Global Information Assurance Certification Paper

A New Evolution in Hack Attacks:

A General Overview of Types, Methods, Tools, and Prevention

Kelley Ealy
GSEC Practical v.1.4b
Option 1

Table of Contents

Abstract

Network administrators spend a considerable amount of time and energy working on ways to prevent “hack attacks,” or unauthorized access to corporate networks and Web sites. Although a lot of focus and study has turned to preventing internal security threats, administrators should not forget the external threat from the Internet, either. In spite of the fact that administrators are becoming much more diligent about hardening perimeter defenses and implementing Intrusion Detection Systems and Intrusion Prevention Systems, hacker methods and tools are becoming much more sophisticated and much harder to detect. In order to provide the best protection possible, network administrators must stay informed about the ongoing evolution of hack attacks.

The purpose of this paper is to provide a general overview of the more current hacking methods that go above and beyond the traditional methods of the recent past. This paper provides a straightforward review of the concepts associated with hack attacks, types of tools used to perform these attacks, and methods of prevention.

Methods of Hacker Attacks

The Internet provides an ideal setting for the development of a newer breed of hacker tools. The evolution of hacker tools has moved towards more powerful architectures that have complex backbones, yet are very easy to use. “The Cert Coordination Center (CCC) has been observing intruder activity since 1988.”[1] The CCC has noted several trends in attacks against organizations in recent years.

First, thanks to automation, hacker tools are much faster now than in previous years. In addition, the sophistication of these tools is forever increasing, thanks to advanced design techniques. These tools have three basic characteristics that increase their sophistication:

- anti-forensics
- dynamic behavior, and
- modularity.

Anti-forensics refers to the ability of the hacker tools to conceal their true identities. Dynamic behavior allows attackers to vary their methods and patterns for attacking a victim. Finally, attack tools are composed of many different modules or utilities, support several different operating system platforms, and allow hackers to launch more attacks from one tool.

New vulnerabilities are being discovered at an alarming rate. The Cert Coordination Center reports that vulnerability discoveries have doubled in the last

year. This increase poses a problem for administrators, because, in part, it is very difficult to keep up with patches on systems that may result in opening up systems to hack attacks. Another trend is the increasing permeability of firewalls. Development tools such as, Active X and Java, and protocols such as Internet Printing Protocol, allow hackers to open up ports that are traditionally marked for attack by intruders. Finally, infrastructure attacks continue to grow as the sophistication of attacks mature, resulting in malicious tools such as, worms, distributed denial of service attacks, and domain hijacking.[1]

Know Thy Enemy

Newer methods in use by attackers include, but are not limited to, distributed denial of service, distributed password cracking, distributed port scanning, active sniffing, and rootkits. Administrators must take time to learn the mechanisms behind these attacks so that they, hopefully, will recognize some attacks as they occur on their systems. Administrators must fully understand hack tools so they can proactively protect their perimeters and internal systems against attacks. Let's take a look at some of the most common types.

Distributed Denial of Service Attack (DDoS)

In 2000, the distributed denial of service (DDoS) attack made itself widely known to the public by shutting down some of the largest Web sites on the Internet, including Yahoo, Ebay, and CNN. In a nutshell, according to Tanase, a distributed attack occurs when a hacker creates a "zombie" network by installing remote control client software on open systems on the network, and uses the remote control software to flood a single target with unwanted data traffic. Eventually the target is forced to shut down and becomes inaccessible by legitimate users.

How does a DDoS attack work? Here is a simple outline of what happens:

1. The hacker or client locates a large number of vulnerable systems on the Internet.
2. The client installs the DDoS remote control program on the systems, creating "slave" systems.
3. The hacker controls the remote control programs from a "master" device directing the "slaves" to attack on a target. During the attack, large packets of data are sent by the slaves to the target system.
4. Due to the large streams of packets, the target is flooded with communications, and the system either shuts down, or legitimate users are denied access. [2]

Types of DDoS

There are several common Distributed Denial of Service attacks. The most common types include buffer overflow attacks, SYN flood attacks, teardrop attacks, so-called "Smurf" attacks, and viruses or worms. Here is a brief description of each type:

- **Buffer Overflow Attacks**. Buffer overflows take advantage of poorly written code, like a program that does not check the size of data being inserted into a buffer. Attackers cause a buffer overflow by changing the value of a program variable to a number greater than expected and executing arbitrary code under a privileged user account. [3, p.160:4]

