Domain 7
Malicious Code

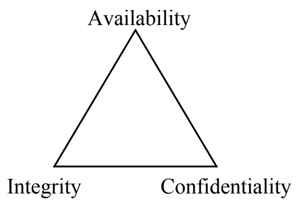
Diana-Lynn Contesti, CISSP

Introduction

The malicious code domain addresses computer software/code that can be described as being malicious or destructive to the computing environment. This includes the virus, worm, logic bomb, Trojan horse, and other related code that exhibits deviant behavior.

Information Protection Requirements

The A-I-C Triad This chapter discusses techniques that can be used to help address issues associated with Availability as they relate to the A-I-C triad (see below). The goal is to ensure that information and computer systems are available for use when required. Malicious code (in any form) can make either a system unusable, data unavailable, or both. Viruses can contain a payload that is able to compromise both the confidentiality and integrity of files on an infected computer system.



A History of Computer Viruses

The history of computer viruses shows that since the late 1980s the number of computer viruses has grown exponentially and that their behaviors have changed over time. Figure 7.1, shows the growth in numbers of computer viruses. At the end of 2003, it was estimated that 1,200 viruses were being created each month.

In 1949, John von Neumann wrote a paper titled “Theory and Organization of Complicated Automata” for the Institute for Advanced Study in Princeton, New Jersey. In the paper, he theorized that it was possible for a computer

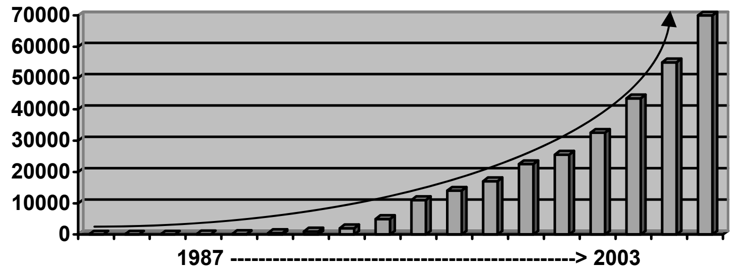


Figure 7.1 The exponential growth of viruses

program to replicate. Included in the paper was a model for what we call a computer virus.

Early in the 1950s, Bell Laboratories (McIlroy, Vysottsky, and Morris), developed a game that would test von Neumann's theory. The intent of game was to create programs which would attack, erase, and propagate on an opponent's system.

Author John Brunner published *The Shockwave Rider* in 1975. This book is often referred to as the cyberpunk novel as Brunner envisioned computer worms as they autonomously move from host to host, rather than attaching their code to other programs in the manner of viruses. This work documented the basic concept of computer programs that could self-replicate. It was not until 1984, that Dr. Fred Cohen's "Computer Viruses—Theory and Experiments" defined the computer virus and described experiments he and others performed to prove the viability of viral code.

The first computer virus to pass (or replicate) from PC to PC was Brain, although viruses had been passing between other platforms (VAXs and Apple IIs for some time). Brain was a boot sector virus that would stealthily leave contact information for Brain Computer Services (making it the only known virus to contain the creators' real names, addresses, and phone numbers). Stealth is a technology that enables a virus to actively hide itself from anti-virus software by either masking the size of the file that it hides in or temporarily removing itself from the infected file and placing a copy of itself in another location on the drive, replacing the infected file with an uninfected one that it has stored on the hard drive. The owners of Brain Computer Services, Basit and Amjaad (software vendors in Pakistan) claimed that they wrote the code to prevent their software from being pirated in Pakistan. The virus would just put a copyright in the directory of the program. Also, this virus could not affect hard disks; it could only affect floppy disks. However, Brain leaked through the Pakistani borders and infected computers worldwide

The only mainframe worm was the Christmas tree worm and it was a combination of a Trojan and a chain letter. It successfully managed to paralyze the IBM network on Christmas day, 1987.

Written in a language called Exec, it asked the user to type the word "Christmas" on the screen. Then it drew a Christmas tree and sent itself to all the names of people stored in the user files "Names" and "Netlog" and in this way propagating itself.

In the late 1980s, the Internet was a network that primarily connected university computers to each other. This network was vulnerable to programs, which could propagate using existing communications protocols. A university student named Robert Morris, who unleashed a major malware[1] incident, the Morris Worm, in November 1988, demonstrated this. This UNIX-based worm overwhelmed approximately 10 percent of all DEC computers on the Internet, causing a lot of media interest and many headlines.

Through the 1990s viruses began to change in their form and behaviors. Virus writers began to encrypt viruses and cause them to change their form as they spread to avoid detection (a trait that would later be coined as polymorphic). The decrypted version was stored at the beginning of the virus. This made it more difficult to determine if something was a virus as the virus signature was moved or changed. To detect a virus, it was necessary to write an algorithm that applied logical tests to the file, and decide whether the bytes it was looking at were one of the possible decryptors. The advent of polymorphic viruses brought about an increased false alarm rate as many innocent files were marked as being infected.

The most significant event of the 1990s was probably the Michelangelo virus. The virus was first discovered in 1991, and upon examination it was determined that it would erase PC hard disks on March 6 —birthday of Renaissance painter Michelangelo. Until 500 PCs were accidentally shipped with the virus in January of 1992, Michelangelo remained a limited threat. Coincidentally, another manufacturer immediately announced its decision to include anti-virus software with every computer.

To compound the growing number of virus problems, virus hoaxes were becoming popular in the 1990s. Electronic mail (e-mail) hoaxes were being sent out to see how far the e-mail would go and how fast it would travel. Similar to chain letters, their purposes included harassment, pyramid schemes, or defamation of someone's (or an organization's) reputation. Spotting hoaxes and how to handle them will be covered later in this chapter.

By the end of the decade, most companies were relying on local area networks for communications (particularly e-mail and data file transfer) and use of the Internet had become widespread. Traditional viruses (Boot Sector and File Infector) began to diminish in frequency as Macro (document) viruses arrived on the scene. Another contributing factor may have been

1. Malware is short for malicious software. Software designed specifically to damage or disrupt a system, such as a virus or a Trojan horse or a worm.

the change in how software programs were distributed with the introduction of CD-ROM technology.

The first computer virus that would affect computer hardware arose in 1999. The CIH/Chernobyl virus would infect executable files and spread once that file was executed. This virus had the ability to spread quickly as many files are executed during normal use of a computer. Once active, CIH would attempt to erase the entire hard drive and then overwrite the system BIOS causing some machines to need a new BIOS chip to recover the computer.

As e-mail grew in popularity, it provided people with a new venue to create malicious code that could be used to spread new computer viruses. This new form of computer virus had the ability to disguise itself as an innocent looking file attachment and then would generate and send numerous infected e-mail messages to addresses found in the personal contact list. These early e-mail-based viruses required a user to open the attachment to be successful. Some of the newer forms (or worms) do not require user intervention, instead they exploit security weaknesses in the operating systems. Code Red, for instance, was designed to exploit vulnerabilities in Microsoft® Web servers (IIS) and had exceptional replication speed.

In addition, new programming languages designed for portability and functionality presented new opportunities to develop additional forms of malicious code. StrangeBrew (while harmless and actually more a proof of concept virus) was the first virus to infect Java files.The virus modifies Java "CLASS" files to contain a copy of itself within the middle of the file's legitimate code. When the "infected" class file is executed, the virus gets control and then passes control to the original code in the file. Being Java based, StrangeBrew is capable of executing in almost any platform that has Java runtime environment installed. While StrangeBrew did not do anything except spread, the virus is capable of executing on Windows and Linux platforms and in PDA devices that have Java runtime installed.

Nimda is a complex virus with a mass-mailing worm component that utilizes multiple methods to spread itself. The W32/Nimda worm, taking advantage of back doors left behind by the Code Red II worm, will propagate itself via several methods (multipartite), including e-mail, network shares, and an infected Web site. The worm spreads between clients and Web server by using the client to scan for vulnerable servers. It then uses a Unicode exploit to infect the Web server, and subsequently "uses" the Web server to propagate itself to other clients browsing the site. The e-mail variant transmits an attachment, which might be executed automatically on the recipient machine.

Klez is an e-mail virus that infects executables by creating a hidden copy of the original host file and then overwriting the original file with itself. There are numerous variants of this hybrid that exhibit characteristics of different forms of malicious code, including classic worms, polymorphic viruses, and file infector viruses. Klez variants all follow the same basic form. Polymorphic is a virus that changes its virus signature (i.e., its binary pattern) every time it replicates and infects a new file in order to keep from being detected by an antivirus program. In some variants, it searches and disables anti-virus and integrity checker software (retro virus), and even attacks other virus forms (such as Nimda, SirCam, and Code Red). In many variations, it acts as a worm (generates e-mail messages to propagate itself to other systems) but also "drops" a file infector virus called Elkern (Win32. Klez.b), which then survives independently of Klez.

The replication speed of Internet worms, using today's high-speed computers, as well as constant probing for vulnerabilities by hostile individuals and groups, mandate continuous awareness, improvement, monitoring, and testing of IT security by organizations and user communities. As personal and corporate Internet connectivity has continued to increase over the past several years, authors of popular browser technologies have added numerous companion tools, or plug-ins, using specialized scripting codes that can automate common functions. These tools comprise a generation of active content code that exposes additional opportunities to attackers. Table 7.1 pressents a brief overview of the chronological progression of computer viruses.

Macro Typically affecting Microsoft Office Products (e.g., Melissa, Concept), macro viruses are currently portable between platforms making them an extremely popular virus attack format.

Most macro viruses known to be in the wild infect MS Word files, but they can infect Excel files (although this is less common than Word macro infectors). Word macro viruses replicate into other documents by first copying themselves into the Word global template (normal.dot). Once the Normal. dot (this is the default template for all Word documents) is infected, any document opened will be infected.

Excel macro viruses show a less distinctive pattern, when compared to Word viruses. From the few that exist, they all replicate by creating their own workbook, containing an auto startup macro (auto_open), and place that workbook in the Excel startup directory — normally ...\Excel\XLStart.

Macro viruses infect other programming codes within a document by inserting or creating additional macro commands. When the infected document is transmitted and opened within the application, the infection spreads to the application components and then may re-propagate to other similar documents as they are accessed.

File Infector Typically attached to .COM or .EXE files (e.g., Monkey, AntiEXE), file infectors attack executable program files, typically those with a

Table 7.1 A Chronological Progression of Computer Viruses

1949	von Neumann suggests computer programs could reproduce
1950s	Bell Labs creates a game that attacks other computers
1975	Brunner imagines a computer worm
1984	Fred Cohen introduces the term “computer virus”
1986	First virus created (BRAIN)
1987	Christmas tree Worm cripples IBM
1988	Internet worm spreads through DARPA
1990	First polymorphic virus (1260) — Mark Washburn
1992	Michelangelo spreads and so does the panic
1994	First virus HOAX hits (Good Times)
1995	First MACRO virus hits (Concept)
1995	First virus specifically for Windows 95
1998	First virus to affect computer hardware (CIH/Chernobyl)
1999	First e-mail virus — Melissa forwards itself
1999	First virus infects computer when e-mail is read (Bubbleboy)
2000	Love Bug becomes the most successful e-mail virus
2000	First virus for the Palm operating system
2001	An estimated $2 billion in damages caused by Code Red Worm
2003	Attackers change their method of operation by combining viruses/worms with hacking techniques causing new concerns that are more difficult to fight against
2004	83% of all computer viruses are transmitted via electronic mail (e-mail) Approximately 20 billion e-mails are sent each day

COM or EXE extension. Sometimes files having an executable structure are targeted by viruses, regardless of their extension name. File infectors may corrupt non-executable files as well but they cannot spread this way.

File infectors can be memory resident (TSR). Once an infected file is executed, the virus will remain resident in the computer's memory until the computer is powered off. While in memory, the virus will continue to infect other programs. This activity could eventually interfere with normal operations. Once the PC is turned off, the virus will lie dormant in infected files until those programs are executed.

These viruses exhibit the classic “replicate and attach” behavior. Because of the wide acceptance and popularity of MS-DOS and Windows-based platforms, most of the well-know file infectors target those systems. They attack (typically) DOS program files with “COM” or “EXE” file extensions. Newer 32-bit virus strains are designed to work as well with “SYS” and many other file types.

Many of these viruses are written and compiled in C++ and other higher level languages, unlike Boot Record infectors which are often written in Assembly. While the coding of file infector viruses can be quite complex, the architecture of PC DOS .COM and .EXE applications is relatively straightforward.

The objective of this virus type is to attach itself to the original program file in such a manner as to control the execution of that file until it can replicate and infect other files, and possibly deliver a payload.

One derivative file infector, called a *Companion Virus*, is really a separate program file that does not require attachment to the original host program. Instead, a new (companion) program with a matching file name but a higher precedent extension is created that will be executed first in the same directory path as the real program. DOS will always execute COM files before EXE files, so issuing the Run command (or clicking the program icon) causes the virus to be directly executed instead of the intended legitimate program (e.g., Format.com and Format.exe). Once virus activity is completed, the illicit program simply executes the command to start the original program.

Boot Sector Infectors These typically affect boot sectors of hard disks and floppies (e.g., Form, Michelangelo). Figure 7.2 depicts how boot infectors work.

