Chapter 1

An Overview of Electronic Attacks

Thomas M. Chen, Southern Methodist University, USA

Chris Davis, Texas Instruments, USA

Abstract

This chapter gives an overview of the major types of electronic attacks encountered today and likely to continue into the foreseeable future. A comprehensive understanding of attackers, their motives, and their methods is a prerequisite for digital crime investigation. The range of possible cyber attacks is almost unlimited, but many attacks generally follow the basic steps of reconnaissance, gaining access, and cover-up. We highlight common methods and tools used by attackers in each step. In addition, attacks are not necessarily directed toward specific targets. Viruses, worms, and spam are examples of large-scale attacks directed at compromising as many systems as possible.

Introduction

Today computer systems are often invaluable for business and personal uses. Computer systems store valuable corporate and personal information while computer networks provide convenient data access and processing services. They are naturally very tempting targets, as shown by statistics that track the frequency and prevalence of cybercrimes. For example, an CSI/FBI survey found that 71 percent of organizations had experienced at least one attack in 2004, while the remaining organizations did not know the number of attacks (Gordon, 2005).

The ease of carrying out electronic attacks adds to the temptation for attackers. It is widely known that computer systems have numerous vulnerabilities, although not every attack exploits vulnerabilities (Hoglund & McGraw, 2004). In the second half of 2004, 54 new vulnerabilities per week were discovered on average, and 50 percent were serious enough to be rated as highly severe, meaning that exploitation of the vulnerability could lead to complete compromise of a system (Turner, 2005). Attackers are keenly aware of new vulnerabilities because it takes time for organizations to set up adequate protection. New vulnerabilities are announced along with a software patch, but organizations are sometimes slow to apply patches. In late 2004, exploit codes for new vulnerabilities appeared on average only 6.4 days after the announcement of the vulnerability; in early 2004, it was 5.8 days. Organizations that are slow to patch are often vulnerable to new exploits.

Attackers are also well aware that virtually all computers are interconnected by the Internet or private networks. Moreover, mobile and handheld devices with Internet connectivity have steadily grown in popularity. Networks make attacks easier to carry out remotely and more difficult to track to their sources.

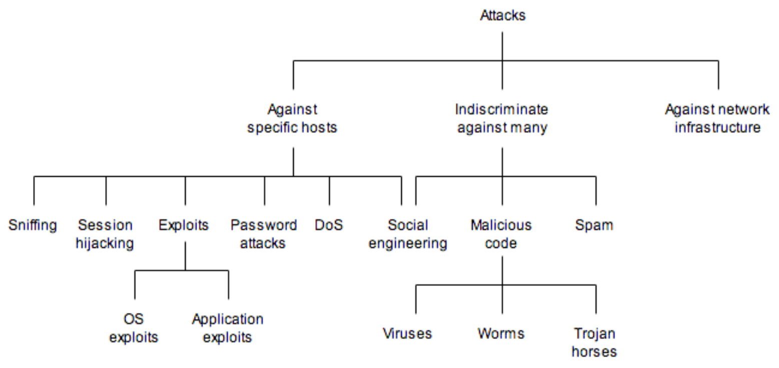
This chapter gives an overview of electronic attacks, organized according to the basic steps of reconnaissance, gaining access, and cover-up. We focus here on network-enabled attacks, but this is not meant to imply that all electronic attacks are carried out remotely. Direct physical attacks on computers are also quite common but not covered here. This chapter also describes large-scale attacks such as viruses, worms, denial of service, and spam. An understanding of attackers and their attack methods is a prerequisite to digital forensics, which is concerned with the collection and analysis of evidence of electronic crimes. This chapter serves as necessary background for other chapters in this book that cover aspects of digital forensics in depth.

Types of Attackers and Motives

As one might expect, there are as many different types of attackers as there are different types of attacks. Attackers can be categorized in a number of different ways. For example, attackers may be either internal or external, depending on their relationship to the target. In the past five years, the fraction of attacks from inside have been roughly equal to the fraction from outside (Gordon, 2005). Insiders are worrisome because they have certain advantages such as trust and knowledge of the target organization that can increase the chances of a successful attack. Moreover, insiders do not have to overcome perimeter defenses designed for external attackers.

Attackers can also be viewed as amateurs or professionals. Many people probably visualize an attacker as the stereotypical male teenage “hacker” perpetuated by the mass media. While amateur hackers are undoubtedly responsible for a substantial fraction of viruses and worms and other vandalism, the involvement of professionals and perhaps organized crime is suggested by the sophistication of attacks and number of attacks apparently driven by profit motives (Swartz, 2004). Besides professional hackers, other professionals involved in electronic attacks include national governments, military agencies, and industrial spies.

Figure 1. A taxonomy of attacks



The motivations for electronic attacks depend on the attacker. Because there are many different types of attackers, motivations can be almost anything ranging from fun and fame to extortion, profit, espionage, revenge, or a political agenda (Shinder & Tittel, 2002).

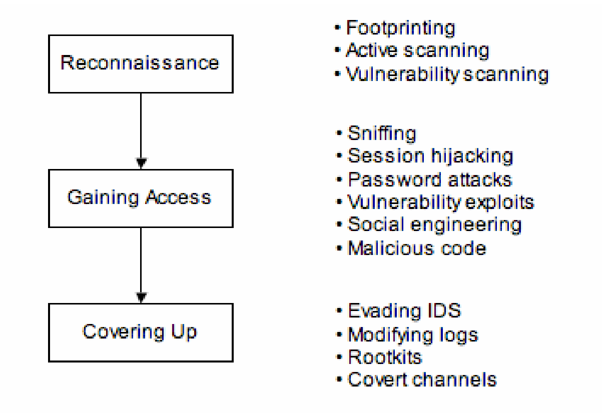
The stereotypical teenage hacker is believed to be usually interested in gaining fame or notoriety. On the other hand, organized crime and white collar attackers are more interested in profit. Attacks oriented towards invasion of privacy or theft of confidential data is a growing trend, as evidenced by an escalation in spyware and phishing attacks (described later in this chapter). Cyber attacks for political purposes have become a growing concern since international attention has turned to terrorism.

Types of Attacks

A taxonomy of attacks is offered in Figure 1. At the highest level, attacks can be targeted against specific hosts, the network infrastructure, or indiscriminately at as many hosts as possible. This chapter does not cover attacks against infrastructure; the interested reader is referred to the literature (Chakrabarti & Manimaran, 2002).

Attacks directed at specific hosts include sniffing, session hijacking, exploits of vulnerabilities, password attacks, denial of service, and social engineering. Social engineering can also be used in large-scale indiscriminate attacks. Other large-scale attacks include spam and malicious code (otherwise known as malware). Each of these attack methods are described later in this chapter.

Figure 2. Basic steps in attacks against specific targets



Attack Phases

An attack to compromise a particular target is often carried out through a progression of steps, analogous to the steps of a physical attack (Chirillo, 2002; McClure, Scambray, & Kutz, 2001; Skoudis, 2002). As shown in Figure 2, the first step is reconnaissance to collect intelligence in preparation for attack. Knowledge of a target and its vulnerabilities can be critical to the success of an attack. The second step is gaining access, which could have many different goals such as control, theft, or destruction. During and after the attack, the attacker may take actions to try to avoid detection, such as changing system logs or installing a rootkit. We elaborate on each step in the remainder of this chapter.

