process, an HIDS does consume some system resources. An HIDS is dependent upon the auditing and logging systems of the host OS. It is effective at detecting known attacks against the local system.

Common examples of HIDS are antivirus software, anti-spyware scanners, and security anomaly detectors.

Personal software firewalls

A personal software firewall is a security application that is installed on client systems. A client firewall is used to provide protection for the client system from the activities of the user and from communications from the network or Internet. A personal firewall must

be kept current with patches and updates. It is often able to limit communications to only approved applications and protocols and usually can prevent external initiations of communications.

Antivirus

Antivirus software is an essential security application. Antivirus software is one example of a host IDS. It monitors the local system for evidence of malware in memory, in active processes, and in storage. Most antivirus products can remove detected malicious code and repair any damage caused by such malicious code. In order for antivirus software to be effective, it must be kept current with daily signature database updates. It is also important to use the most recent engine as new methods of detection and removal are only found in the most current versions of antivirus software.

Anti-spam

Anti-spam is a variation on the theme of antivirus software. Anti-spam specifically monitors e-mail communications for spam and other forms of unwanted e-mail in order to stop hoaxes, identity theft, waste of resources, and possible distribution of malicious software. Some antivirus software products include an anti-spam component.

Popup blockers

Popup blockers are used to prevent websites from opening additional web browser windows without your consent. Often these popup windows are used for advertisements or possibly to distribute malicious code or interact with questionable content. Popup blockers simply prevent active web browser processes or code from websites from launching or initiating new windows. There is usually an easy bypass for those times when you want to allow popups. One common bypass is to hold down the Ctrl key while the popup opens. Popup blockers are common components of modern web browsers, but they may also be part of antivirus software or stand-alone third-party applications.

Exam Essentials

HIDS A host-based IDS is used to protect the local client, user, and network from various malicious events.

1.6 Explain the purpose and application of virtualization technology.

Virtualization technology is used to host one or more operating systems within the memory of a single host computer. This mechanism allows virtually any OS to operate on any hardware. It also allows multiple operating systems to work simultaneously on the same hardware. Common examples include VMWare, Microsoft's Virtual PC, Microsoft Virtual Server 2005, Hyper-V with Windows Server 2008, and Apple's Bootcamp.



For more information on this topic, refer to Chapter 1 of the *CompTIA Security+ Study Guide, 4th Edition* (Sybex, November 2008).

Virtualization has several benefits, such as being able to launch individual instances of servers or services as needed, real-time scalability, and being able to run the exact needed OS version for the needed application. Virtualized servers and services are indistinguishable from traditional servers and services from a user's perspective. Additionally, recovery from damaged, crashed, or corrupted virtual systems is often quick: Simply replace the virtual system's main hard drive file with a clean backup version, and then relaunch it.

Exam Essentials

Virtualization technology Virtualization technology is used to host one or more operating systems within the memory of a single host computer.

Review Questions

1. Which of the following is commonly found to be a nonessential service on a web server?
 - A. Server service
 - B. DNS service
 - C. FTP service
 - D. Print spooler service
2. Which is the best countermeasure against malicious code?
 - A. Manage user behavior
 - B. Prevent reuse of external removable media
 - C. Use antivirus software
 - D. Disable mobile code on web browsers
3. What TCP port does IMAP function over?
 - A. 143
 - B. 25
 - C. 110
 - D. 443
4. What is the first action you should take when you receive an e-mail that describes a removal process to clean your system from a rapidly spreading high-risk virus?
 - A. Follow the instructions immediately
 - B. Forward the message to everyone you know
 - C. Open any attachment enclosed with the message
 - D. Report the message to your network administrator
5. Which of the following threats is eliminated when only signed applets are allowed to download through a web browser?
 - A. CGI
 - B. ActiveX
 - C. Cookies
 - D. Instant messaging
6. Which of the following actions in operating system hardening should come earliest in the process?
 - A. Enable secure remote administration
 - B. Remove unneeded services and protocols
 - C. Enable logging and auditing
 - D. Connect the system to the network/Internet

7. Illegal or unauthorized zone transfers are a significant and direct threat to what type of network server?
 - A. Web
 - B. DHCP
 - C. DNS
 - D. Database
8. What type of virus is able to regenerate itself if a single element of its infection is not removed from a compromised system?
 - A. Polymorphic
 - B. Armored
 - C. Retro
 - D. Phage
9. A rootkit has been discovered on your mission-critical database server. What is the best step to take to return this system to production?
 - A. Reconstitution
 - B. Run an antivirus tool
 - C. Install an HIDS
 - D. Apply vendor patches
10. A security template can be used to perform all but which of the following tasks?
 - A. Capture the security configuration of a master system
 - B. Apply security settings to a target system
 - C. Return a target system to its precompromised state
 - D. Evaluate compliance with security of a target system

Answers to Review Questions

1. D. A nonessential service is any element that isn't needed by the primary function of the server. In most cases, a web server doesn't use the print spooler service, but it often uses the server, DNS, and FTP services.
2. C. The most reliable countermeasure against malicious code is an antivirus scanner. User-behavior modification, managing media, and disabling mobile code are all countermeasures against malicious code, but they aren't as reliable and effective as antivirus scanners.
3. A. IMAP functions over TCP port 143. SMTP functions over TCP port 25. POP3 functions over TCP port 110. SSL and TLS function over TCP port 443.
4. D. This e-mail is likely a hoax. When you receive an e-mail hoax, the first step is to inform your network administrator. Don't follow its directions, forward it to others, or open any attachments.
5. B. If only signed applets are allowed to download through a web browser, you gain protection from unknown sourced ActiveX components. Applet signing doesn't affect CGI, cookies, or IM.
6. B. Removing unneeded services and protocols is an operating system hardening step that should come before any of the other three.
7. C. Illegal or unauthorized zone transfers are a significant and direct threat to DNS servers.
8. D. A phage virus is able to regenerate itself from any of its remaining parts.
9. A. The only real option to return a system to a secure state after a rootkit is reconstitution.
10. C. A security template alone cannot return a system to its precompromised state.