All disks (hard and floppy, bootable and non-bootable) contain a boot sector. The boot sector contains specific information relating to the formatting of the disk, the data stored there and also contains a small program called the boot program (which loads the DOS system files). You can be

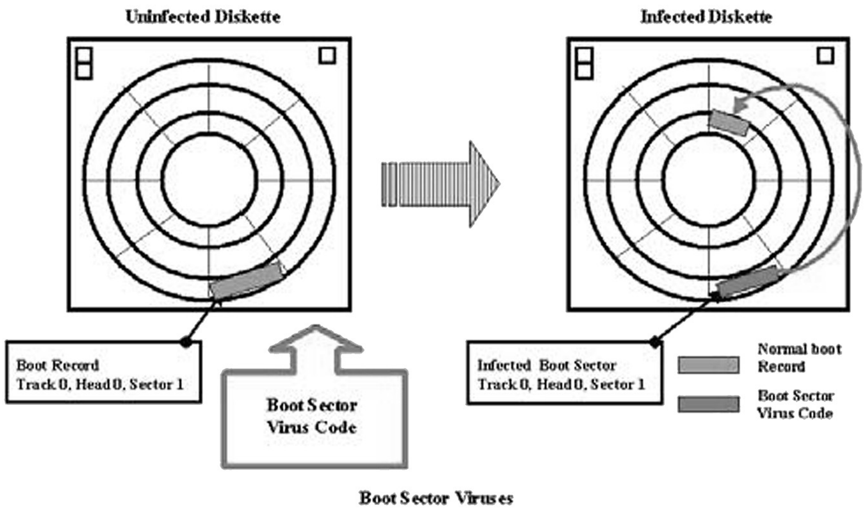


Figure 7.2 Multipartite viruses (boot and file infectors; e.g.,Tequila)

infected with a boot sector virus by using a diskette with a virus in a floppy drive and rebooting the machine. Once the boot sector program is read and executed, the virus becomes memory resident and will infect the boot sector on your hard drive.

System Infectors include a family of viruses targeting key hardware and system software components in a computing platform. The infected components are usually associated with system startup processes, allowing the virus to take control and execute before most software protective measures can be implemented. The most prevalent types of system infectors include Floppy Boot Record Infectors and Hard Drive Master Boot Record Infectors. These viruses are transmitted predominantly through the exchange of media (typically floppy disks) or are "dropped" as a special payload from a file infector virus.

Multipartite viruses infect both executable files and boot-partition sectors, sometimes the boot sector on floppies. Some multipartite viruses become infectious only after rebooting the computer from the infected MBR (Master Boot Record), like Tequila, others can be equally infectious if loaded from a file or through the boot process.

A Master Boot Record Infector moves or destroys the original Master Boot Record and replaces it with viral code. It can then gain control from the Bootstrap program and perform its hostile mission. Typical Master Boot Record infectors perform most of their tasks and then return control to the legitimate master boot record or the active partition boot record in order to mask their existence.

Both types of Boot Record Infectors typically load a viral proxy for the ROM-based system service provider process, thereby intercepting all normal application and operating system hardware requests. These requests include functions like opening and closing files and file directory services, thus creating an opportunity for the virus to execute other types of malicious code routines and also cover its tracks.

Virus Characteristics

Virus Structure All computer viruses typically, only an infection has to be present to be called a virus, have a Trigger, a Payload, and an Infection. The trigger is typically defined as the mechanism that defines whether a payload will be delivered or not. The payload is defined as what the virus does (beside the replication). The infection is defined as the way or ways that the virus spreads.

The payload of some computer viruses while sometimes harmless to an individual, can be used to clog e-mail systems and in some cases cause them to crash.

Table 7.2. Payloads Delivered by Computer Viruses

Payload	Action
Access denied	Files have been password protected, owner cannot read own files
Data corruption	Changes are made to the original data
Data deletion	Hard disks are overwritten
Data theft	E-mails information about the user or a machine to the author
Display messages	Messages can be displayed on a user's screen
Hardware disabled	The BIOS is overwritten, making the machine unusable
Pranks	Like displaying messages, it may play a game or a tune on the PC

Other viruses have a destructive payload that might change or corrupt files or post strange/obscene messages on the screen. The effect of these payloads can be immediate or in some cases, the virus may lay dormant until a specific date or time. See Table 7.2 for some of the payloads that can be delivered by computer viruses/worms.

Polymorphic Viruses While duplicating the main body of the virus, polymorphic viruses include a separate encryption engine which stores the virus body in encrypted format. Only the decryption routine itself is exposed for detection. The control portion of the virus is embedded in this decryption routine, which seizes control of the target system and decrypts the main body of the virus so that it can execute.

True polymorphic viruses use an additional mutation engine to vary the decryption process for each iteration, making even this portion of the code more difficult to identify.

Bugbear.B is a very complex polymorphic virus that spreads through both e-mail and network shares. The worm sends e-mails with various contents. It uses a known vulnerability to execute the attachment automatically when the e-mail is opened.

Stealth Viruses Stealth viruses use a number of techniques to conceal themselves from the user or detection software. By installing a low-level system service function, they can intercept any system request and alter the service output to conceal their presence. Stealth viruses are further classified as having size stealth, read stealth, or both.

Size stealthing is typically used by file infector viruses to mask the increase in file size by intercepting system requests for file information and subtracting its size from the reply before passing it back to the requesting process.

Read stealthing is typically used by boot viruses, again by intercepting any read/write requests for the normal boot sector (which has been relocated and replaced by the viral code). The request is essentially redirected to the new hidden location as necessary to satisfy the request, thus masking the presence of viral code.

Slow Viruses These viruses were conceived to counter the ability of anti-virus programs to detect changes in files that become infected. This class of virus is memory-resident (where anti-virus software cannot detect it) and is programmed to wait until certain tasks are requested, like copying or moving files. As the file is read into memory, the virus alters it before writing to the output file, making it much harder to detect.

Retro Viruses Some viruses (one term is Retro viruses) are designed specifically to attack or defeat countermeasures, such as anti-virus signature files or integrity databases. The virus, once active, searches for these data files and deletes or alters them, thereby crippling the AV software's ability to fully function. Other viruses, especially boot viruses (which gain control of the target system at startup) modify Windows Registry keys and other key files to disable AV, firewall, and IDS software if found. "Cpw" is a retro-virus that will delete SCAN.EXE if it's attempted to run (http://www.europe.f-secure.com/v-descs/cpw.shtml).

Zarma is a memory resident encrypted COM and EXE infector (stealth). It was found in France during May, 1995. Zarma is a stealth virus that intercepts interrupt functions to mask its presence on an infected system. The virus hooks int 3 to its own decryption routine. This routine decrypts a second decryptor on the stack. Zarma is also a retro-virus and is able to deactivate VSAFE, VDEFEND and VWATCH.

Klez, a more recent virus (or worm) example, also exhibits counterattack technology by searching for a variety of specific (AVP related) text strings in running processes. If it recognizes any of those Strings, it terminates that process. It can also remove or alter Registry keys of anti-virus software so that it is disabled when Windows starts. The virus strain W32/Elkern-C, which is dropped by Klez-H, contains routines to disable the on-access component of popular virus scanners.

Why Care?

Global competition has led many businesses to rely on their computing infrastructure. Technologies like the Internet, while a key business enabler, have also opened vulnerabilities in some organizations. The more connected computers become the faster computer viruses (or malicious code) can spread.

In August of 2003 virus developers demonstrated new techniques for the delivery of computer viruses. In one week, the names Welchi, Lovsan/Blaster, and Sobig became household names. Computer users worldwide began experiencing network problems, such as slow response or in some cases lost e-mails. Sobig at its height was able to send over 100 million e-mails from the computers it had successfully infected.

As previously stated, these new blended threats (hacker techniques combined with virus like behaviors) can spread faster, farther, and can cause more damage than before.

Annually, the FBI and the Computer Security Institute (http://www.gocsi.com) gather statistics for a "Computer Crime and Security Survey." These surveys confirm that the threat from computer crime and other information security breaches continues unabated, and that the financial toll is mounting. The survey states that financial loss due malicious code attacks or computer breaches continues to increase with the most serious financial losses occurring through theft of proprietary information.

Worms and Trojan Horses As we have already seen, computer worms have been around, or at least thought about, as long as computer viruses. A computer worm is a computer program that self-replicates, similar to a computer virus. The main difference between the two is that a virus attaches itself to, and becomes part of, another executable program, while a worm is self-contained; it does not need to be part of another program to propagate itself.

On November 2, 1988, Robert Tappan Morris (then 23) unleashes a worm that invades ARPANET computers. The small program disables roughly 6,000 computers on the network by flooding their memory banks with copies of itself. Morris later confessed that he created the worm out of boredom. His fine was $10,000 and three years' probation (he was convicted under the US Computer Crime and Abuse Act). Some of the notable computer worms were MELISSA and the CODE RED worm; the latter gained notoriety because it targeted the White House Web site.

Worms remained relatively simple until the early 2000s, when Klez sparked speculation that such worms could employ genetic algorithms.[2] Klez uses electronic mail to propagate, infecting Microsoft Windows systems (typically Outlook mail clients). Klez had both a text portion and attachment(s). The first attachment contained the worm. Ironically, the internals of this worm would vary slightly. The text portion would either contain a HTML[3] internal frame tag, which could be mistakenly executed by clients that were not properly patched or simply a few lines of code that claimed to be a fix for the Klez worm.

January of 2003 brought about a new form of the computer worm known as the SQL Slammer. This worm caused widespread problems on the Internet, as it created a denial of service attack, which caused general traffic on

2. A *genetic algorithm* is used to find approximate solutions to difficult-to-solve problems.
3. HTML — HyperText Markup Language — is a markup language designed for the creation of Web pages, that will be presented on the World Wide Web

the Internet to slow down. This worm spread to 75,000 host computers in less than ten minutes. SQL Slammer did not actually use SQL to deliver its payload but instead, it exploited two buffer overflow bugs in Microsoft's SQL Server. A buffer overflow is a type of computer bug. When the length limitation of a space reserved for data — a buffer — is not properly enforced, a buffer overflow may occur. Input data is written to the buffer and, if the input data is longer than the buffer size, the space beyond the end of the buffer is overwritten. If this space then contains executable code, it may execute in privileged mode.

Both Sobig and Blaster worms were released in August of 2003, resulting in the highest clean-up costs and largest downtime related to computer worms. These latest worms sparked many people to call for government intervention to prevent further damages.

Sobig is a computer worm in the sense that it replicates by itself, but is also a Trojan horse as it masquerades as something different than a virus. The Sobig "viruses" infect a host computer by way of attachments. However, when the attachment is run, the worm begins to replicate by using its own SMTP agent engine. This particular worm harvests e-mail addresses and stores the gathered information in files on the host computer. One of the variants (Sobig.F) was programmed to contact 20 IP addresses using UDP port 8998 and begin installing programs or simply updating itself. The Sobig worm was written using the Microsoft Visual C++ compiler, and was subsequently compressed using a data compression program.

In January, 2004, MyDoom became the fastest spreading worm and quickly exceeded all records set by previous worms.

Trojan Horses A Trojan horse is a computer program with an apparent or actual useful function that contains additional, malicious hidden functions[4] (sometimes a method of placing logic bombs, salami attacks, viruses, and such onto a system). The name comes from ancient Greek history. In this story, a giant wooden horse was offered as a gift to the citizens of Troy. To the dismay of the citizens, they quickly learned that this wondrous gift housed an army of their enemies. The name is usually shortened to Trojans.

Differing from a virus that is a stand-alone program, the Trojan does not attach itself to other programs. While a worm moves from computer to computer on its own, a Trojan does not as it requires human interventions, such as an e-mail.

One of the earliest known forms of a Trojan was in 1975, when the ANIMAL program, a game to identify an animal, also spread itself to other users on UNIVAC Exec 8 computers.

4. Tipton and Ruthberg, *Handbook of Information Security Management* (Boca Raton, FL: Auerbach, 1993), 757.

A Trojan can be easily hidden in the code of common business applications, which can be composed of hundreds of thousands of lines of code. The Trojan waits for the execution of the target program. Once executed, the Trojan can insert the new malicious instructions, execute them and then remove all traces in only a few milliseconds.

If a Trojan is suspected, a comparison of the operational program to the master or backup copy can be done to determine if there are any unauthorized changes. Additionally, examining system logs or audit logs may provide information on suspicious programs.

Typically, Trojans do not infect other programs and are quite simple to delete.

Double File Extensions Viruses, worms, and Trojan horses all make use of the double file extension. The Windows operating systems allows the creation of files names with a number of spaces in it. This trick is intended to fool users into believing that the file they are viewing cannot be executed. as in this example:

PLAIN.TXT.EXE

The .EXE at the end of the spaces, makes the program executable. Unfortunately in e-mail, users will only see the .TXT and potentially believe that the file is simply a Text file. This is why much has been done to educate users on not running e-mail attachments.

As a number of file extensions can be used to deliver or contain malicious code, it is recommended that the administrators block specific File Extensions at the Firewall. Table 7.3 is a partial list of suggested file extensions that should be blocked.

It is difficult for end users to understand all the file extensions that can be used and those that may be considered dangerous or Executable. Therefore, it is a good idea to develop a list of extensions that will be blocked at the Firewall by default. Every organization is unique and the list that is correct for one organization may not be correct for another. It is a good idea to educate users on some of the basic file extensions that you may not be able to block (i.e., .EXE, .PIF, .SCR, .COM).

A complete list of file extensions and their meanings is available at http://filext.com; also view http://whatis.techtarget.com/fileFormatA/0,289933,sid9,00.html.