Reconnaissance

In order to prepare for a successful attack, it would be common sense to first try to learn as much as possible about the target. The reconnaissance phase can reveal a surprising amount of information such as account names, addresses, operating systems, and perhaps even passwords. Moreover, most reconnaissance techniques are not viewed as malicious or illegal, and can be carried out relatively safely. Reconnaissance activities are so common that potential targets may not be alarmed.

Many different reconnaissance techniques are possible, and attackers do not follow a unique sequence of steps. We outline three general steps subsequently to progressively discover more information about a potential target. First, footprinting attempts to learn the location and nature of a potential target from public directories. Second, scanning provides more detailed information about a target by active probing.

Footprinting

The initial step in discovery is footprinting (also known as fingerprinting or enumeration) with the primary objective of locating and learning the nature of potential targets. For example, an attacker will want to know how many potential hosts are available and their IP addresses.

An abundant amount of information is readily available on the Web in large public databases. These databases can be interrogated by a number of utilities such as nslookup, whois, or dig (Kloth.net, 2005). Many of these databases have easy-to-use interfaces and do not require any advanced technical knowledge. In general, the information gained in footprinting is common, easily found, and presents a very low risk to corporate, government, and military entities.

The whois databases contain data about the assignment of Internet addresses, registration of domain names, and contact information. Domain names such as www.company.com are registered through the Internet Network Information Center (InterNIC), a consortium of several companies and the U.S. government (InterNIC, 2005). For a given domain name, the whois database can provide the registrant's name and address, domain servers, and contact information.

The American Registry for Internet Numbers (ARIN) database provides information about ownership of ranges of IP addresses (ARIN, 2005). It allows lookup of contact and registration information including IP addresses, autonomous system numbers, and registered organizations in the Americas. European IP address assignments can be discovered from Réseaux IP Euoropéens Network Coordination Centre (RIPE NCC). Likewise, Asian IP address assignments are maintained by the Asia Pacific Network Information Center (APNIC).

Another well-known and useful database is the Domain Name System (DNS). DNS is a hierarchy of servers used to associate domain names, IP addresses, and mail servers. For example, it resolves a domain name such as www.company.com to the IP address of the corresponding server. The hierarchy extends from the root DNS servers down to DNS servers for individual organizations and networks. These DNS servers contain information about other low-level DNS servers and IP addresses of individual hosts (DNSstuff, 2005).

From a digital forensic perspective, examination of an attacker's system should look for evidence of artifacts on the hard drive that show the Web sites and information gained during the footprinting process. This information is often found in the active cache or as remnants on the drive (Davis, Philipp, & Cowen, 2005).

Active Scanning

Footprinting may be viewed as similar to looking up names and numbers in a telephone book. To follow up, scanning is a more active step to learn about potential targets from their responses to various probes. There are many different ways to conduct scans, and most of them are automated for convenience and speed.

During a postmortem digital forensic examination of an attacker's host, it is important to look for tools similar to those described below. This will help an experienced examiner understand the probable skill level of the attacker. This step increases in importance when trying to understand the extent of a possible enterprise-wide compromise. Attackers generally like using the same tools over again, and in this early stage the attacker is likely to load some of these tools on other compromised hosts.

War Dialing

War dialing is an old and primitive method but still useful. Many organizations allow remote users to access an enterprise network through dial-up modems, but they can be misconfigured or overlooked by system administrators (Skoudis, 2002). War dialers are simply automated machines for dialing a set of phone lines to find accessible modems. A telephone number within an organization is usually easy to find through the Internet or telephone books, then an attacker could dial a surrounding range of numbers to discover phone lines with modems. Some war dialers include a nudging function that sends a predefined string of characters to a modem to see how it responds. The response may reveal the lack of a password, the type of platform, and perhaps a remote access program (such as the popular pcAnywhere). Many popular war dialers exist, including: *Toneloc*, *THC Scan*, *Phone Tag*, *Rasusers*, *Microsoft's Hyper-Terminal*, *PhoneSweep*, *Sandtrap*, and *Procomm Plus* (Packet Storm, 2005).

Although war dialers have been in use for decades, they can still be effective in attacks when a modem is not properly secured. Obviously, modems without password protection are completely vulnerable. Also, modems can be attacked by guessing the password. A successful attack through an unsecure modem can lead to compromise of an entire organization's network, effectively bypassing firewalls and other sophisticated defenses.

Ping Sweeps

The internet control message protocol (ICMP) is an essential part of the Internet protocol to enable notification of troubles and other control functions. ICMP includes a very useful utility called ping, typically used to verify that a specific host is operational (IETF RFC 1739, 1994). Ping messages consist of a pair of ICMP messages called Echo Request and Echo Reply. A host that receives an ICMP Echo Request message should reply with an ICMP Echo Reply.

Ping is frequently used by attackers to sweep or scan a block of IP addresses for active hosts. Many tools can easily perform a ping sweep. However, ping sweeps have two drawbacks for attackers. Ping sweeps can be noticed and alert potential targets of an imminent attack. Also, organizations will sometimes block ICMP messages as a matter of policy. To avoid this problem, TCP packets to well-known ports will also work. An initial TCP SYN packet (used to request a new TCP connection) to a target will prompt a TCP SYN-ACK reply.

Network Mapping

Ping sweeps will reveal the addresses of active hosts but no information about their networks. Traceroute is a widely used utility for mapping a network topology (Stevens, 1994). It takes advantage of the time-to-live (TTL) field in the IP packet header. When an IP packet is sent, its TTL field is set to the maximum time allowed for delivery; a limited lifetime prevents IP packets from looping endlessly in the network. Each router decrements the TTL field by the time spent by the packet in that router. Routers typically forward packets quickly and then must decrement the TTL value by the minimum unit of one. The TTL field essentially ends up serving as a hop count. If the TTL field reaches a value of zero, a router should discard the packet and send an ICMP Time Exceeded message back to the source IP address in the discarded packet.

The traceroute utility sends out a sequence of UDP packets, starting with a TTL field value of one and incrementing the value by one for each successive packet. When ICMP Time Exceeded messages are returned, they reveal the addresses of routers at incremental distances. Similarly, ICMP messages could be used instead of UDP packets.

Port Scanning

Applications using TCP and UDP are assigned port numbers conveyed in the TCP and UDP packet headers. The headers allow a range of 65,535 TCP and 65,535 UDP ports. Certain port numbers are "well known" and pre-assigned to common protocols, as listed

Table 1. Some well-known ports

Port	Description
TCP 20	FTP data
TCP 21	FTP control
TCP 23	Telnet
TCP 25	Simple Mail Transfer Protocol
TCP 53	Domain Name System
TCP 80	HTTP
UDP 161	Simple Network Management Protocol
TCP 179	Border Gateway Protocol

in Table 1 (IETF RFC 1700, 1994). For example, Web servers listen for HTTP requests on TCP port 80. The other ports may be used dynamically as needed.

An attacker is almost always interested to discover which ports are open (or services are active) on a potential target. An open port means that the target will be receptive on that port. Also, exploits are often targeted to the vulnerabilities of a specific service. However, probing every possible port manually would be very tedious. A port scanner is an automated tool for sending probes to a set of specific ports in order to see which ports are open.

The most widely used tool for port scanning is probably the open-source Nmap. Nmap is perhaps the most capable port scanner, providing options for many different types of scans which vary in degree of stealthiness and ability to pass through firewalls. Other popular tools include Foundstone's superscan, hping, and nemesis (Insecure, 2005).