Other Attacks There are many other forms of attacks that can affect the operation of a computer system. These attacks include (but are not limited to):

- Denial of service
- Flooding

Table 7.3 A Partial List of File Extensions That Should Be Blocked

File Extension	Descriptions
.API	Acrobat Plug-in Used to view Adobe Acrobat files
.BAT	Batch processing file Used to execute a series of commands in a sequential order
.BPL	Borland package libraries Used in programs developed with the Delphi software language
.CHM	Compiled HTML Help file Could include a link that would download and execute malicious code
.COM	Command File Contains scripts and executables for DOS or windows
.DLL	Dynamic Link Library Executable code that is shared by other programs on the system
.DRV	Device Driver Used to extend the hardware support of a Windows machine
.EXE	Windows binary executable program
.OCX	Object linking and embedding (OLE) control Used to orchestrate the interaction of several programs on a windows machine
.PIF	Program Information File Used to tell windows how to run non-windows applications
.SCR	Screen saver programs Includes binary executable code
.SYS	System configuration file Used to establish system settings
.VB*	Visual Basic® files (VBE and VBS) Used to script in visual basic which is built into many Windows-based machines
.WSH	Windows Script Host Settings File Used to configure the script interpreter program on Windows machines

- Brute force
- Dictionary attacks
- Spoofing
- Spamming
- Logic bombs

Denial of Service Attacks

TCP SYN Flood Attacks When a system ("the client") attempts to establish a TCP connection to a system providing a service ("the server"), the client and server exchange a set sequence of messages. This connection technique applies to all TCP connections — Telnet, Web, e-mail, and so forth.

The client system begins by sending a SYN message to the server. The server then acknowledges the SYN message by sending SYN-ACK message to the client. The client then finishes establishing the connection by responding with an ACK message. The connection between the client and the server is then open, and the service-specific data can be exchanged between the client and the server.

The potential for abuse arises at the point where the server system has sent an acknowledgment (SYN-ACK) back to client but has not yet received the ACK message. This is what is meant by a half-open connection. The server has built in its system memory a data structure describing all pending connections. This data structure is of finite size, and it can be made to overflow by intentionally creating too many partially open connections.

Creating half-open connections is easily accomplished with IP spoofing. The attacking system sends SYN messages to the victim server system; these appear to be legitimate but in fact reference a client system that is unable to respond to the SYN-ACK messages. This means that the final ACK message will never be sent to the victim server system.

The half-open connections data structure on the victim server system will eventually fill, then the system will be unable to accept any new incoming connections until the table is emptied out. Normally there is a timeout associated with a pending connection, so the half-open connections will eventually expire and the victim server system will recover. However, the attacking system can simply continue sending IP-spoofed packets requesting new connections faster than the victim system can expire the pending connections.

In most cases, the victim of such an attack will have difficulty in accepting any new incoming network connection. In these cases, the attack does not affect existing incoming connections or the ability to originate outgoing network connections. However, in some cases, the system may exhaust memory, crash, or be rendered otherwise inoperative.

The location of the attacking system is obscured because the source addresses in the SYN packets are often implausible. When the packet arrives at the victim server system, there is no way to determine its true source. Since the network forwards packets based on destination address, the only way to validate the source of a packet is to use input source filtering.

Systems providing TCP-based services to the Internet community may be unable to provide those services while under attack and for some time after the attack ceases. The service itself is not harmed by the attack; usually only the ability to provide the service is impaired. In some cases, the system may exhaust memory, crash, or be rendered otherwise inoperative.

SYN Denial of Service attack begins with (1) spoofed TCP SYN request to standard port, with spoofed source address. Server responds to spoofed

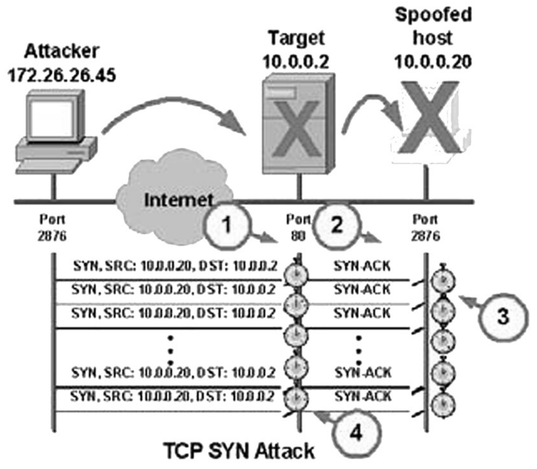


Figure 7.3 TCP SYN attack

address with SYN-ACK (2) and opens a connection state (see Figure 7.3). Since spoofed host did not originate SYN, it ignores the message (3), and the half-open sessions accumulate on the server (4), filling up buffer space and preventing legitimate new requests from being serviced.

Smurf Attacks In the "smurf" attack (see Figure 7.4), attackers are using ICMP echo request packets directed to IP broadcast addresses from remote locations to generate denial-of-service attacks. The two main components to the smurf denial-of-service attack are the use of forged ICMP echo request packets and the direction of packets to IP broadcast addresses.

There are three parties in these attacks: the attacker, the intermediary, and the victim (note that the intermediary can also be a victim). The intermediaries receive an ICMP echo request packet directed to the IP broadcast address of their network. If the intermediary does not filter ICMP traffic directed to IP broadcast addresses, many of the machines on the network will receive this ICMP echo request packet and send an ICMP echo reply packet back. When (potentially) all the machines on a network respond to this ICMP echo request, the result can be severe network congestion or outages.

The ICMP message is sent as a directed broadcast to all hosts in a network with a spoofed source address which appears to be from the target system. All hosts respond with a reply to the target, overloading it with traffic.

Distributed Denial of Service (DDoS) Attacks (DDoS) (see Figure 7.5) also uses intermediaries (unsuspecting) hosts to conduct an attack. These intermediaries are compromised systems on which Trojan handler pro-

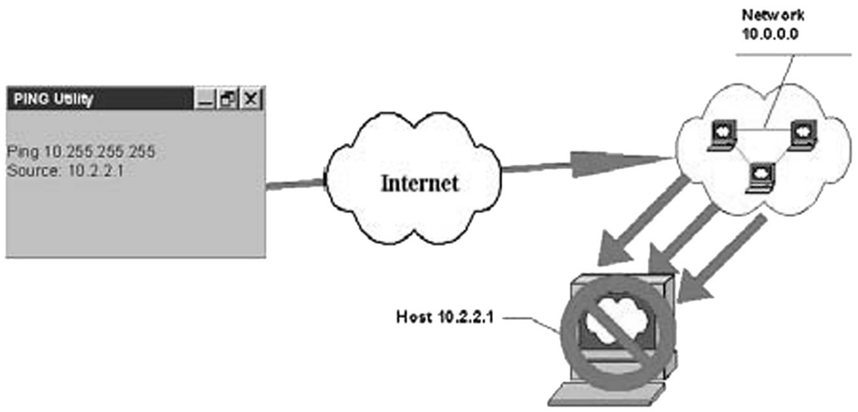


Table 7. 4 Smurf attack

grams are installed. These Trojan programs then act as agents to execute a coordinated attack on a target system or network. The attacker(s) control one or more "master" handler servers, each of which can control many agents or "daemons." The agents are all instructed to coordinate a packet-based attack against one or more victim systems.

There are three parties in these attacks: the attacker, the intermediaries

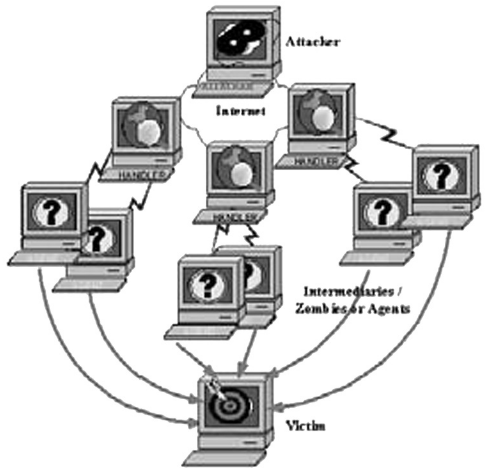


Figure 7.5 DDos attack

Table 7.4 Tools Used for Flooding or Attacking

Tools	Flooding or Attack Methods
Trin00	UDP
Tribe Flood Network	UDP, ICMP, SYN, Smurf
Stacheldracht	UDP, ICMP, SYN, Smurf
TFN 2K	UDP, ICMP, SYN, Smurf
Shaft	UDP, ICMP, SYN, combo
Mstream	Stream (ACK)
Trinity, Trinity V3	UDP, SYN, Random Flag, ACK, Fragment

(handlers and agents), and the victim(s). Even though the intermediary is not the intended "victim," the intermediary can also be victimized by suffering the same types of problem that the "victim" does in these attacks.

Attackers have developed automated DDoS tools (see Table 7. 4) that enable them to send these attacks to multiple intermediaries at the same time, causing all of the intermediaries to direct their responses to the same victim. Attackers have also developed tools to look for network routers that do not filter broadcast traffic and networks where multiple hosts respond. These networks can then subsequently be used as intermediaries in attacks.

Additional information on these exploits can be researched at CERT (http://www.cert.org) or by searching on the exploit.

IP Fragmentation and RPC Null Fragment Attacks IP fragmentation exploits involve generating TCP or ICMP packets using tools or command extensions which permit arbitrary modification of the packet flag fields. This takes advantage of the protocol's ability to break large messages into fragments for transmission across firewalls to meet transmission size constraints (Maximum Transmission Units or MTU).

Under normal circumstances, destination hosts must wait until all fragments have arrived before processing the message. If the message appears to contain additional fragments, the destination service must wait. The exploit arbitrarily generates multiple messages which indicate more fragments are to follow.

Mail Bombing and Spamming E-mail bombing is characterized by abusers repeatedly sending an e-mail message to a particular address at a specific victim site. In many instances, the messages will be large and constructed from meaningless data in an effort to consume additional system and network resources. Multiple accounts at the target site may be abused, increasing the denial of service impact.

E-mail spamming is a variant of bombing; it refers to sending e-mail to hundreds or thousands of users (or to lists that expand to that many users). E-mail spamming can be made worse if recipients reply to the e-mail, causing all the original addressees to receive the reply. It may also occur innocently, as a result of sending a message to mailing lists and not realizing that the list expands to thousands of users or as a result of an auto-responder message that is setup incorrectly.

E-mail bombing/spamming may be combined with e-mail spoofing or through the use of bulk "re-mailers" (which alter the identity of the account(s) sending the e-mail), making it more difficult to determine who actually sent the e-mail. Re-mailer services are becoming popular for individuals wanting to privatize their personal e-mail, but they can be used as an agent to distribute spam as well. When large amounts of e-mail are directed to or through a single site, the site may suffer a denial of service through loss of network connectivity, system crashes, or failure of a service because of:

- Overloading network connections
- Using all available system resources
- Filling the disk as a result of multiple postings and resulting in syslog entries

(Note: There is an increasing amount of pending legislation designed to place controls and restrictions on anonymous bulk mailing due to the serious business impact of spam and other unsolicited high volume traffic.)

Pestware and Pranks A Pest can refer to undesired processes or code that might be found on your PC or your network after installing freeware applications that are downloaded from an Internet site.

Pests are generally nuisances but might include more dangerous Trojans, spyware, remote administration tools, hacker tool kits, and more. Pestware may threaten confidentiality and privacy, and, due to the increasing incidence of this, there is an administrative impact on productivity.

Pestware in the form of tracking cookies, is sometimes employed by commercial Web sites as a means of tracking a user's Internet browsing habits or as pop-up or pop-under Web pages to push unsolicited offers to a user who visits a competitive Web site.

Pranks, on the other hand, are often the work of adventurous "script kiddies" or other internal or external users, and can be malicious as well as a nuisance.

ANSI Bombs These were quite possibly the original e-mail "virus," although they were not viruses at all. ANSI Bombs are simply modified TEXT or ANSI files that, when TYPED (with the DOS command) would re-map the target host keyboard. ANSI Bombs use the ANSI Escape sequence (with the reserved command "P") that was designed to perform a relatively simple

Macro task when you typed certain keys and relied on two very critical factors that were generally present under DOS:

- The ANSI driver (ANSI.SYS) was loaded in CONFIG.SYS
- The software displayed information via the system BIOS

The attacker would disguise the ANSI bomb as a downloadable DOS program (game, etc.) application on BBS sites or transmit the file via diskette. Once executed, the program could re-map keyboard keys (e.g., the space bar) to perform some malicious task such as formatting drives or erasing critical files.

These threats generally diminished when Microsoft Windows became the dominating operating system on PCs. Windows applications do not utilize the system BIOS to process and display information, thus the character sequence is never routed through the ANSI driver.

Again, they were not viruses — they were simple commands, designed to be loaded into the ANSI driver and modify the behavior of your display and/or keyboard.

Adware, Web Site Tracking Cookies (Trackware) Many adware applications install Trojan advertising components on your system, that run — downloading ads and wasting system resources — even if you are not using the software that installed them. Often, these components operate in stealth mode and remain installed and continue to perform after the associated app has been uninstalled. These advertising Trojans make clandestine connections to adservers behind your back, consume precious network bandwidth and may compromise the security of your data. These include the TimeSink/Conducent TSADBOT and the Aureate/Radiate advertising Trojans.

A "cookie" is a token that stores information about a browser session. The server side of a user connection to a Web server can place certain information in the cookie, then give that cookie to the user's browser. On some other page, that server can ask the browser for the cookie, and retrieve the information previously stored. This becomes useful in any intelligent interaction between browser and Web site, because the connection between a browser and server is not persistent.

Cross-site tracking cookies (sometimes referred to incorrectly as spyware) are simply those cookies that are not used only by a single site for its private interactions with its users, but are shared across sites. Some cookies are persistent and are stored on your hard drive indefinitely without your permission. They can reveal and share private information collected among multiple sites, as they are visited.