Operating System Detection

An attacker may attempt to discover a target computer's operating system because specific vulnerabilities are known for different operating systems (and their different versions). Eavesdropping on network traffic with a sniffer can find clues about a host's operating system (McClure, Scambray, & Kutz, 2001). Different operating systems exhibit specific behavior in setting TTL values in IP packet headers and TCP window sizes, for example. An active technique used by attackers is TCP stack fingerprinting which can be found in the popular Nmap tool. TCP stack fingerprinting takes advantage of the fact that while the TCP protocol is standardized in terms of its three-way connection establishment handshake, the standards do not cover responses to various illegal combinations of TCP flags. Operating systems can differ in their implementations of responses to illegal TCP packets. By probing for these differences with various illegal TCP packets, the operating system and even its particular version can be identified (Fyodor, 2002). Once an operating system is identified, an attacker could attempt exploits targeted to vulnerabilities known for that operating system.

Versatile Scanning Tools

A large number of free and commercial scanning tools are available. Many of these are used for legitimate purposes by system administrators as well to learn about or verify the configurations of hosts on their enterprise networks. We list here a number of tools that appeal to attackers because they conveniently combine several of the mapping and scanning functions mentioned earlier.

Sam Spade is a combination of useful reconnaissance tools with a Windows graphical user interface (Sam Spade, 2005). Its functions include ping, whois, IP block whois (ARIN database query), nslookup, traceroute, and a utility to verify e-mail addresses on a specific mail server. A version of Sam Spade is available as a Web-based tool, as shown in Figure 3.

Figure 3. Screenshot of Sam Spade

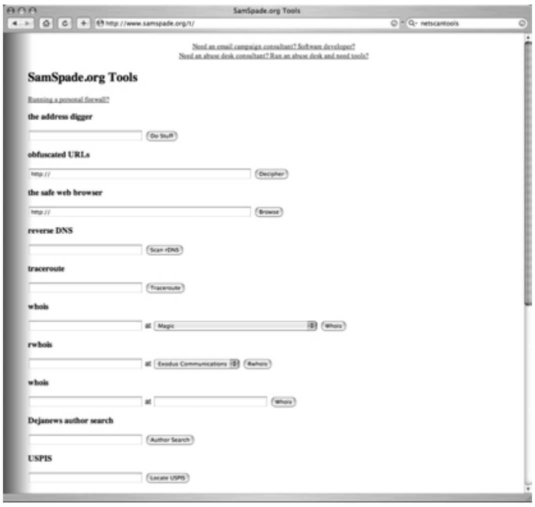
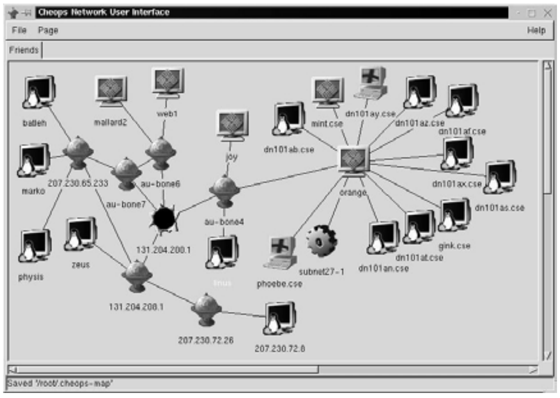


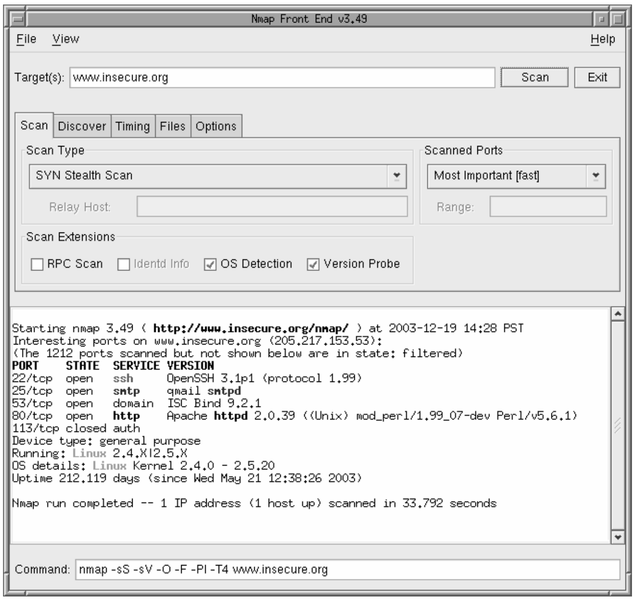
Figure 4. Screenshot of Cheops



Other examples of free scanning tools include CyberKit and Cheops (Cyberkit, 2005; Cheops, 2005). Cheops is a popular, easy-to-use utility for network mapping that can automatically draw out a network topology based on discovered hosts and distances. A screenshot of the Cheops interface is shown in Figure 4. It can also discover active services through port scanning and identifies operating systems by TCP stack fingerprinting.

Northwest Performance Software's NetScanTools Pro is an example of a commercial tool. It includes ping, port scans, traceroute, netscanner (ping sweep), custom ICMP packet generation, whois, nslookup, IP packet capturing, e-mail address validation, and operating system identification. It uses an unusual method for operating system identification based on observing responses to four types of ICMP messages and variations of

Figure 5. Screenshot of Nmap



Target(s): www.insecure.org

Scan Type: SYN Stealth Scan

Scanned Ports: Most Important [fast]

Scan Extensions: [ ] RPC Scan [ ] Identd Info [✓ OS Detection] [✓ Version Probe]

```
Starting nmap 3.49 ( http://www.insecure.org/nmap/ ) at 2003-12-19 14:28 PST
Interesting ports on www.insecure.org (205.217.153.53):
(The 1212 ports scanned but not shown below are in state: filtered)
PORT    STATE  SERVICE VERSION
22/tcp  open   ssh     OpenSSH 3.1p1 (protocol 1.99)
25/tcp  open   smtp    qmail smtpd
53/tcp  open   domain  ISC Bind 9.2.1
80/tcp  open   http    Apache httpd 2.0.39 ((Unix) mod_perl/1.99_07-dev Perl/v5.6.1)
113/tcp closed auth
Device type: general purpose
Running: Linux 2.4.X|2.5.X
OS details: Linux Kernel 2.4.0 - 2.5.20
Uptime 212.119 days (since Wed May 21 12:38:26 2003)

Nmap run completed -- 1 IP address (1 host up) scanned in 33.792 seconds
```

Command: `nmap -sS -sV -O -F -PI -T4 www.insecure.org`

them. WildPackets' iNetTools is another commercial tool providing many of the functions as other scanners.

Nmap was already mentioned earlier as a port scanner but it is more than a simple scanner. A screenshot of Nmap is shown in Figure 5. Other interesting options in Nmap include: scanning for RPC (remote procedure calls) services on a target machine; sending decoy scans with fake source addresses; sending scans with different timing options to avoid detection; and identifying a computer's operating system via TCP stack fingerprinting.

Vulnerability Scanning

Active scanning is invaluable to an attacker for learning a wide variety of information about a potential target, such as host addresses, network topology, open ports, and operating systems. The next basic step in reconnaissance is to scan for specific vulnerabilities that might be exploitable for an attack. Although one could manually scan each host for vulnerabilities, this method is not practical. Automated vulnerability scanners are readily available and often used by system administrators to evaluate the security of their internal network.

Attackers' toolkits have grown in sophistication over the years to the point that many functions are combined in the tools. For example, many tools perform active scanning and vulnerability scanning. Scanners evaluate several types of vulnerabilities, searching for one of three general system weaknesses that include faulty operating system code, faulty application code, or faulty configurations.