Cookie Poisoning While cookies are intended to store personal information and only forward that information back to the server, this information can be modified by an attacker. As an example, if one is able to modify the total value of an online order, they could pay less than originally intended when the data is returned to the server.

Homepage Hijacking The goal of these attacks is to change your browser's homepage to point to their site. There are two forms of hijacking:

- Exploiting an IE vulnerability to automatically reset the homepage.
- Covert installation of a Browser Helper Object (BHO) Trojan program, which contains the hijacking code.

Once a BHO Trojan gets executed, it changes (or forces) the browser's homepage back to the hijacker's desired site. Typically, hijacker programs put a reference to themselves into the Windows StartUp folder or Registry Run key, so that the hijacker runs every time the computer is started. If the user tries to change any of these settings, the hijacker repeatedly changes them back until the hijacking software can be found and removed.

Web Page Defacements The terms "Web defacement" or "Web graffiti" refer to an incident when someone gains unauthorized access to a Web server and alters the index page of that particular violated site. Usually the attacker exploits known vulnerabilities in the target server and gains administrative access. Once in control, html pages are replaced with altered versions.

Typically, the defacement represents graffiti; however, while most security practitioners consider this attack a nuisance, the potential for embedding more malicious active content code (such as viruses or Trojans) into the Web site exists. Code Red, for instance, included payload that installed a backdoor Trojan. This Trojan allows remote access to an infected IIS server which can be used to deface the front page of the Web server.

Ensuring that current software versions and security patches are installed and active monitoring of Web sites will minimize the risks.

Brute Force Attacks Brute force attacks utilize exhaustive trial and error methods to obtain the information about passwords or cryptographic keys.

If a brute force attack was launched against a password containing five (5) numbers, there would be 10*10*10*10*10 or 100,000 possible combinations that the system would test. Adding the complexity of numbers (0–9), characters (upper and lower case A–Z) and special characters (! @ # $) increases the number of possible combinations making a brute force attack less attractive and more difficult as it would require more processing power. Brute force can eliminate a large number of possible combinations.

Trying to crack a 56 bit DES[5] key using a brute force attack with today's processing power could take hundreds of years with one PC. However, with

5. DES — Data Encryption Standard

the growth of the Internet and the ability to link computers on the Internet, the time to perform this exhaustive task has been drastically reduced. As this is the case, you will find that DES is obsolete and AES or triple DES are now the standard.

Dictionary Attacks Using a dictionary attack, an attacker will take a specific list of words (typically a dictionary listing) and compare those words against passwords stored in the access control listing. Once the attacker has obtained the password file, tools can be used to try to obtain the password. Most passwords can be guessed using a brute force attack but utilizing one of these tools drastically cuts the attacker's time as words in the list are compared.

In order to gain access to the password files, one must either be internal to an organization or be able to break through the corporation's defenses. Once this is accomplished, tools are available that will allow the dump (or make a copy of) the password file to an alternate location, allowing the attacker ample time to extract the passwords.

The reader is reminded that the English dictionary is not the only one that can be used. All foreign language dictionaries as well as dictionaries such as Lord of the Rings, Star Trek, Star Wars, and so forth have been written and can be used with these tools.

Hashed Password or Password-Verifier Passwords stored in a database should be stored in a one-way hashed form, to prevent casual retrieval of the information. Since passwords are often vulnerable to a dictionary attack, preventing unauthorized access to this data thus remains a high priority. In general, the requirement for secure host storage is characteristic of all mutual authentication cryptographic systems. Alternative public-key methods are especially sensitive to the theft of a stored private key. Since the hashing algorithm to "encrypt" the password file can be discovered easily, it is not difficult to use that to hash the dictionary words and compare the hashed versions to the downloaded password file to find a match that leads to the password in the dictionary. That is why reusable passwords are considered to be virtually useless today.

Online versus Offline Attack An online attack requires the active participation of a legitimate user or host. The important things are to minimize the information revealed in each attack, and to ensure that the legitimate party is aware that an attack or failure has occurred. A user naturally becomes suspicious and reports trouble when a large number of failures occur, and the system should encourage this. A host typically counts bad or suspicious attempts, and takes remedial action when a limit is exceeded.

Offline password attacks have historically been harder to prevent. We must assume that an attacker has a large amount of CPU power, has technical expertise, and can monitor or probe the network to gather password protocol messages. This is true for anyone who has a Pentium, Web access, and can click to download and run a cracker's tool. Strong password protocols ensure that gathered messages cannot be used offline to computationally determine the password.

Salt "Salt" is a value incorporated into the calculation of the hashed-password. The salt is typically chosen randomly at the time of password selection, and stored along with the hashed-password. Another choice is to calculate salt from the user's name. Using salt, two users with the same password will have different hashed values, which makes it harder to create a pre-built dictionary of likely hashes. This technique decreases the efficiency of broad-based dictionary attack against a readable password database for many users.

The UNIX /etc/passwd mechanism used a random two-character salt for each user. Modern protocols use a larger salt, which makes broad-based attack impossible — each hashed-password must be attacked individually.

Salt plays the same role in a hashed-password database for network authentication, to reduce the threat if the database is revealed. The salt is typically sent from the host to the client as a prelude to password verification. But regardless of whether salt is used, protection of the password database remains a higher priority.

Logic Bombs A logic bomb is code added to the software of an application or operating system, which will lie dormant until an event occurs. The event can be a date or a specific condition on the system. This event will trigger the code to take some actions. Typically, logic bombs are malicious in their nature. The ultimate motive of a logic bomb is usually to delete, alter or corrupt data that the program has access to.

Ping of Death Ping is a "utility" program used to determine if a specific IP address is accessible and is primarily used to troubleshoot network or Internet connections. Ping sends packets to the specified address and waits for a reply.

Typically, IP packets are 65,535 bytes in length (as per RFC-791), including the header length (generally 20 bytes if no IP options are specified). Packets that are bigger than the maximum size the underlying layer can handle (the MTU) are fragmented into smaller packets, which are then reassembled by the receiver. For ethernet devices, the MTU is typically 1500.

An ICMP ECHO request "lives" inside the IP packet, consisting of eight octets of ICMP header information (RFC-792) followed by the number of data bytes in the "ping" request. Hence, the maximum allowable size of the data area is 65535 - 20 - 8 = 65507 octets.

Most computers do not process packets until all the fragments have been reassembled. A Ping of Death would occur when an illegal echo packet with grater than 65507 bytes is sent due to the way the fragmentation is performed. Fragmentation relies on the offset value in each fragment to determine where the packet goes on reassembly. Thus, when the packets are reassembled, there is the chance for system buffers to overflow causing the system to crash, reboot, hang protocols, or the like.

Spoofing Attacks

IP Spoofing Spoofing is a technique used to attain unauthorized access to computers or in a sense, masking the true identity or source.

IP Spoofing allows an attacker to attain unauthorized access to a network or computer making it look like the message (malicious or not) is coming from a trusted source. This is accomplished by forging the IP address of the trusted source.

Through the years there have been many forms of IP spoofing attacks, some of which are still pertinent to security today.

Non-Blind Spoofing To perform this type of spoof, the attacker is usually on the same subnet as the victim.

In this attack, a sniffer is used to gather information on the sequence and acknowledgement numbers eliminating the potential difficulty of calculating them accurately. Once gathered, the largest threat is typically session hijacking. An established connection is interrupted (or corrupted) and a new session connected using the sequence and acknowledgement numbers that were collected using the sniffer. This can effectively bypass any authentication measures built into the process.

ARP Spoofing An ARP spoof forges the packet source hardware address (MAC address) to the address that the attacker wishes to use.

Man in the Middle Attack In a Man in the Middle attack (sometimes referred to as TCP hijacking), the attacker intercepts communications between trusted parties. Once the communication is intercepted, the attacker has control of the communications and can modify or delete the information that is being sent between the victims. This attack is useful in attaining confidential information.

Denial of Service Attack (DoS) In this attack, the attacker attempts to flood the victim with as many packets as possible in a relatively short space of time. A DoS attack is difficult to defend against as the cracker will use multiple spoofed IP addresses to perform the attack making it almost impossible to trace or stop.

Active Content Vulnerabilities The term "active content" includes ActiveX™, Java, JavaScript/JScript, VBscript, macros, browser plug-ins, scrap files, Windows scripting host files, and Postscript™. This code runs in the context of the user signed on to a PC and can do everything that the user can do. Another term that seems to be gaining in usage, "vandal," is defined as any malicious auto-executable application, which includes the above-mentioned active content.

These active content threats, including ActiveX, Java, and JavaScript code in HTTP data streams, are often referred to as "mobile code" since these programs are written to run on a wide variety of computer platforms. Key points to consider regarding active code:

- Java applets are considered to be untrusted code.
- They are run within a *virtual machine* that uses a *sandbox* approach to theoretically restrict what they can do, preventing inappropriate actions on the user's computers.
- It is possible for Java code to trigger (or spawn) execution of OS functions that do not have this restriction (although this is more of an issue of loopholes prevalent in Microsoft Internet Explorer implementations).
- ActiveX is widely considered to be the greater threat because it is essentially an outgrowth of *OLE*, which permits direct access to native Windows calls and links them to system functions.
- ActiveX has no built-in language restrictions controlling code behavior.
- ActiveX controls can be built utilizing many different programming languages.

Many Internet Web sites now rely on Java applets and ActiveX controls to create their look and feel. For these schemes to operate properly, these bits of mobile code are downloaded to the user's PC, where they gain access to the local host and can then spawn malicious activity (viral infections, worm behavior, or Trojan functions).

Types of Mobile Code Attacks While mobile code attacks can take many forms, there are four primary types of assaults:

- **Launch Point —** The malicious mobile code can use the targeted computer as a launch point to infect and target other computers.
- **Distributed Denial of Service (DDoS) —** In a DDoS attack, zombie agents automatically generate hundreds of authentication requests to the server simultaneously, which can quickly overwhelm targeted Web sites. Since all the requests have false return addresses, the server is unable to find the user when it tries to send the authentication approval. The server waits, sometimes more than a minute, before closing the connection. When it does close the connection, the zombie agent sends a new batch of forged requests, and the process begins again, tying up the service indefinitely.
  - This attack will impact Availability of computers, networks, and servers.

- **Data Modification —** The mobile code is instructed to access a file on a local or network drive, and modifies, deletes, or overwrites it with new data. Sometimes this type of mobile code is used to modify system settings or browser security settings. Data modification can impact the Integrity of data.
- **Data Export —** Malicious mobile code can steal information from your computer and forward it over the Internet or e-mail to an attacker. For instance, many Trojan horses will forward your user name and password to an anonymous e-mail address on the Web. A third party can then use the password to access protected resources. Exporting data could impact the Confidentiality of the data.

Attacks and Exploits Using Malformed Data

Server applications, in general, operate within some defined set of protocols or program specifications. These define the normal or expected parameters of operation of the related service. Exploitation of the sometimes inflexible boundaries of these parameters has been a popular target for years. Common exploits include:

- Overwhelming the predefined capacity of these services to handle new requests (denial of service types of exploits).
- Sending information in specially crafted packets that contain erroneous information (such as size information or commands that contain special characters), which causes program buffer overflows. This error condition may freeze or crash the server, or may open error handling services which can be exploited to gain control and execute additional (malicious) code.
- Sending specific strings of characters to well know services (IIS for instance), which contain nested or imbedded command syntax that is arbitrarily executed by the server.

As an example, there are two major Unicode vulnerabilities: the IIS/PWS Extended Unicode Directory Traversal Vulnerability and the IIS/PWS Escaped Character Decoding Command Execution Vulnerability. Many current worms have used these two Unicode vulnerabilities in IIS to good effect

Worms having being using buffer overflow conditions to gain service control of a target platform since 1988.

These vulnerabilities can allow attackers to run arbitrary code on the target servers, possibly uploading further compromises (as Nimda does using TFTP).

IT security practitioners can protect themselves from these specific vulnerabilities by installing the recent service packs and security updates from Microsoft. Other recommendations include:

- Do not use default directory/share names or locations. Customize them for your site.
- Carefully set permissions on shares.
- Turn off all unneeded functions or disable unused extensions in IIS.

How Active Content Operates

Active content comes in many types these days from application macros, to applets to background scripts. Postscript, Java, JavaScript, and Visual Basic Script, ActiveX, Macros, and browser plug-ins are all capable of being exploited through well-defined and documented active content features. All of these have potential weaknesses that can be exploited.

ActiveX is part of the Microsoft® Component Object Model, COM for short, that provides reusable code segments for application programs. These compact code modules help control many aspects of Wintel applications and hardware and allow for the automation of many background tasks.

The security model for ActiveX relies heavily on user interaction in order to ensure that unsigned controls are not downloaded or executed on a host system. Other than this trust model, ActiveX controls have little restrictions on what they can do once they are given permission to proceed. Once active, ActiveX assumes all of the rights of any local program on the host, and as such, can arbitrarily execute any damaging routines that other malicious code can.

ActiveX controls can be registered and digitally signed by commercial certificate authorities like Verisign (and also by local/private CAs). These signed applets, like any legitimate application program, become known as "Authenticode" and carry a digital signature to verify their integrity and source.

Of the 1,000 or so registered controls, only 50 to 100 have the marked designation as safe for scripting. Privately signed controls (possibly custom controls) authenticated by CAs which do not maintain Certificate Revocation Lists (CRLs) present an additional concern, since the security model is based on all or nothing permissions. If the control appears safe, and is executed (accepted), then it will assume full permissions and capability.

One problem with unsigned ActiveX controls is they are vulnerable to exploits launched by text (scripts) imbedded within HTML documents. Accessing an infected HTML file may cause a malicious script in the HTML file to run automatically if your browser security settings allow it.

In viruses carried by this method, infected script might search for all *.HTM and *.HTML files in the current directory and all directories above it and infect them as well. The infection basically prepends the virus to the

HTML file which can then be propagated to other hosts accessing that Web page or e-mail.

JavaScript and Visual Basic Script

Both of these languages belong to a class of scripting tools whose code can be embedded into Web pages for creating highly interactive documents. The theory of their operation dictates that they cannot directly access a client file system and communications are restricted to the host from which the content originates. There are numerous design and implementation bugs in both of these products.

The biggest flaw is that they work within the context of the browser and, in theory, are limited or bounded by whatever is legal within the browser. Unfortunately, modern browsers are bound tightly to other applications, such as e-mail and have various plug-ins and ActiveX controls that can be accessed. Additionally, these scripts will assume whatever your privileges are on the systems you are logged into at the time of their execution. If you hit a malicious script while logged in as an administrator, then the script will run with administrative privileges which could be devastating to security.

Java Active Code

Java is the universal code that is similar in nature to ActiveX in stated purpose only. It is the reusable code set that can be written once and run on many different types of hardware platforms if there is an interpreter called the Java Virtual Machine (JVM) on the target host.

The resulting byte code created from Java is conveyed by an HTML Web page as an applet. Java, unlike ActiveX, tried to address the security shortcomings of other modular programming languages by addressing security from the onset. It provides a security architecture known as a *sandbox*.

According to the National Institute of Standards and Technology (NIST), the sandbox is a bounded area that "restricts the access of the applet code to computational resources based on its permissions." That is to say, unless the applet is trusted, it can only use resources within the bounded area (it does not acquire permissions to execute external code). The problem with permissions is that they are defined again as trusted resources (e.g., where it comes from and who wrote the applet).

While the sandbox is supposed to prevent the applet from accessing files or even changing them and prevent accessing the network, there have been many successful exploits that circumvent the sandbox security construct.

These exploits are primarily related to implementation flaws in developing applets.

Structure and Focus of Malicious Code Attacks

Uncoordinated (or unstructured) attacks against network resources are generally perpetrated by moderately skilled persons such as script kiddies and cyberpunks. Often, the initial intent is personal gratification (or the thrill of the challenge) of achieving illegal access to a network or target system without any real purpose in mind. Any level of success may in fact lead to further exploration and more malicious activity such as defacements or crashing systems. All such activity is of concern to IT security practitioners as it represents a compromise of defensive measures.

Occasionally, such an uncoordinated attack exposes an unintended vulnerability and the attacker may then change his or her activities to a more methodical approach. When an attacker finds additional or unplanned targets to pursue during or after an attack this is is sometimes called “rat dancing,” “shaking doors,” or “rattling locks.”

Coordinated, pre-planned attacks are usually conducted by adversaries who are highly motivated and technically skilled crackers, using complex tools and focused efforts. These attackers may act alone or in groups, and they understand, develop, and use sophisticated hacking techniques to locate, identify, penetrate, probe, and then carry out malicious activities. Such an attacker’s motives may include money, anger, destruction, or political objectives.

Regardless of the motivations, these attackers can and do inflict serious business damage to networks. Structured attacks are usually conducted in phases, once an overall goal is established. It may be aimed at a specific organization or a specific technology (such as an OS version).

Directed Attacks against specific targets (specific organizations) or target classes (i.e., networks that are using certain hardware, operating system versions, or services) are often conducted as interactive sessions using predefined exploits. An example might be an IIS Unicode attack against specific Web servers in an organization.

These exploits may be uncoordinated or unstructured, as when a script kiddie uses well-known hacker tools to discover vulnerable sites, and then conducts random exploits to rat dance around the compromised network through trial and error.

They may also be conducted as coordinated or pre-planned attacks, by individual crackers or by coordinated cyber-terrorist groups, and advance methodically through phases to achieve their desired goals.

Indirect Attacks occur as a natural result of preprogrammed hostile code exploits, such as Internet worms or viruses. These attacks are unleashed indiscriminately and are designed to propagate rapidly and widely. While the worm or virus itself may be written to exploit a specific system or application vulnerability, the replication and transmission components of the code are designed to propagate indiscriminately.

It is very likely that one of the goals of a directed attack against a specific target might be to establish a starting point for a more indirect attack against a more widely disbursed population. The compromise of a single Web server to install an e-mail worm as a denial of service exploit might be the intended goal.

Phases of an Attack

Reconnaissance and Probing

Once the overall goal or objective of an attack is clear, an attacker must then probe a target network and identify points of possible entry (the vulnerabilities). This phase generally involves the use of common tools that are readily available on the Internet, are part of the underlying protocol suite, or a custom developed to exploit specific or potential targets. These can include:

- Use of DNS and ICMP tools within the TCP/IP protocol suite
- Use of standard and customized SNMP tools
- Using port scanners and mappers to locate potential target services
- Dissemination of spyware Trojan programs to collect reconnaissance data

These tools might be used independently or as a coordinated suite of activities designed to provide a complete understanding of a targeted network (what protocols and operating system is used, what server platforms exist, what services/ports are open, what actual or probable network addressing and naming is being used, etc.).

The Internet and other public sources can provide additional information necessary to profile targets, including locations of facilities, key personnel, and likely business partners. This last piece of information may seem trivial, but an indirect assault committed through a trading partner having serious security breaches is very possible.

DNS Commands and Tools

The Domain Name System (DNS) is a hierarchy of servers that provide an Internet-wide symbolic name to IP address mapping for hosts connected to the Internet. Publicly available information on registered addresses is obtainable through a number of searchable Web sites. In addition, discovery tools that are built into TCP/IP, such as whois and finger, can be used to gather preliminary information in profiling a target site.

Reverse DNS lookup or nslookup are additional utility commands that will also interrogate DNS information and provide cross-referencing. These services are often provided free on the Internet and can be located simply by searching on the command name itself.

The example above is from http://ww1.arin.net/whois/ which can be used to locate the IP address of a potential target network. Arin's whois will *not* locate any domain-related information, or any information relating to military networks. Many operating systems provide a whois utility. To conduct a query from the command line, the format generally is:

> Whois-h hostname identifier (e.g., whois-h whois.arin.net<query string>).

ICMP and Related TCP/IP Tools

The Internet Control Management Protocol (ICMP) PING command and several closely related tools are readily available on most computer operating systems and would be a key profiling tool to verify that target systems are reachable. The PING command can be used with a number of extension flags to test direct reach ability between hosts or as part of the actual attack plan (see Ping of Death attacks).

Once a target network has been located, many attackers then perform a Ping Sweep of all, or a range of, IP addresses within the major network or subnet to identify other potential hosts that may be accessible. This information alone sometimes exposes the likely network size and topology, and, because many networks use a structured numbering scheme, may also point to likely server and network device locations.

If gaining access is one of the objectives, a simple telnet login attempt might be performed initially to test the softness of perimeter controls. Rpcinfo might also be used to determine if this service is active for remote command execution.

Remote SNMP Agent Discovery utilities let you discover responsive SNMP agents running on network devices. This example reflects, for instance, a server ("APOLLO"), located at 212.30.73.70, which is responding to SNMP queries, and might now be probed or scanned for other open service ports.

Using SNMP Tools

The Simple Network Management Protocol (SNMP) is an application layer protocol that facilitates the exchange of management information between network devices. It is part of the Transmission Control Protocol/Internet Protocol (TCP/IP) protocol suite. SNMP enables network administrators to manage network performance, find and solve network problems, and plan for network growth.

Many of the more popular Network Management software suites, like HP OpenView, SunNet Manager, and AIX NetView, are SNMP compliant and offer full support for managed devices, agents, and network-management systems. In addition, there are many utility programs which can be used to gather network device information, including platform, operating system version, and capabilities. Poorly configured network management facilities would allow moderately skilled attackers to gather significant attack profile information.

Port Scanning and Port Mapping

Once a target network has been initially identified, the attacker might proceed to explore what systems and services are accessible for further investigation. There are several popular port scanning applications an attacker might use. One of the most popular, shown above, is nMap (available for UNIX and Windows). MingSweeper is another network reconnaissance tool for Windows NT/2000, designed to facilitate large address space, high-speed node discovery and identification.

These tools permit an attacker to discover and identify hosts by performing ping sweeps, probe for open TCP and UDP service ports, and identify operating systems and vulnerable applications that might be running.

The utility program (CHKSATAN) represents a countermeasure often used to detect the presence of SATAN (Security Administrator Tool for Analyzing Networks) performing a ping sweep and remote procedure call probe. If detection is made, the routine creates a temporary access control filter blocking further access for the specified period (ten minutes).

Security Probes

SATAN is a tool designed to help systems security administrators evaluate a number of vulnerabilities. It recognizes several common networking-related security problems and reports the problems without actually exploiting them. SATAN collects information that is available to anyone with access to the network. With a properly configured firewall in place that should be near-zero information for outsiders.

For each type or problem found, SATAN offers a tutorial that explains the problem and what its impact could be. The tutorial also explains what can be done about the problem: correct an error in a configuration file, install a bugfix from the vendor, and use other means to restrict access, or simply disable service.

On networks with more than a few dozen systems, SATAN will inevitably find problems, such as:

- NFS file systems exported to arbitrary hosts or to unprivileged programs via portmapper
- NIS password file access from arbitrary hosts
- Arbitrary files accessible via TFTP
- Remote shell access from arbitrary hosts
- Writable anonymous FTP home directory

These are well-known problems. They have been subject of CERT, CIAC, or other advisories, or are described extensively in practical security handbooks. The problems have been exploited by the intruder community for a long time.

SATAN is a two-edged sword, however; like many tools, it can be used for good and for evil purposes. It is a good idea to include scanning for evidence of SATAN reconnaissance of your network.

Use of Spyware and Backdoor Trojans

Spyware is a term for Trojan software that employs a user's Internet connection in the background (the so-called "back-channel") without his or her knowledge or explicit permission. Spyware exists as an independent, executable program on your system, and has the capability to monitor keystrokes, arbitrarily scan files on your hard drive, snoop other applications such as word-processors and chat programs, read your cookies, change your default homepage, interface with your default Web browser to determine what Web sites you are visiting, and monitor other user behavior, and transmitting this information back to the author.

War Dialing A war dialer is a computer program used to identify the phone numbers that can successfully make a connection with a computer modem. The program automatically dials a defined range of phone numbers and logs and enters in a database those numbers that successfully connect to the modem. Some programs can also identify the particular operating system running in the computer and may also conduct automated penetration testing. In such cases, the war dialer runs through a predetermined list of common user names and passwords in an attempt to gain access to the system.

A war dialer, usually obtained as freeware, is typically used by a hacker to identify potential targets. If the program does not provide automated penetration testing, the intruder attempts to hack a modem with unprotected log-ins or easily cracked passwords. Commercial war dialers, also known as modem scanners, are also used by system administrators to identify unauthorized modems on an enterprise network. Such modems can provide easy access to a company's intranet.

Access and Privilege Escalation

Once a potential target network has been profiled and probed for potential vulnerabilities, an attacker must succeed in accessing the target system(s). The primary goal of access is to establish the initial connection to a target host (typically a server platform). In order to conduct additional reconnaissance activities, such as covert installation of hacking tool kits, the attacker must then gain administrative rights to the system.

The method of access depends upon the connection technology necessary to reach the target network. As many organizations evolve to Web-centric business models, they often maintain legacy dial-up access infrastructures, either as secondary remote gateways or due to oversight. In some instances, organizations may not even be aware of modem facilities left connected to outside phone lines or PBXs or may not consider the security risks of leaving unattended modem connections, which often compromise existing network perimeter defenses.

A more recent problem issue is wireless networks which are inadequately configured to restrict access. A very common vulnerability relates to reliance on default settings (SSIDs) and passwords on new installations.

Password Capturing and Cracking

A password logger may be installed as a backdoor Trojan on a target machine and monitor specific protocol and program activity associated with remote login processes. Or, if login strings are captured remotely, a program such as LophtCrack (http://www.atstake.com/research/lc/download.html) might be used to decrypt and compromise administrator and user passwords very quickly.

Malformed Data Attack Crafting commands or scripts containing illegal or unrecognized commands, or incorrectly coded parameters, is a method of attacking a target system once it has been identified. One of the more popular exploits to gain access to Microsoft Web servers, for example, is the IIS Unicode attack, discovered in Microsoft Internet Information Server 4.0/5.0 (as well as the Microsoft Personal Web Server shipped on client systems). Remote users (attackers) can write URLs that allow them to access and run files anywhere on the Web server. Then the attacker can run the "CMD.EXE" file, which allows him to execute any command and have full run of the system.

For example, if an attacker sent the UR: www.victim123.com/scripts/..%c1%1c../winnt/system32/cmd.exe?/c+dir

to a vulnerable server, he would open a command shell and execute the "dir" command. More nefarious commands could be substituted, allowing an attacker to upload a rootkit to deface a site or hijack the site for other attacks.

The string "/..%c1%1c../" translates as "/../ /../," which causes the Web server to go up to the root directory when looking for a file. IIS is smart enough to recognize and block the attack if an attacker tried this in the clear.

However, since the string is obfuscated in Unicode, the server doesn't recognize the attack and readily executes the code. Microsoft issued a patch to fix this, but many older servers exist which are still vulnerable.