System Vulnerabilities

New vulnerabilities in operating systems are being discovered constantly (Koziol et al., 2004). The most critical vulnerabilities are often published by vendors along with a software patch. In practice, organizations find it hard to dedicate the time and effort needed to keep up regularly with security bulletins and patches. The time between the publication of a security vulnerability and the installation of patches leaves a window of opportunity for attackers to exploit that vulnerability. A Symantec report estimated that the average time between publication of a vulnerability and appearance of an exploit for that vulnerability is less than a week (Turner, 2005). Consequently, organizations should keep up with patching diligently.

Application Vulnerabilities

Vulnerabilities are found in applications as well as operating systems (Hoglund & McGraw, 2004). Applications introduce new risks to hardened operating systems by opening up new ports, installing new services, and otherwise spawning privileged processes that are sometimes faulty and susceptible to hijacking or buffer overflows. Commonly targeted applications include Web browsers and desktop applications such as Microsoft Word and Excel, which are capable of running downloaded code. A Web browser, for example, can be made to execute Javascript from an untrusted server that could make the client download and execute a malicious program.

Misconfiguration Errors

Network equipment requires significant technical expertise to configure properly. Incorrect configuration settings due to ignorance or accident can defeat any security offered by networking equipment. An example is a misconfigured firewall that could be too permissive in allowing incoming packets.

Additionally, many operating systems and service applications ship with default accounts and passwords (which are easy to find on the Web). These are intended to help ease the installation process, or simplify troubleshooting in case of lost passwords. Default passwords should be changed but can be overlooked or ignored. Attackers often look for the existence of default configurations because they offer an easy way to compromise a system.

Vulnerability Scanners

Most vulnerability scanners operate basically in a similar way (Skoudis, 2002). First, they try to search for active hosts within a given address range using ping or similar utility. Next, they run a basic set of scans to discover open ports and active services running on the hosts. Based on this information, they proceed to more customized probes to identify vulnerabilities. In the final step, they generate output in the form of a report. Some vulnerability scanners include a function for network mapping as well.

SATAN (Security Administrator's Tool for Analyzing Networks) was an early well-known vulnerability scanner developed in 1995. SATAN has two modern descendents, the open-source SARA (Security Auditor's Research Assistant) and the commercial SAINT (Security Administrator's Integrated Network Tool). SARA enhances SATAN's security engine and program architecture with an improved user interface and up-to-date vulnerability tests. SARA can discover information about hosts by examining various network services (ARC, 2005). It can also find potential security flaws, such as misconfigured network services, well-known system vulnerabilities, or poorly chosen policies. It can generate a report of these results or execute a rule-based program to investigate any potential security problems.

Nessus is a popular open-source vulnerability scanner (Nessus, 2005). It works in a client-server architecture, where the client and server may run on the same machine. The client consists of a tool for user configuration and a tool for recording and reporting results. The server consists of a vulnerability database, a knowledge base to keep track of the current scan, and a scanning engine. Nmap is included as the built-in port scanning tool. The vulnerability database is designed to be modular in the form of plug-ins. Each plug-in is designed to check for a specific vulnerability. Nessus contains over 500 plug-ins, and the user community continually contributes new ones. Vulnerabilities are rated and classified into categories such as finger abuses, Windows-related vulnerabilities, backdoors, CGI (common gateway interface) abuses, RPC vulnerabilities, firewall misconfigurations, remote root access, FTP, and SMTP (mail server vulnerabilities).

Commercial vulnerability scanners include TigerTools' TigerSuite Pro, McAfee's CyberCop ASaP, ISS's Internet Scanner, eEye Digital Security's Retina Network Security Scanner, and Cisco Systems' Secure Scanner.

Gaining Access

The attack phase to gain access to a target can take many different forms and serve different purposes, such as stealing confidential data, tampering with data, compromising the availability of a resource, or obtaining unauthorized access to a system. As shown previously in the taxonomy in Figure 1, attacks may be viewed in three broad categories: focused attacks directed at specific targets, large-scale attacks aimed indiscriminately at as many targets as possible, or attacks directed at the network infrastructure. The first two attack types are covered in this section. Quite often, large-scale indiscriminate attacks have the side effect of widespread disruption of networked systems, even if that is not the real intent.

The major types of attack covered here include sniffing, session hijacking, password attacks, exploits, social engineering attacks, Trojan horses, spyware and adware, viruses and worms, spam, and denial-of-service (DoS) attacks. This list is certainly not exhaustive, but intended to highlight the most common attack types seen today and most likely to be encountered in the near future. It should be noted that the taxonomy does not imply that methods are mutually exclusive; in fact, attack methods are often combined. For example, worms can simultaneously spread by social engineering and exploits, and carry out denial of service.

Sniffing

Sniffing is a passive attack that attempts to compromise the confidentiality of information. It might be considered part of reconnaissance (e.g., sniffing to learn passwords) prefacing an attack but can just as well be argued to be an attack to gain access to information. Sniffers traditionally used by network administrators for traffic monitoring and LAN troubleshooting have also been one of the most commonly used attack tools over the years. On a LAN, every host sees all of the traffic broadcast on the LAN medium, but normally ignore the packets that are addressed to other hosts. A sniffer program puts the network interface of a host into promiscuous mode to capture all packets seen on the LAN medium. Thus, the sniffer can eavesdrop on everything transmitted on the LAN including user names, passwords, DNS queries, e-mail messages, and all types of personal data.

Many free and commercial sniffers are available, including tcpdump, windump, Snort, Ethereal, Sniffit, and dsniff (Tcpdump, 2005; Snort, 2005; Ethereal, 2005; Dsniff, 2005).

Session Hijacking

Session hijacking gained national attention from Kevin Mitnick's alleged 1994 attack on Tsutomu Shimomura's computer (Shimomura & Markoff, 1996). Session hijacking is a combination of sniffing and address spoofing that enables the compromise of a user's remote login session, thus providing an attacker unauthorized access to a machine with the privileges of the legitimate user. Address spoofing is sending a packet with a fake source address. This is quite simple because the sender of an IP packet writes in the IP source address in the packet header. Address spoofing enables attackers to masquerade as another person.

If a user is currently engaged in an interactive login session (e.g., through telnet, rlogin, FTP), a session hijacking tool allows an attacker to steal the session. When most hijack victims see their login session disappear, they usually just assume that the cause is network trouble and try to login again, unaware of the hijacking attack.

Popular session hijacking tools include Juggernaut and Hunt (Hunt, 2005). The hijacking attack begins with the attacker sniffing packets of an interactive session between two hosts, carefully noting the TCP sequence numbers of all packets. To hijack the session, the attacker injects packets with a source address spoofing one of the hosts. The proper TCP sequence numbers must be used for the attack to work, because the receiving host must be convinced to accept the faked packets from the attacker.

Password Attacks

Password attacks attempt to gain access to a host or service with the privileges of a current user. Passwords continue to be very frequently used for access control despite their major weakness: if a password is guessed or stolen, an attacker could gain complete access. The most well-protected systems could be compromised by a single weak password. Understandably, many attacks are often directed at guessing or bypassing passwords.

Easy passwords to guess are the default passwords installed by many operating systems and service applications. For example, 3Com routers ship with Admin access with no password; Cisco CiscoWorks 2000 includes an admin account with password 'cisco' (Phenoelit, 2005). Extensive lists of default accounts and passwords are not hard to find by searching on the Web, and they are sometimes overlooked or ignored by system administrators.

The most powerful password attacks, called password cracking, can be performed if the attacker can obtain the password file (Shimonski, 2002). Computer systems store a list of user accounts and passwords in a password file, but the information is encrypted or hashed for protection against attackers. If an attacker can obtain the password file, the attacker has the advantage of time (translating into more CPU cycles) to crack the passwords by brute force (i.e., attempting all possible combinations of characters).

Brute-force password guessing can be very time consuming but is often not necessary. The natural human instinct is to choose passwords based on common words or names. A dictionary attack takes advantage of this tendency by guessing a set of common words and names. However, modern computer systems are usually programmed with policies to prevent users from choosing easily guessable passwords. Hence, the chance of guessing simple passwords is not as likely today as in the past.

More sophisticated hybrid password guessing tools combine dictionary attacks with limited brute-force attacks. They begin with guesses of common words but then methodically add characters to words to form new guesses. A few examples of password cracking tools include John the Ripper, Cain and Abel, Crack, Lincrack, L0phtcrack, Nutcracker, PalmCrack, and RainbowCrack (Password Crackers, 2005).

Exploits

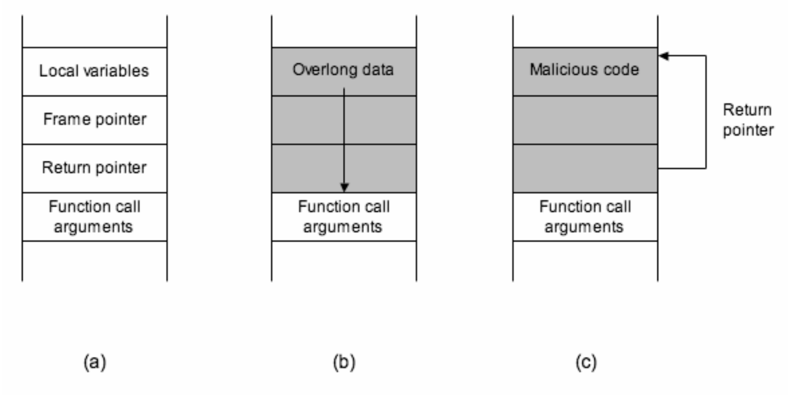
As mentioned earlier, new vulnerabilities in operating systems and application software are being discovered constantly. A vulnerability is a description of a security hole which is not dangerous per se. However, given knowledge of a vulnerability and sufficient time, attackers will write an exploit to take advantage of that vulnerability (Hoglund & McGraw, 2004). The danger arises when the exploit appears and is shared among attackers. Vulnerabilities are associated with different levels of seriousness, where the most critical vulnerabilities can potentially lead to exploits that completely compromise a target host.

A vendor usually has knowledge of a vulnerability but withholds the information from the public at large until there is a fix for the problem. Then vulnerabilities are announced at the same time as a patch for fixing the vulnerability. Unfortunately, patches take time to download and apply, particularly for large organizations with many computers. For practical reasons, organizations and individuals can have a hard time keeping up to date with patches. If an organization is slow to patch however, it can be exposed to new exploits.

SANS maintains a "top 20" list of the most critical Internet security vulnerabilities (SANS, 2005). A buffer overflow vulnerability is one of the most commonly sought by attackers to exploit. Buffer overflow attacks are used particularly often by worms. This type of exploit is appealing to attackers because many applications and operating systems do not perform proper bounds checking and are thus vulnerable to a buffer overflow. Moreover, a successful buffer overflow attack could lead to complete control of a target host.

A well-known example is a stack-based buffer overflow attack, popularly known as "smashing the stack" (AlephOne, 1996). During a function call, various pieces of data are pushed onto the program stack: function-call arguments, return pointer, frame pointer, and local variables. This is illustrated in Figure 6(a). Normally, at the end of the function call, the pieces of data are popped off the stack, and the return pointer is used to resume execution of the main program. A stack-based buffer overflow depends on inputting more data than expected into the local variables. The excess data is written into the allocated buffer space and then overwritten onto the frame pointer and return pointer,

Figure 6. Buffer overflow attack.



(a) Data pushed onto stack in normal function call. (b) Data overflows allocated space and overwrites return pointer in buffer overflow attack. (c) Return pointer now points back into stack, causing the malicious code to execute.

as shown in Figure 6(b). If the excess data can be crafted carefully enough, the overwritten return pointer can be made to point back into the stack somewhere in the data input by the attacker, as shown in Figure 6(c). Hence, when the main program resumes execution, the attacker's data (malicious code) will be run.

Obviously, a buffer overflow attack requires careful coding and significant technical knowledge about the target processor architecture. Hence, buffer overflow attacks are not easy to write from scratch. However, pre-written exploits are often shared among attackers and can be used without requiring a great deal of technical knowledge.

Social Engineering

Social engineering attacks take advantage of human interaction; social skills are used to trick the victim into a compromising action, such as revealing personal information or opening an infected e-mail message. Social engineering can be combined with many of the other attack methods to compromise security for just about any purpose. Although social engineering attacks are simple and low tech, they can be surprisingly effective if executed well.

In the past, the telephone was a favorite avenue for social engineering attacks. Today, many social engineering attacks are carried out through e-mail, due to the low risk and low cost of mass e-mailing. Also, e-mail works across different computing platforms and various types of devices. E-mail became the preferred medium after the success demonstrated by mass e-mailing viruses, such as the 2000 Love Letter and 2001 Anna Kournikova viruses. E-mail viruses typically offer a provocative reason to entice the recipient into opening an e-mail attachment, which results in a virus infection. More recently, e-mails might pretend to be security bulletins, bounced e-mail, notifications from an ISP or system administrator, or other official-looking messages.

Recently, a type of social engineering attack called phishing has escalated in frequency. Phishing attacks begin with e-mail seemingly from a reputable credit card company or financial institution that requests account information, often suggesting that there is a problem with an account or a transaction. These e-mails are carefully crafted to appear official and often include stolen corporate graphics. The e-mails typically include a link directing the victim to a Web site that appears to be genuine, but is actually fake. The purpose of the fake Web site is to capture any account or personal information submitted by the victim or download malicious code to the victim host. The Anti-Phishing Working Group counted 3,326 active phishing Web sites in May 2005, compared to 1,518 sites in November 2004 (Anti-Phishing Working Group, 2005).

Trojan Horses

Trojan horses are defined as malicious software that appear to be benign (analogous to the Greek wooden horse in the Trojan War) (Grimes, 2001). The purpose of the disguise is to entice a user into installing and executing the program. If executed, Trojan horses are capable of doing anything that other programs can do, running with the privileges of the associated user. Trojan horses can be combined with many of the other attack types (such as social engineering) to compromise security for just about any purpose.

In common usage, the term Trojan horses include some types of stealthy malicious code which attempt to hide their existence on a victim host. These Trojan horses are distributed by any number of stealthy ways including virus and worm payloads, peer-to-peer file sharing, and Web site downloads. Victims are often unaware of their installation.

The most worrisome Trojan horse may be backdoor programs, sometimes called remote access Trojans (RATs) because backdoors allow an attacker to remotely access a victim's machine (Grimes, 2002). Backdoors circumvent the usual access control security (e.g., login with password). Many backdoor Trojans are known and some are promoted for legitimate administrative uses, including Sub7, Back Orifice 2000, and VNC (Sub7, 2005; BO2K, 2005; RealVNC, 2005).