Eavesdropping, Data Collection, and Theft

Covert Channel Communication Covert channels refers to hiding malicious code (or simply sending command strings) within normal appearing traffic in such a manner that it is not normally inspected by firewalls, intrusion detection, or other screening countermeasures. As an example, ICMP_ECHO traffic can be used to construct covert communications channels through networks.

The normal PING protocol involves the originating host sending an ICMP_ECHO REQUEST packet to a target destination Host. That destination host then sends an ICMP_ECHO REPLY back. These ICMP_ECHO packets have an option to include additional data about timing information to determine round-trip packet times.

Firewalls and filtering routers only forward or filter these packets based on the protocol itself, but do not inspect the data content, so it is possible to transmit malicious information inside this packet. Because there is no inspection of the content, it is possible to masquerade (hide) Trojan packets inside valid ICMP_ECHO packets. This would be a covert channel attack.

A Remote Administration Tool, or RAT, is a Trojan that, when executed, provides an attacker with the capability of remotely controlling a machine via:

- A "client" in the attacker's machine, and
- A "server" in the victim's machine.

The server in the victim "serves" incoming connections to the victim and runs invisibly with no user interface. The client is a GUI front-end that the attacker uses to connect to victim servers and "manage" those machines. Examples include Back Orifice, NetBus, SubSeven, and Hack'a'tack. What happens when a server is installed in a victim's machine depends on the capabilities of the Trojan, the interests of the attacker, and whether or not control of the server is ever gained by another attacker.

Infections by remote administration Trojans on Windows machines are becoming as frequent as viruses. One common source is through File and Print Sharing. Another common method of installation is for the attacker to simply e-mail the Trojan to the user along with a social engineering hack (compelling message) that convinces the user to run it against his or her better judgment.

Backdoor programs are typically more dangerous than computer viruses as they can be used by an intruder to take control of a PC and potentially gain network access. Until now, the most widely distributed backdoors have been Netbus and the first version of Back Orifice. These programs are also commonly referred to as Trojan horses due to the fact that they pretend to do something other than they actually do.

Backdoor programs are typically sent as attachments to e-mails with innocent looking file names. Back Orifice also has a plug-in architecture that enables it to be disguised upon installation.

Authors of these programs often make the claim that they have not written them as intrusion tools, but rather as remote-control tools or (sometimes) to demonstrate the weaknesses in operating systems. Their real purpose however, as seen by past activity, is to gain access to computers for unauthorized use.

Hackers, Crackers, and Other Perpetrators

What's in a Name?

There are a number of popular descriptive terms used to refer to individuals and groups engaging in exploiting software and system vulnerabilities, although these are by no means standardized. The IT security practitioner should understand the differences. By understanding their motives and methods, you can better protect your systems against malicious code threats from each of these groups:

- Hackers
- Crackers

- Phreakers
- Organized cyber-terrorists and cyber-criminals

Hackers The term "hacker" was initially used by computer programmers to recognize someone who was a computer enthusiast. However, the definition has changed over time to refer to someone who uses computers to commit computer crimes. This change has come about due to the media who have used the term to refer to anyone who gains unauthorized access to computer systems. Hackers maintain the true name for these people is crackers.

To distinguish benevolent from malicious intent, several derivatives have emerged to classify individuals who engage in compromising IT protective measures of third-party organizations:

- Ethical Hackers (sometimes referred to as White Hats)
- Black Hat (corrupt) Hackers
- Wannabes, Grey Hats, or Whackers

White Hat (or Ethical) Hackers are generally described as information security or network professionals using various penetration test tools to uncover and fix vulnerabilities. They are typically hired by concerned organizations to perform their activities with explicit permission of the "victim" in order to identify vulnerabilities and stress test protective measures. Many of these individuals may have actually evolved from less benevolent backgrounds, but have become recognized specialists at discovering and exploiting weaknesses in security systems.

Black Hat Hackers might be described as esoteric hackers who engage in compromising IT security for the mere challenge, to prove vulnerabilities or technical prowess, usually without regard to observing the ethics of who owns which networks. They probe and penetrate target organizations on an ad hoc basis without permission, and their goals often are malicious. Because they most often publicize their exploits to emphasize weaknesses found, it can be argued that these groups contribute something to overall improvements in information security.

Wannabes (also Wnnabes) often refers to "would-be-hackers" capable of becoming black hat, who investigate and study the IT security field and perform general reconnaissance of systems and networks as a prelude to possibly more devious undertakings.

There is another popular term for this type of hacker — Grey Hats. These people may ultimately become either criminals or security consultants.

The term "whacker" has also recently been recirculated to refer to an emerging group of hackers focusing on attacking wireless networks.

The "Cracker," who is primarily defined by hostile intent and sophisticated skills, may have any one of dozens of motivations, not the least of which is financial gain.

System Crackers are the individuals and groups (whether organized or not) who pose the greatest threat to networks and information resources and who are actively engaged in developing and propagating viruses, worms, and Trojans. They may also engage in interactive (real-time) probing and reconnaissance activities by exploiting common security flaws in network operating systems and protocols. Once they discover a security weakness, they often script the exploit and create automated tool-kits for others to reuse and improve upon. They frequent popular news groups and Web sites to "share" exploits and tools. As their success and notoriety increase, their exploits usually escalate to more sophisticated and damaging activities.

Program crackers are technically skilled individuals who take commercial application software programs and "crack" their protections, usually by altering the programs or by reverse engineering them to see how they work and then creating bogus registration keys. They generally limit themselves to stealing popular software, perhaps distributing it to others as black market products or simply giving it away free of charge. These "cracked" programs may actually contain poorly modified (even viral or Trojan) code and thus pose a serious concern for IT Security Practitioners. Beyond the significant legal implications of using pirated software, users should be educated to avoid these "WAREZ," which may also contain malicious code.

Script Kiddies is the name given to less sophisticated, often younger crackers, who generally rely on automated tools ("scripts") written by the more skilled system crackers. They show no bias and scan all systems, regardless of location and value. Their methodology is a simple one, scan the Internet for a specific weakness, and when you find it, exploit it. Since most of the tools they use are automated, requiring little interaction, once they gain access to a network, they will launch the tool(s), then return later to get the results. Their exploits are much less successful if organizations disable unnecessary services and ensure that security patches are up to date.

Click Kiddies is a newer term coined to reflect the enhancements due to GUI-based point and click software.

Phreakers are persons who break into telecommunications networks (telephone and cellular) to exploit and illegally use the provider's services. This includes physical theft and reprogramming of equipment, as well as compromising codes and other mechanisms to gain unauthorized use of facilities.

Cyberpunks, Cyber-Criminals, and Cyber-Terrorists may represent composites of all of the above descriptions and may be engaged in coordinated, potentially state-sponsored acts of defacement, denial of service, personal identity and financial theft, or worse, compromise of government or industrial secrets.

There are a number of known, organized, hostile groups engaged in ongoing exploitation of the Internet on a global basis. There is mounting evidence that malicious cyber-activity by various terrorist organizations is also on the rise. As these groups have become more organized and sophisticated, government, law enforcement, and intelligence agencies in the United States and many other countries have increased coordination and collaboration efforts to monitor and respond to these threats.

These types of activities have been limited mostly to individuals claiming to represent groups. However, the overall global security environment is becoming much more problematic for IT security practitioners as these activities require constant vigilance.

Where Do Threats Come From?

Employees While the more popular security threats are initiated from outside a corporate network, a number of significant vulnerabilities still exist inside a trusted network and require the IT security practitioner's attention.

Probably the most common vulnerabilities are exploitable due to unsafe computing practices by employees such as:

- Exchange of untrusted disk media and files among host systems
- Installation of unauthorized, unregistered software (application and OS)
- Unmonitored downloading of files from the Internet
- Uncontrolled dissemination of e-mail attachments, which may release malicious code
- Other unrestricted and unsafe use of network resources

In addition to unsafe practices, traditional security breaches, both reported and unreported, have originated from within the victim organization, perpetrated by current and former employees and are often undetected due to weak personnel and security policies or ineffective countermeasures, or unreported by the organization involved. These include:

- Unauthorized access to system and network resources
- Privilege escalation
- Theft, destruction, and unauthorized dissemination of data
- Use of corporate network to initiate hostile code attacks against outside target

Social Engineering

Exploits from the Internet

While the more popular security threats relate to intentional hostile attacks or unsafe security practices, another growing concern is the unscrupulous practices engaged in by both legitimate and illicit marketing activities.

This includes "spam" (high-volume, bulk unsolicited e-mail), exploitation of active-code browser plug-in features such as "pop-up" ads and home page hijacking, and the more insidious proliferation of spyware (Trojan-like) applications propagated to monitor and track Internet use.

While specific malicious intent may not be the primary motivation, the negative organizational impact on network productivity and service availability is becoming a key business issue among those companies heavily committed to Internet-based commerce.

One of the current phenomena is personal identity theft and financial destabilization by organized groups hacking into banking and other financial networks and stealing account and other personal information for use in financing criminal and terrorist activities. Propagation of Trojan password sniffers and other keystroke monitoring software through e-mail, instant messaging software, and other pervasive personal computing applications has increased over the past few years.

Spyware and Adware Any unsolicited background process that is installed on a user's computer when he or she visits a Web site for the purpose of collecting information about the user's browsing habits and activities is considered spyware. These programs are generally installed when users download freeware programs and they impact privacy and confidentiality. Adware programs, which trigger such nuisances as pop-up ad pages and banners when users visit certain Web sites, impact productivity, and may also be combined with techniques like home-page hijacking code and other active background activities.

Spam According to Ferris Research (http://www.ferris.com), spam will cost U.S. organizations over $10 billion in 2003. For U.S.-based ISPs, 30 percent of inbound e-mail is spam, while at U.S.-based corporate organizations, spam accounts for 15–20 percent of inbound e-mail. Despite the increasing deployment of anti-spam services and technology, the number of spam messages, and their size, is continuing to increase rapidly. Some more recent research in this area indicates even higher percentages of unwanted e-mail traffic.

While not specifically malicious code, spam represents the following threats to organizations:

- Spam consumes computing resources (bandwidth and CPU time).
- Spam diverts IT personnel's attention from more critical network security efforts.
- Spam e-mail is a potential carrier of malicious code attachments.

Spammers have developed techniques to compromise intermediate systems to facilitate "re-mailing" services, masking the real source addresses and constituting a denial of service for victimized systems. Opt-out (unsubscribe) features in spam messages can represent a new form of reconnaissance attack to acquire legitimate target addresses.

How Can I Protect against These Attacks

Defense in Depth is the practice of "layering" defenses into defensive zones to increase overall the protection level and provide more reaction time to respond to incidents. It should be designed such that a failure in one safeguard is covered by another. This combines the capabilities of people, operations, and security technologies to establish multiple layers of protection, eliminating single lines of defense and effectively raising the cost of an attack. By treating individual countermeasures as part of an integrated suite of protective measures, the IT security practitioner is able to ensure that all vulnerabilities have been addressed. Managers must strengthen these defenses at critical locations and then be able to monitor attacks and react to them quickly. With respect to malicious code threats, these layers of protection extend to specific critical defensive zones (see Figure 7.6):

- Application defenses
- Operating system defenses
- Network infrastructure defenses

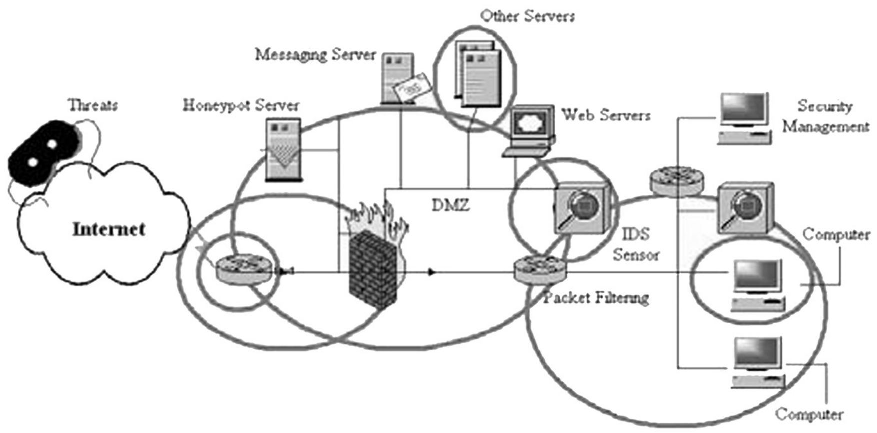


Figure 7.6 Critical defensive zones

While "defense in depth" represents a much broader issue than just protecting against forms of malicious code, this approach means:

- Creating multiple layers of security and detection capability, even on single systems (AVP, IDS, restrictive settings in applications, etc.)
- Implementing acceptable use policies and monitoring for compliance
- Limiting methods of acquiring data and program code (media restrictions)
- Using the other defense in depth measures depicted in the graphic to add robustness of the overall system of safeguards

This graphic depicts a number of countermeasures that might be deployed to create multiple zones of defense. They include techniques like screening routers, firewalls, intrusion detection, anti-virus protection, honeypot techniques, and other measures used to add layers of protection.

Application countermeasures (see Figure 7.7) include things such as hardening of applications, anti-virus and IDS protection on hosts and servers, host-based intrusion detection, and network security monitors.

Application Defenses

- Educate users on malware in general and implement acceptable use policies.
- Implement regular anti-virus screening on all host systems and network servers and ensure that virus definition files are kept up to date.