Adware and Spyware

Adware is software to monitor and profile a user's online behavior, typically for the purposes of targeted marketing. Adware is often installed at the same time as other software programs without the user's knowledge. Even when the user is alerted to the presence of the adware (often buried in the ignored licensing agreement), adware can be an attack on the privacy of the user when information about the user is communicated back to a marketing organization. Adware is primarily an annoyance, sometimes causing pop-up marketing windows during Web surfing.

A more serious and growing concern is another type of software that profiles and records a user's activities, called spyware. A Webroot report estimated that 88 percent of PCs were infected by spyware and 89,806 Web pages contained spyware for possible download during the first quarter of 2005 (Webroot, 2005). Similar to adware, spyware can sometimes be installed with a user's or system administrator's knowledge. For example, commercial versions of spyware are sold as means to monitor and regulate the online actions of children or an organization's employees. Often though, spyware can be installed stealthily on a machine as a Trojan horse or as part of a virus or worm infection. Spyware can record keystrokes (also known as keystroke loggers), Websites visited, passwords, screenshots, and virtually anything done on a computer. After capturing data, spyware can communicate the stolen data by various channels (e.g., e-mail, FTP, upload to the Web, or Internet Relay Chat) to an attacker. Spyware, like adware, is an attack on user privacy, but spyware is also more likely to compromise confidential data for identity theft.

Viruses and Worms

Viruses and worms are software designed for self-replication (Grimes, 2001; Harley, Slade, & Gattiker, 2001). While there is a certain disagreement among definitions, viruses are commonly considered to be snippets of program code that replicate by modifying (infecting) a normal program or file with a copy of itself. They are not complete (stand-alone) programs themselves but depend on execution of the infected program. When the host program or file is executed, the virus code is executed and takes over control to copy itself to other files. Usually human action is needed to execute the host program, so viruses are sometimes said to require human action to replicate (Webopedia, 2005).

In contrast, worms are stand-alone programs that replicate by spreading copies of themselves to other systems through a network. Worms have become more predominant than viruses in the past few years due to the growth of computer networks. Today, virtually all computers are connected to private networks or the Internet, which is an environment naturally friendly to worms. In particular, the widespread popularity of e-mail has made it easier for worms to spread across different computing platforms. E-mail continues to be the most popular vector for worm propagation.

Viruses have evolved in their complexity over the years, often in response to counter-measures put in place by anti-virus vendors. The first viruses often simply added their code to either the beginning or the end of the host file. In order to evade simple detection, viruses later began to intersperse their code throughout the host file. Another technique that viruses have adopted to evade detection is to encrypt their code within each host file instance, thus making it more difficult for a signature of the virus to be developed. When anti-virus programs began keying on the decryption algorithm as the signature, viruses became polymorphic, changing their decryption algorithm with each copy (Nachenberg, 1996). Taking it one step further, some viruses have become metamorphic, in other words, they change their logic (not just the decryption algorithm) with each infection instance (Szor, 2005).

Network-enabled worms have not had to evolve in the same way as file-infecting viruses. Functionally, a worm program must carry out a few specific steps to spread to another target after infection of a victim host.

First, an algorithm chooses candidates for the next targets. The simplest algorithm, which is used by quite a few worms, is to choose an IP address (32-bit number) at random. This is not efficient because the IP address space is not populated uniformly. More sophisticated target selection algorithms choose addresses within the same networks as the victim because local networks have shorter propagation delays to allow faster spreading. Other target selection algorithms may choose targets discovered from a victim's e-mail address book, mail server, DNS server, or countless other ways.

Second, some worms will perform scanning of selected targets. Scanning prompts responses from the potential targets that indicate whether the worm's programmed exploits can be successful. This process identifies suitable targets among the selected candidates.

The third step is the actual exploit or attack to compromise a suitable target. A common attack is to send e-mail to the target, usually carrying an infected attachment that has to be executed. More sophisticated e-mail worms are activated when their message is just previewed or read. Other worms might attack via file sharing, password guessing, or any number of exploits. It is also common for worms to combine multiple exploits to increase the likelihood of success and rate of spreading.

The fourth step after successfully gaining access is to transfer a copy of the worm to the target. Depending on the exploit, a copy of the worm might have been transferred during the exploit (e.g., by e-mail). However, some exploits only create a means of access, such as a backdoor or shell. The worm takes advantage of the access to transfer a copy of itself via any number of protocols including FTP, TFTP, or HTTP.

An optional last step is execution of the worm's payload, if there is one. The payload is the part of the worm's program that is directed at an infected victim and not related to its propagation. The payload could be virtually anything, and not necessarily destructive. In recent cases, payloads have included: opening backdoors and thus allowing remote access, installing spyware, downloading worm code updates from the Internet, or disabling anti-virus software.

Spam

Spam, the e-mail equivalent of unsolicited junk mail, has been a growing problem over the past few years. The volume of spam has been estimated as 60 percent of all e-mail traffic during the second half of 2004 (Turner, 2005). E-mail addresses are harvested from the Internet or generated randomly. They typically advertise a product, service, or investment scheme (which may well turn out to be fraudulent). E-mail is appealing because spammers can send enormous volumes of e-mail at much lower cost than postal mail. The necessary equipment is modest: a PC, software, and an Internet connection. Even if the response rate is very small, a sizable profit can be made easily.

At the very least, spam wastes network resources (bandwidth, memory, server processing) and necessitates spam filtering at ISPs and organizations. It also wastes the valuable time of users and system administrators. The seriousness of the problem has steadily grown as the volume of spam has escalated.

A growing concern with spam is evidence of collaboration between spammers, virus/worm writers, and organized crime. A substantial number of worms have been used as a delivery vehicle for Trojan horses that set up "bot networks." Bots are stealthy programs that listen for instructions from a remote attacker or allow backdoor access. A bot net is formed by a number of bots under coordinated control. Bot nets as large as 50,000 hosts have been observed (Honeynet Project, 2005). Bot nets are being used for distributed DoS attacks or spamming. Moreover, spam is increasingly being used for phishing (as described earlier). Phishing attacks attempting identity theft with increasing sophistication suggests the involvement of organized crime.

Denial of Service

Most people tend to think of denial of service (DoS) attacks as flooding, but at least four types of DoS attacks can be identified:

- starvation of resources (e.g., CPU cycles, memory) on a particular machine;
- causing failure of applications or operating systems to handle exceptional conditions, due to programming flaws;

- attacks on routing and DNS;
- blocking of network access by consuming bandwidth with flooding traffic.

There are numerous examples of DoS attacks. A "land attack" is an example of starvation. On vulnerable machines with Windows NT before service pack 4, the land attack would cause the machine to loop, endlessly consuming CPU cycles. The "ping of death" is an ICMP Echo Request message exceeding the maximum allowable length of 65,536 bytes. It caused earlier operating systems to crash or freeze (that programming flaw has been remedied in later operating systems).

The "Smurf" attack is an example of an indirect flooding attack, where the ICMP protocol is abused to cause many response packets to be sent to a victim machine in response to a broadcast packet. It is indirect because the real attacker's address is not seen in any packets. It is also interesting as an example of amplification: a single attacker's packet is multiplied into many packets by the recipients of the broadcast.

The most harmful flooding attacks take advantage of amplification through a distributed DoS network (Dittrich, 2004). A famous distributed DoS attack occurred in February 2000 taking down several Websites including Yahoo, eBay, e*Trade, and others for 1-3 hours (Harrison, 2000). Examples of automated distributed DoS tools include Trin00, TFN (tribe flood network), TFN2K, and Stacheldraht. In addition, viruses and worms have been known to infect victims with DoS agents.

Distributed DoS attacks generally proceed in two phases. The first phase is stealthy preparation of the DDoS network. The attacker attempts to compromise a large number of computers, often home PCs with a broadband connection, by installing a DoS agent (i.e., a Trojan horse). Distributed DoS tools such as Trin00 and TFN set up a two-level network. A small fraction of compromised machines are designated as "masters," waiting for commands from the attacker. The remainder of compromised machines are "daemons" waiting for commands from masters. The daemons carry out the actual flooding attack to a specified target.

Covering Up

Cover-up is the last basic step in an attack. During reconnaissance or an attack, an attacker would naturally prefer to avoid detection, which could trigger defensive actions. The problem is evasion of intrusion detection systems (IDSs) which are designed to catch attacks.

After a successful attack gaining access or control of a target, an attacker would like to hide evidence of the attack for the same reasons. Detection of a compromise would lead to defensive actions to defeat the attack, trace the attack back to the attacker, and increase defenses against future attacks.

Evading Intrusion Detection Systems

IDSs are designed to alert system administrators about any signs of suspicious activities. They are analogous in concept to burglar alarms, designed to react against intruders who are able to penetrate preventive defenses (e.g., firewalls). Network-based IDSs monitor the network traffic and might be implemented in a stand-alone device or integrated in firewalls or routers. Host-based IDSs are processes that run on hosts and monitor system activities. IDSs are now commonly used by organizations. Naturally, an intelligent attacker would want to avoid detection by IDSs.

Without special precautions, an attacker could be easily detected by an IDS during reconnaissance because scanning tools are noisy. A port scan might involve thousands of packets, while a vulnerability scan could involve hundreds of thousands of packets. These scans would have a noticeable impact on normal traffic patterns in a network. Moreover, these scans are exactly the signs that IDSs are designed to look for.

Most commercial IDSs attempt to match observed traffic against a database of attack signatures. This approach is called misuse or signature-based detection. Hence, an attacker could try to evade a signature match by changing the packets or traffic pattern of an attack. One approach to changing the appearance of an attack is to take advantage of IP fragmentation. An IDS must be able to reassemble fragments in order to detect an attack. An IDS without the capability for fragment reassembly could be evaded by simply fragmenting the attack packets. An IDS might also be overwhelmed by a flood of fragments or unusual fragmentation.

IDS evasion is also possible at the application layer. For example, an IDS may have a signature for attacks against known weak CGI scripts on a Web server. An attacker could try to evade this signature by sending an HTTP request for a CGI script, but the HTTP request is carefully modified to not match the signature but still run on the Web server.

Another strategy for evading detection by IDSs is to simply overload them with common, unimportant events to mask the actual attack. “Flying under the radar” of an IDS is somewhat easy to do when thousands of meaningless port scans and ping sweeps are filling the operators’ consoles and logs, while a more sophisticated attack is executed.

Modifying Logs

Covering up evidence after an attack is particularly important if an attacker wants to maintain control of the victims. One of the obvious necessities is to change the system logs on the victim computers. Unix machines keep a running system log about all system activities, which can be viewed by system administrators to detect signs of intrusions. Likewise, Windows NT/2000/XP systems maintain event logs including logins, file changes, communications, and so on.

An attacker needs to gain sufficient access privileges, such as root or administrator, to change the log files. It is unwise for attackers to simply delete the logs because their absence would be noticed by system administrators searching for unusual signs. Instead, a sophisticated attacker will try to carefully edit system logs to selectively remove suspicious events, such as failed login attempts, error conditions, and file accesses.

Rootkits

Rootkits are known to be one of the most dangerous means for attackers to cover their tracks (Hoglund & Butler, 2005). Rootkits are obviously named for the root account which is the most prized target on Unix systems because the root user has complete system access. If an attacker has gained root access, it is possible to install a rootkit designed to hide signs of a compromise by selectively changing key system components. The rootkit cannot be detected as an additional application or process: it is a change to the operating system itself. For example, Unix systems include a program ifconfig that can show the status of network interfaces, including interfaces in promiscuous mode (or a sniffer). A rootkit could modify ifconfig to never reveal promiscuous interfaces, effectively hiding the presence of a sniffer. Another program find is normally useful to locate files and directories. A rootkit could modify find to hide an attacker's files.

Kernel-level rootkits have evolved from traditional rootkits (Wichmann, 2002). In most operating systems, the kernel is the fundamental core that controls processes, system memory, disk access, and other essential system operations. As the term implies, kernel-level rootkits involve modification of the kernel itself. The deception is embedded at the deepest level of the system, such that no programs or utilities can be trusted any more. Kernel-level rootkits might well be impossible to discover.

Covert Channels

Although logs and operating systems can be modified to escape detection, the presence of a system compromise might be given away by communications. For example, system administrators might recognize the packets from an attacker trying to access a backdoor through a particular port. Clearly, an attacker would prefer to hide his communications through covert channels.

Tunneling is a common method used to hide communications. Tunneling simply means one packet encapsulated in the payload of another packet. The outer packet is the vehicle for delivery through a network; the receiver has to simply extract the inner packet which is carried through the network unchanged. The outer packet is usually IP for routing through the Internet. Also, ICMP messages and HTTP messages have been used. Since the inner packet has no effect on network routing, any type of packet can be carried by tunneling.

Conclusions and Future Trends

Computer systems are common targets for a wide range of electronic attacks. Instead of an exhaustive catalog, this chapter has attempted a quick tour of the most pressing types of attacks in preparation for later chapters with more details.

An understanding of attacks is a necessary prerequisite to designing proper digital forensic methods to collect and analyze evidence of attacks. Clearly, analysis of evidence to look for an attack can not be done properly without knowing the attack behavior. We have seen that attacks can be viewed as a sequence of phases proceeding from reconnaissance to access to coverup. Each step could leave digital evidence for crime investigators. Signs of reconnaissance could include existence of tools for scanning and network mapping. Attack tools such as session hijacking tools or sniffers would be obvious implications of crime. Evidence of coverup could include changed system logs or signs of a rootkit.

Predictions about the future of cyber attacks are difficult due to the unpredictability of cyber criminals. The perpetual struggle between cyber criminals and law enforcement means that both sides continually attempt to adapt. One side continually invents new types of attacks and attack tools, while the other side has historically followed. Extrapolating current trends, we might predict:

- attacks will increase in sophistication and coordination, out of necessity to evade more sophisticate law enforcement;
- attacks designed for profit and identity theft will increase;
- social engineering attacks will continue through e-mail, given its current success;
- spam volume will continue to increase, unless measures are taken to change the profitability for spammers;
- malicious code (viruses, worms, Trojan horses) has been the single most prevalent attack found in the CSI/FBI surveys over the last five years and will continue to be the most prevalent attack;
- malicious code will increase in new vectors such as instant messaging and mobile handheld devices (such as cell phones) ;
- attackers will seek to construct more and bigger bot nets.

Increasing sophistication of attacks implies that digital forensics will have proportionately greater importance in investigating, diagnosing, and analyzing cyber crimes. Digital forensic techniques will be challenged by attackers who will have access to more and better attack tools. These attackers will be capable of effective remote exploits and evasion of detection. Cyber crime investigators will need better knowledge of attacks and better forensic tools for collecting and analyzing electronic evidence.

References

Aleph One, Smashing the stack for fun and profit. Retrieved April 30, 2005 from http://www.insecure.org/stf/smashstack.txt

Anti-Phishing Working Group homepage. Retrieved July 30, 2005 from http://www.antiphishing.org.

ARC, Security Auditor's Research Assistant. Retrieved July 30, 2005 from http://www-arc.com/sara/.