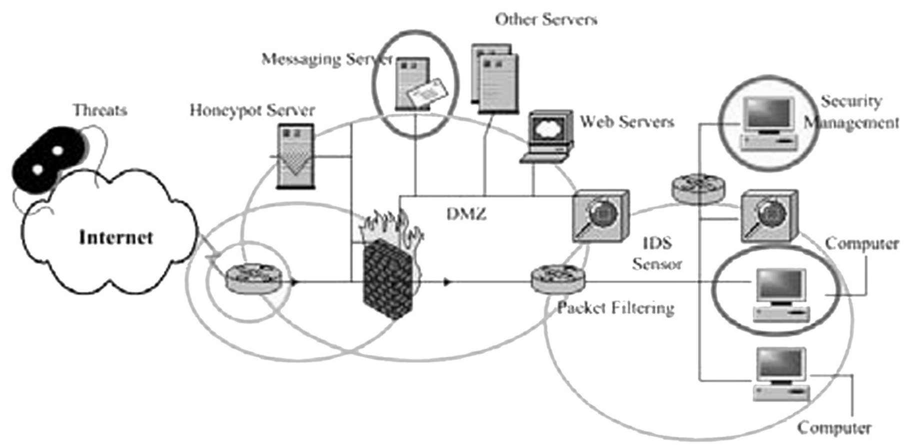


Figure 7.7 Application countermeasures

- Require scanning of all removable media and e-mail (especially attachments.
- Consider installation of personal firewall and IDS software on hosts as an additional security layer.
- Deploy change detection/integrity checking software and maintain logs.
- Implement e-mail usage controls and ensure that e-mail attachments are scanned.
- Deploy specialized anti-malware software and e-mail filters to detect and block unwanted traffic (anti-spam filters, etc.).
- Establish a clear policy regarding new software development/engineering practices, installations and upgrades.
- Ensure only trusted sources are used when obtaining, installing, and upgrading software, through digital signatures (e.g., authenticode and other validations).

Operating system countermeasures (see Figure 7.8) such as hardening of operating systems include all devices involved in network communications, including routers and switches, servers, as well as hosts. The practitioner should regularly check to ensure that the latest security patches are deployed.

Operating System Defenses (Hardening the OS)

- Deploy change detection software and integrity checking software and maintain logs.

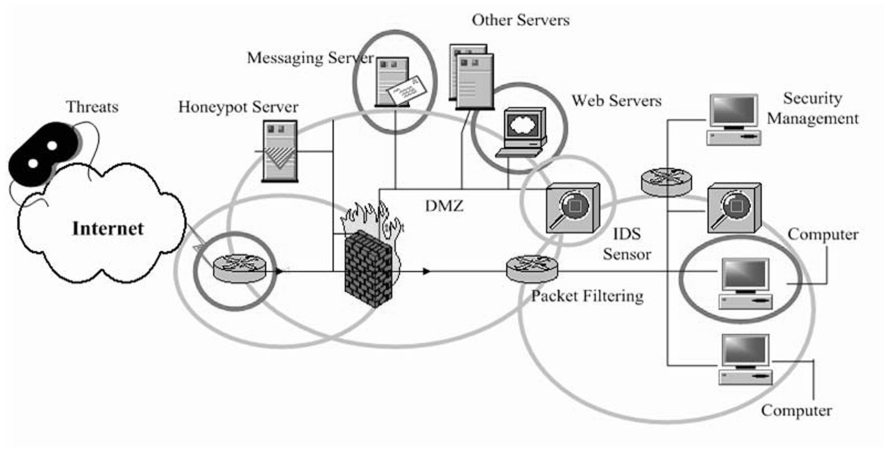


Figure 7.8 Operating system countermeasures

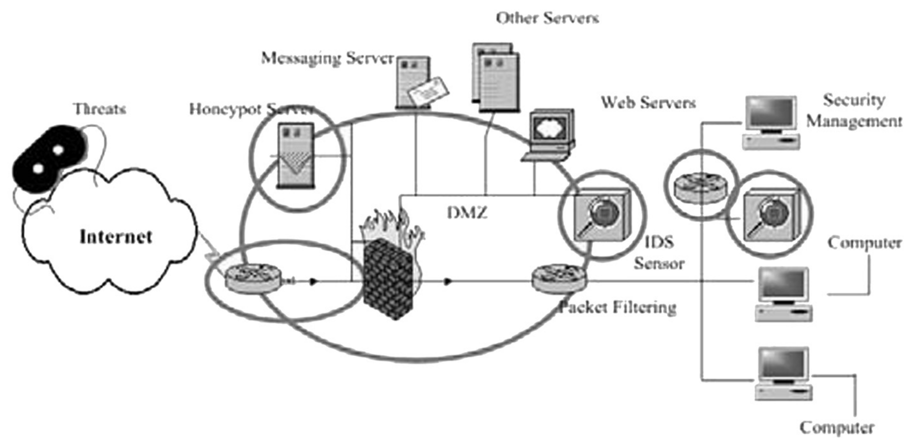


Figure 7.9 Network infrastructure defenses

- Deploy or enable change detection and integrity checking software on all servers.
- Ensure that all operating systems are consistent and have been patched with the latest updates from vendors.
- Ensure only trusted sources are used when installing and upgrading OS code.
- Disable any unnecessary OS services and processes which may pose a security vulnerability.

Network infrastructure defenses (see Figure 7.9) involve identifying where the traffic patterns are and where to deploy various countermeasures. The red circles indicate some of these locations, and the devices which might provide elements of protection.

Network Infrastructure Defenses

- Create choke points in the network.
- Use Proxy services and Bastion hosts to protect critical services.
- Use content filtering at chokepoints to screen traffic.
- Ensure only trusted sources are used when installing and upgrading OS code.
- Disable any unnecessary network services and processes that could pose a security vulnerability.
- Maintain up-to-date IDS signature databases.
- Apply security patches to network devices to ensure protection against new threats and to reduce vulnerabilities.

Incident Detection Tools and Techniques

Intrusion Detection Systems

Intrusion detection tools are an integral component of Defense in Depth and should be deployed in critical areas of the network as an early warning system. There are a variety of implementations and each has features that provide unique capabilities to protect networks and hosts from malicious activity.

Intrusion detection is designed to quickly recognize a security event (malicious code attack, denial of service attack, network reconnaissance attack, and the like) so that immediate countermeasures can be executed to isolate and react to the event.

Intrusion detection can be deployed on separate appliances called sensors that are managed by a security server, or they can be deployed as software on the hosts within the network. Each type has advantages and disadvantages.

Network-Based Intrusion Detection is typically deployed on dedicated appliances and is independent of the operating systems being run on network hosts and servers. This tool is available in a range of price-performance classes for small to enterprise networks.

To minimize the risk of compromise, network-based intrusion detection platforms use hardware sensors that are ideally connected at key choke points in the network infrastructure. The physical connections to the monitored network are passive (promiscuous) and often nonaddressable (to avoid attack and compromise).

Each sensor has a second connection to a special command-and-control network that hosts the IDS management server. When a network attack profile is detected, sensors alert the management server and begin logging the suspected traffic.

Some configurations permit active attack responses, such as IP traceback, transmitting TCP session resets to terminate the activity, or reconfiguring firewall or router access lists dynamically (shunning).

One of the weaknesses of network-based sensors is that they cannot normally analyze encrypted host traffic or easily recognize attack profiles in fragmented packets.

Host-Based Intrusion Detection is typically implemented in software on individual hosts in the network. These programs might be installed as a component of an overall personal computer anti-virus/firewall/IDS package, or as a client/agent component of a security server suite.

An advantage to host-based detection is that it can detect and log the success or failure of an attack against the host. However, these systems are limited in perspective to the host and may not detect general attacks against other network resources.

They are also OS dependent and, to be effective, must be deployed on all network hosts.

Signature-Based IDS profiling involves code pattern matching against a database of known attack profiles. This requires the IDS signature database to be continuously updated for new signatures.

The advantages of signature-based detection are a lower incidence of False Positive detections (misdiagnosing a benign event as something malicious). False Positives can result in a high volume of unnecessary alarms being triggered, which tend to encourage administrators to lower triggering thresholds to compensate.

One disadvantage due to the increasing number of new virus strains and those that utilize stealth or polymorphic features, is the potential for False Negatives, that is, incorrectly missing a new form of malicious code that is not recognized by the sensor. In addition, these types of detection algorithms require regular updating to ensure that the latest threats are recognized.

Anomaly Detection is based on developing a network baseline profile of normal or acceptable activity (such as services or traffic patterns), then measuring actual network traffic against this baseline.

This technique might be useful for detecting attacks such as denial of service or continuous login attempts, but it requires a learning or preconfiguration period.

An advantage to anomaly based detection is that events that are unusual or out of the ordinary tend to prompt administrators to focus on the network and attempt to understand traffic patterns and normal behavior, and exception events may receive quicker attention. Another advantage is that new forms of malicious code, and new attack methodologies may be recognized quicker.

The main disadvantage of anomaly-based detection is the higher incident of False Positives. New application deployments or changes in traffic flow may trigger alarms which require attention. The tendency of administrators to lower sensitivity levels (alarm thresholds) to reduce this overhead can lower the overall effectiveness of the countermeasure. Baselining what is normal is usually a cumbersome task as well.

A layered defense in depth approach (see Figure 7.10) would suggest deploying both network-based and host-based intrusion detection and products that permit both signature-based and anomaly-based detection schemes.

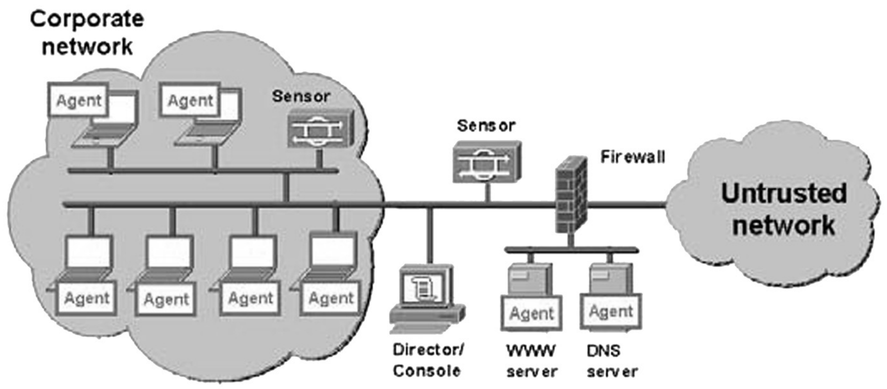


Figure 7.10 Layered defense in depth

Anti-Virus Scanning Software Most PC users today use some form of virus protection to detect and prevent infection. Just as intrusion detection can be layered at the host and network levels, anti-virus protection should be deployed on all devices that support the programs. The key vulnerabilities to host-based anti-virus software are:

- The continuing requirement to keep every host system updated to the most current virus definition files, and
- Potential compromise of the protection through unsafe user practices (such as installing unlicensed or unauthorized software or indiscriminately exchanging infected e-mail or document files).

Network-based anti-virus software is an option that permits screening of files and e-mail traffic at the servers and providing remote scanning and inoculation of clients on a consistent basis.

Many organizations employ both network and host-based protection; some deploy multiple products in order to maximize detection capabilities. It is imperative, however, that virus definition files be kept up-to-date. Most vendors now offer automatic updating of software as soon as new definitions are added.

Types of Anti-Virus (Anti-Malware) Software

There are a variety of anti-virus software packages that operate in many different ways, depending on how the vendor chose to implement their software. What they have in common, though, is that they all look for patterns in the files or memory of your computer that indicate the possible presence of a known virus. Antivirus packages know what to look for through the use of virus profiles (sometimes called "signatures") provided by the vendor.

- New viruses are discovered daily. The effectiveness of antivirus software is dependent on having the latest virus profiles installed on your computer so that it can look for recently discovered viruses. It is important to keep these profiles up to date.
- First generation anti-virus scanners used brute force to analyze every byte of data in boot records and files looking for known patterns and strings associated with virus activity. This method is obviously time consuming and, with the number of known and new virus strains around, this became obsolete, in favor of more intelligent algorithms that searched specific portions of files where virus code typically resides. Although these scanning approaches are still used today, other more efficient techniques (like algorithmic code entry point scanning) are used to combat newer forms of viral code such as polymorphic strains.
- Generic Decryption (GD) is a newer technique designed to use a virtual machine (isolated and controlled) environment to trick a polymorphic virus into decrypting itself and exposing recognizable viral code components. As long as there is at least a small portion of the malicious code in machine language that can be executed in a virtual environment, and the code successfully decrypts and transfers control to the resulting virus instructions, this type of AVP can usually detect it. Most new viral code, even polymorphic varieties, generally are iterations of previous generation code.

Heuristic scanners operate by looking for telltale signs or patterns of behavior consistent with known virus activity, and then logging this and alerting the user to its presence (allowing the user to make a final decision on eradication). This is how many Anti-Spyware products work as well. There are also schemes called behavior blocking that monitor system calls and other signs of activity which might indicate the presence of viral code. Many of these will isolate the offending code and prompt the user to make a decision or link to additional descriptive information on the vendor's Web site.

Data integrity tools are also important tools for discovering whether any files have been modified on a system. This is useful for protecting systems against computer viruses because integrity checkers do not require updating signature files to detect computer viruses. When an integrity checker is installed it creates a database of checksums for a set of files.

The integrity checker can then tell if files have been modified by comparing the current checksum to the checksum it took when installed. If the checksums do not match, then the file has been modified in some manner. Some integrity checkers may be able to identify the virus that modified a file, but others may just be able to alert you to the changes. Integrity checkers are not only useful for detecting a possible infection, but also useful for helping to detect intruders.