ARIN, Whois database search. Retrieved July 30, 2005 from http://www.arin.net/whois/

BO2K homepage. Retrieved July 30, 2005 from http://www.bo2k.com

Chakrabarti, A. & Manimaran, G. (2002). Internet infrastructure security: a taxonomy. *IEEE Network*, *16*, 13-21.

Cheops homepage. Retrieved July 30, 2005 from http://www.marko.net/cheops/

Chirillo, J. (2002). *Hack attacks revealed* (2nd ed.) Indianapolis, IA: Wiley Publishing.

Cyberkit homepage. Retrieved July 30, 2005 from http://www.gknw.com/mirror/cyberkit/

Davis, C., Philipp, A., & Cowen, D. (2005). *Hacking exposed: Computer forensics secrets and solutions*. New York: McGraw-Hill/Osborne.

Dittrich, D. (2005). *Distributed denial of service (DDoS) attacks/tools*. Retrieved April 30, 2005 from http://staff.washington.edu/dittrich/misc/ddos/

DNSstuff homepage. Retrieved July 30, 2005 from http://www.dnsstuff.com

Dsniff homepage. Retrieved July 30, 2005 from http://www.monkey.org/~dugsong/dsniff/

Ethereal homepage. Retrieved July 30, 2005 from http://www.ethereal.com

Fyodor, Remote OS detection via TCP/IP stack fingerprinting. Retrieved April 30, 2005 from http://www.insecure.org/nmap/nmap-fingerprinting-article.html

Gordon, L., et al. (2005). *2005 CSI/FBI computer crime and security survey*. Retrieved July 25, 2005 from http://www.gocsi.com

Grimes, R. (2001). *Malicious mobile code: Virus protection for Windows*. Sebastopol, CA: O'Reilly.

Grimes, R. (2002). *Danger: remote access trojans*. Retrieved July 30, 2005 from http://www.microsoft.com/technet/security/alerts/info/virusrat.mspx

Harley, D., Slade, D., & Gattiker, U. (2001). *Viruses revealed*. New York: McGraw-Hill.

Harrison, A. (2000). *Cyberassaults hit Buy.com, eBay, CNN and Amazon*. Retrieved on July 30, 2005 from http://www.computerworld.com/news/2000/story/0,11280,43010,00.html

Hoglund, G., & Butler, J. (2005). *Rootkits: Subverting the Windows kernel*. Reading, MA: Addison Wesley Professional.

Hoglund, G., & McGraw, G. (2004). *Exploiting software: How to break code*. Boston, MA: Pearson Education.

Honeynet Project. (2005). *Know your enemy: Tracking botnets*. Retrieved July 30, 2005 from http://www.honeynet.org/papers/bots/

Hunt homepage. Retrieved July 30, 2005 from http://lin.fsid.cvut.cz/~kra/index.html

IETF RFC 1700. (1994). *Assigned numbers*. Retrieved July 30, 2005 from http://www.ietf.org/rfc/rfc1700.txt

IETF RFC 1739. (1994). *A primer on Internet and TCP/IP tools*. Retrieved July 30, 2005 from http://www.ietf.org/rfc/rfc1739.txt

Insecure, Nmap free security scanner, tools & hacking resources. Retrieved July 30, 2005 from http://www.insecure.org

InterNIC, The Internet's network information center. Retrieved July 30, 2005 from http://www.internic.net

Kloth.net, Online services. Retrieved July 30, 2005 from http://www.kloth.net/services/

Koziol, J., et al. (2004). *The shellcoder's handbook: Discovering and exploiting security holes*. Indianapolis, IA: Wiley Publishing.

Markoff, J., and Shimomura, T. (1996). *Takedown: The pursuit and capture of Kevin Mitnick, America's most wanted computer outlaw – By the man who did it*. New York: Hyperion Books.

McClure, S., Kutz, G., and Scambray, J. (2001). *Hacking exposed* (3rd ed.) New York: McGraw-Hill.

Nachenberg, C. (1996). *Understanding and managing polymorphic viruses*. Retrieved July 30, 2005 from http://www.symantec.com/avcenter/reference/striker.pdf

Nessus homepage. Retrieved July 30, 2005 from http://www.nessus.org

Packet Storm, Wardialers. Retrieved July 30, 2005 from http://packetstorm.linuxsecurity.com/wardialers/

Password Crackers, Russian password crackers. Retrieved July 30, 2005 from http://www.password-crackers.com/crack.html

Phenoelit, Default password list. Retrieved July 30, 2005 from http://www.phenoelit.de/dpl/dpl.html

RealVNC homepage. Retrieved July 30, 2005 from http://www.realvnc.com

Sam Spade homepage. Retrieved July 30, 2005 from http://www.samspade.org

SANS, The twenty most critical Internet security vulnerabilities (updated) – The experts consensus. Retrieved July 30, 2005 from http://www.sans.org/top20/

Shimonski, R. (2005). *Introduction to password cracking*. Retrieved April 30, 2005 from http://www-106.ibm.com/developerworks/library/s-crack/

Shinder, D. & Tittel, E. (2002). *Scene of the cybercrime: Computer forensics handbook*. Rockland, MA: Syngress Publishing.

Skoudis, E. (2002). *Counter hack: A step-by-step guide to computer attacks and effective defenses*. Upper Saddle River, NJ: Prentice Hall PTR.

Snort homepage. Retrieved July 30, 2005 from http://www.snort.org

Stevens, W. R. (1994). *TCP/IP illustrated, volume 1: The protocols*. Reading, MA: Addison-Wesley Publishing.

Sub7 homepage. Retrieved July 30, 2005 from http://sub7.net

Swartz, J. (2004). Crooks slither into Net's shady nooks and crannies, *USA Today*. Retrieved July 30, 2005 from http://www.usatoday.com/tech/news/2004-10-20-cyber-crime_x.htm

Szor, P. (2005). *The art of computer virus and defense*. Reading, MA: Addison Wesley Professional.

Tcpdump homepage. Retrieved July 30, 2005 from http://www.tcpdump.org

Turner, D., et al. (2005). *Symantec internet security threat report: Trends for July 2004 - December 2004*. Retrieved July 30, 2005 from http://www.symantec.com

Webopedia, The difference between a virus, worm and Trojan horse? Retrieved July 30, 2005 from http://www.webopedia.com/DidYouKnow/Internet/2004/virus.asp

Webroot. (2005). *State of sypware Q1 2005*. Retrieved July 30, 2005 from http://www.webroot.com

Wichmann, R. (2002). *Linux kernel rootkits*. Retrieved on July 30, 2005 from http://la-samhna.de/library/rootkits/

Appendix: Acronyms

APNIC	Asia Pacific Network Information Center
ARIN	American Registry for Internet Numbers
CGI	Common Gateway Interface
DNS	Domain Name System
DoS	Denial of Service
FTP	File Transfer Protocol
HTTP	Hypertext Transfer Protocol
ICMP	Internet Control Message Protocol
IDS	Intrusion Detection System
InterNIC	Internet Network Information Center
IP	Internet Protocol
ISP	Internet Service Provider
LAN	Local Area Network
RAT	Remote Access Trojan
RIPE NCC	Réseaux IP Euoropéens Network Coordination Centre
SAINT	Security Administrator's Integrated Network Tool

SARA	Security Auditor's Research Assistant
SATAN	Security Administrator's Tool for Analyzing Networks
TCP	Transmission Control Protocol
TFN	Tribe Flood Network
TTL	Time to Live
UDP	User Datagram Protocol