Network Monitors and Analyzers In order to ensure that security practices remain effective, IT security practitioners should consider regular use of network monitoring software or appliances, as well as periodic analysis of network traffic.

Periodically run a security scanning tool such as the Internet Scanner, Nessus, Satan, Trinux, or some combination of all of them, depending upon the operating systems in use. Keep in mind that these same tools can also be used by attackers.

In addition, on a regular basis, the network should be scanned for unnecessary open service ports, as upgrades to software often reset default settings.

Content/Context Filtering and Logging Software Privacy and security must be balanced when implementing countermeasures that are designed to screen content; however, when combined with a clear corporate policy on acceptable use, these become additional layers of defense against malicious code. One of the more popular countermeasures in this category today is Content Filtering, which allows management to control Internet use.

Plug-ins to screen e-mail attachments and content- and context-based filtering (access control lists) on network routers also permits an additional layer of security protection.

Content-based filtering includes analyzing network traffic for active code (Java, Active-X) components and administrative disabling of script processing on Web browser software. Context-based filtering involves comparing patterns of activity to baseline standards, so that unusual changes in network behavior can be evaluated for possible malicious activity.

Other Techniques to Actively Detect and Analyze Hostile Activity Honeypots are sacrificial hosts and services deployed at the edges of a network to act as bait for potential hacking attacks. Typically, the systems are configured to appear real and may be part of a suite of servers placed in a separate network (honeynet), isolated from the real network. The purpose of the honeypot is to provide a controlled environment for attacks to occur so they can be easily detected and analyzed to test the strength of the network. Host-based intrusion detection and monitoring software is installed to log activity.

A honeypot is a tool intended to be compromised. All traffic to and from the honeypot is suspicious because there are no production applications on this system. Few logs should be produced on the honeypot unless the honeypot is under heavy attack. Logs should be easy to read and understand. Once a production honeypot is probed or attacked, an administrator can place preventive controls on his "real" production network.

Honeypots should contain at least the following elements:

- Looks and behaves like a real host
- Does not disclose its existence at any point
- Has a dedicated firewall that prevents all outbound traffic, in case the honeypot is compromised
- Lives in a network DMZ, untouched by normal traffic
- Sounds silent alarms when any traffic goes to or from it
- Begins logging all intruder activity when it first senses an intrusion

Classes of Honeypots

- Low involvement
- High involvement
- Honeynet

A *low involvement* honeypot provides a number of fake services such as HTTP (Hyper Text Transfer Protocol) or SMTP (Simple Mail Transfer Protocol). Low involvement honeypots allow hackers to connect to services, but do nothing else. With this type of honeypot a hacker usually cannot gain operating system access and, therefore, poses no threat.

A *high involvement* honeypot produces genuine services and vulnerabilities by providing a real operating system for the attacker to interact with. This class of honeypot is designed to be compromised so that realistic data can be collected. The difficulty in high involvement honeypots is they must be tightly controlled. A compromised system can become a host to begin an attack on another system.

Honeynets are a group of honeypots made to simulate a real live network. There is added value in honeynets as they provide more data and are more attractive to hackers. However, the set-up and maintenance of honeynets are a little more advanced. A honeynet may include many servers, a router, and a firewall. A honeynet may be identical to the production network or it may be a research lab. Nonetheless, honeynets allow for a more real environment for a hacker to attack.

Attack Prevention Tools and Techniques

One of the simplest prevention techniques is to disable unnecessary network services, especially certain TCP and UDP listening ports. This will defeat any attack that is aimed at exploiting those services. This may not be efficient if those services are required for legitimate users, however. The optimum solution is to ensure availability of services to legitimate users while minimizing vulnerabilities.

By creating perimeter zones of defense, techniques can be deployed which are preventative, detective, or deterrent. Through the use of honeypots or "faux" resource environments, attacks can also be deflected to controlled and monitored zones where they can be analyzed and used to strengthen controls.

As countermeasures are deployed, each countermeasure depends upon the effectiveness of the layer outside it. Ideally, the defensive zone starts outside the network at the entry points. Simple packet filtering reduces the amount of traffic which then needs to be inspected by the firewall. The firewall, through selective and stateful filtering, then limits the number of packets that an intrusion detection sensor needs to analyze, and so on. Finally, host-based preventative software (ant-virus protection, etc.) is one of the last layers protecting network hosts.

There are a wide variety of countermeasures (see Figure 7.11) and practices that can assist in preventing and detecting malicious code attacks, including:

- Employing filtering software that blocks traffic to and from network segments or specific services that will prevent those resources from being accessed and exploited
- Employing active sensors (intrusion detection, anti-virus detection) that react quickly enough to prevent or mitigate damage

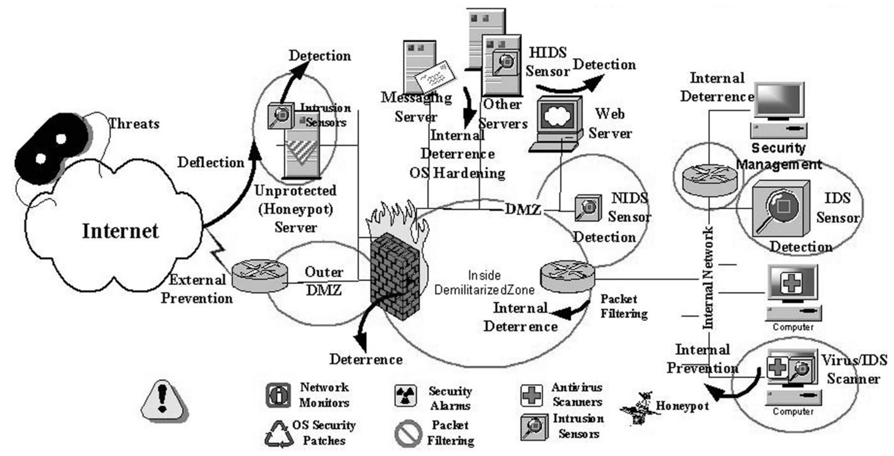


Figure 7.11 Countermeasure techniques

- Employing chokepoints in the network to force traffic to flow through zones of protection, allowing sensors and filters to inspect traffic before allowing it to be passed into the protected network
- Setting security properties within browsers to prohibit or prompt before processing scripts and active code
- Eliminating unnecessary remote connections to networks and employing effective access control measures to protect those connections required to remain open or available
- Avoid circumventing any existing control systems and countermeasures

Safe Recovery Techniques and Practices

Once an event has occurred, it is often difficult or impossible to restore the network resources to their original condition, unless steps are taken beforehand to facilitate quick recovery capability. File backups allow you to restore the availability and integrity of data. Some of the steps that should be taken are:

- Consider storing OS and data file backup images on CD-ROM to prevent possible virus infection.
- Backup all critical configuration files on network devices and servers
- Ensure that new and replacement media is scanned for viruses before re-installation of software.
- Disable network access to systems being restored or upgraded until protection software or services have been re-enabled or installed.

The following is taken from CERT regarding backups (http://www.cert.org/security-improvement/practices/p071.html):

Develop a File Backup and Restoration Plan All system and user files should be backed up on a regular basis. If you have regularly created cryptographic checksums for all files and have securely stored these checksums, you can plan to restore files from trusted backups against which such checksums have been calculated.

Then you must reinstall site-specific modifications, relevant patches, and bug-fixes. You need to ensure that these modifications do not introduce additional defects or vulnerabilities.

Exercise caution when restoring user files from untrusted backups and instruct all users to check for any unexpected changes to their restored files.

For workstations, files are backed up locally at each workstation, often by the user(s) of that workstation and then centrally administered.

For network servers that provide information services backups of the entire information content, OS and application suite; should be backed up on a separate and secure machine. Data is then backed up on a regular basis.

Determine the appropriate medium to contain your backup files based on your requirements for speed (for both reading and writing), reliability, and storage duration. Media you should consider include magnetic tape, optical disk, and CD-ROM.

The plan should specify that

- Source data is encrypted before being transmitted over a network or to the storage medium.
- Data remains encrypted on the backup storage media.
- Storage media are kept in a physically secure facility that is protected from manmade and natural disasters.

The plan should be designed to ensure that backups are performed in a secure manner and that the contents of the backups remain secure.

Install File Backup Tools

- Select file backup tools to allow you to implement your backup plan. You may need to use third-party software, although the backup capabilities of some operating systems are likely to be sufficient. You may also need to install storage devices, either centrally or on each workstation and server, to store the backup copies.
- The tools used to recover backed-up files should be kept offline, rather than on individual workstations and servers. If a computer has been compromised and you need to recover a file, you cannot trust the integrity of any of the tools on that computer.

Configure the Backup Tools and Initiate the Scheduled Backups

- Tool configurations need to reflect your backup and restoration plan. Configure the tools to save access control settings along with file contents, if that feature is available.
- Do the first full backup just before deploying the computer, and then confirm that you can perform a full restoration from that backup.

Confirm that the Scheduled Backups Are Being Performed Successfully

- In many organizations, file backups are completely automated, so system administrators tend to forget that they are happening. Therefore, confirm that the backup procedures for a newly deployed workstation are actually working.

Test the Ability to Recover from Fackups

- For many system administrators, recovering a file from a backup is an uncommon activity. This step assures that if you need to recover a file, the tools and procedures will work.

- Performing this test periodically will help you to discover problems with the backup procedures so you can correct them before losing data.
- Some backup restoration software does not accurately recover the correct file protection and file ownership controls. Check these attributes of restored files to ensure they are being set correctly.
- Periodically test to ensure that you can perform a full system recovery from your backups.

Policy Considerations

Your organization's security policy for networked systems should:

- Require the creation of a file backup and recovery plan.
- Inform users of their responsibilities (if any) for file backup and recovery.

Implementing Effective Software Engineering Best Practices

The organization should adopt an Acceptable Use Policy for network services and resources. This would include prohibitions on certain network activities and PC user habits regarding software licensing and installation and procedures for transmitting files and media. Adopt standardized software so that patches and upgrades can be controlled to ensure vulnerabilities are addressed.

Consider implementing an ISO 17799 compliant security policy. ISO 17799 is one of the most widely recognized security guidelines. Adoption of ISO/IEC 17799 (or indeed any detailed security practice) as an internal standard, can be a far from trivial undertaking, even for the most security conscious of organizations. Security practitioners should evaluate those practices which make business sense to the organization, and keep in mind that policies need senior management commitment to be effective.

Sample Questions

1. Which of the following is a reason to use a Firewall?
   a. To evaluate intrusions as they happen
   b. To watch for internal attacks
   c. To prevent or stop potential intrusions
   d. To signal an alarm on a suspected intrusion
2. Which of the following is an attribute of polymorphic code?
   a. It mutates while keeping the original algorithm intact
   b. It uses encryption for all the code

c. It operates based on specific conditions
d. It is written to self replicate altering the computer configuration

3. Flooding network ports is an example of which type of attack?
   a. Man-in-the-Middle
   b. Brute force
   c. Denial of service
   d. Birthday

4. Which of the following is typically a characteristic exhibited by Script Kiddies?
   a. They are very talented.
   b. They leave no trace of what they are doing.
   c. They are concerned with the quality of the attack.
   d. They see the number of attacks being something to be proud of.

5. Which of the following virus types typically affects the normal.dot file?
   a. Macro virus
   b. Multipartite
   c. Boot sector
   d. MacIntosh

6. Which of the following represents the "SALT" value in a password?
   a. The value is a constant chosen by the system.
   b. The value is chosen randomly.
   c. The value is time based.
   d. The value is a constant chosen by the user.

7. Which of the following techniques BEST defines "Spam"?
   a. Filling a field with false information
   b. Expressing a strong criticism of something
   c. Inducing a resource to take incorrect actions
   d. Posting information repeatedly to overburden the network

8. Which of the following defines a Man-in-the-Middle attack?
   a. Searching through databases for specific information
   b. Bypassing the user authentication system
   c. Manipulation of packets being sent over a network
   d. Overwriting the system buffers with data

9. Which of the following features of the Internet Protocol is used by the Ping of Death to execute?
   a. Its ability to fragment packets
   b. Its capability to encrypt data
   c. Its ability to do site routing
   d. Its capability to parse log files

10. Which of the following defines a Trojan Horse?
    a. It has the ability to gain access to a computer system by circumventing the usual security checks.
    b. It has the ability to self-propagate from one computer to another on the network.
    c. It has the ability to replicate itself between files on the system.
    d. It has the ability to convince the user that is has one function whilst hiding another function.
11. All of the following EXCEPT are forms of Symmetric Block Cipher Attacks?
    a. Linear cryptanalysis
    b. Weak keys
    c. Key streams
    d. Algebraic attacks
12. Which of the following is NOT a phase of a Social Engineering attack?
    a. Intelligence gathering
    b. Eavesdropping
    c. Target selection
    d. The attack
13. Which of the following can be susceptible to a Birthday attack?
    a. Digital signatures
    b. Asymmetric algorithms
    c. Password hashs
    d. Private keys
14. Which of the following defines session hijacking?
    a. Compromising the session between two machines is taken over
    b. Exploitation of the trust relationship in the Internet Protocol
    c. Creation of random addresses for each network packet
    d. Making compute resources unavailable
15. Which of the following is NOT a characteristic of Computer Virus Hoax?
    a. Asks that you send it to everyone to warn them
    b. Makes reference to a false authority
    c. Contains technical sounding nonsense
    d. Usually has a hidden function that executes unknown to the user