- **SYN Flood Attacks**. In normal data exchange, a SYN packet is sent from computer A to computer B. In return, computer B will send a SYN/ACK packet to computer A. Then, computer A will send an ACK packet to computer B, establishing a connection. In a SYN flood attack, an intruder will send a SYN packet from computer A to computer B, but the intruder spoofs the source address of a non-existent system. Spoofing means gaining unauthorized access to a machine by pretending to be someone from a trusted site. Computer B will attempt to send a SYN/ACK to a non-existent system, causing a back-logged queue of connection attempts from computer B to computer A. The intruder can eventually disable a port or service just by sending a few SYN packets [3, p. 492:4]

- **Teardrop Attacks**. Large packets of data often need to be broken into smaller fragments as they are transmitted over the network, depending upon the network's maximum transmission unit. Many older kernels checked for fragments that were too large, but did not check for and reject fragments that were too small. Intruders took advantage of this vulnerability and would construct packets that were smaller than acceptable, causing systems to reboot or halt. [3, p.495:4]

- **Smurf Attacks**. In a so-called "smurf attack," an amplifying or intermediary network's broadcast address receives forged ICMP ECHO packets from the attacker. The packets appear as though the victim has initiated the request, causing all systems on the amplifying network to send a response to the victim. The magnitude of the attack, or amplification ratio, is measured by the number of systems on the amplifying network to the victim. [3, p. 489:4]

- **Viruses/Worms**. Viruses are programs or "malware" code snippets that infect systems and can be harmless or destructive. Self-replicating viruses are worms that consume resources. [11]

Types of DDoS Programs

There are several distributed denial of service tools available for easy download over the Internet. Each tool focuses on the same goal, to inundate their victim with an overwhelming amount of traffic so the victim cannot detect or filter traffic. Unfortunately for network administrators, each tool offers its own complexities and capabilities. The following is a brief description of some of the most common DDoS tools.

- **TFN**. Tribal (or "Teletubby" as it as sometimes called) Flood Network is one of the first DDoS tools to arrive on the block. It is a two-tier based architecture that carries out an attack by the client or master program sending attack instructions (using ICMP echo reply packets) to the TFN servers or daemons. The daemons then attack the target IP addresses that have supplied to the master by the hacker. All client and daemon source is hidden on all communications and attacks. [5]

- **Trin00**. Trin00 is a three-tier based architecture that makes it much more difficult for the attacker to be traced. The intruder contacts the master, which then sends instructions to the daemons to launch attacks via UDP packets sent to the target IP addresses. However, because Trin00 uses its own proprietary channels for communications, it fails to completely hide the source of its attack traffic.[5]

- **TFN2K**. TFN2K did not evolve into a three-tier based architecture, but unlike TFN, it added encryption to its communication between client and daemon, making it even harder to detect the source. TFN2K transports traffic via TCP, UDP and ICMP protocol, sends "decoy" packets for confusion to other nodes, and includes attacks designed to crash systems by sending malformed or invalid data packets. TFN2K is designed to attack Unix-based systems and Windows. [5]

- **Stacheldraht**. Stacheldraht combined the technology of Trin00 and TFN, creating a robust three-tier architecture that hides the source traffic. Stacheldraht offers one feature that it offers that none of the other tools listed do. Specifically, Stacheldraht has the ability to update its daemons from a network server defined in the update command. [6]

DDoS Prevention Methods

There is no single definitive method for preventing DDoS attacks. Securing host machines, of course, is a good starting point for anyone. It is important to

perform virus scans on a regular basis, keep patches up-to-date, close open, unneeded services, and implement basic firewall filtering.

One of the biggest problems with DDoS attacks is spoofed IP addresses. Egress filtering should be setup on routers to resolve this issue. With egress filtering, packets going out to the Internet are inspected before they are forwarded to the outside world from the routers. Because these routers should know every address behind the firewall, they should be able to identify spoofed addresses, and drop these spoofed packets before they reach the outside router. [7]

Another technique for defending against DDoS is to configure routers not to allow broadcast messages into the network, and for their hosts not to respond to broadcast messages. In addition, all public-accessed boxes should exist on a separate network, commonly called a demilitarized zone, and should not be able to access the internal network. Network administrators should also install an Intrusion Detection System to analyze network traffic patterns. However, this should not be the only means of defense. It is important to have a security policy in place to discourage unauthorized access. Also, it's helpful to have an emergency response team in place; the team members should be trained in how to respond to attacks when they're detected. [8]

Distributed Password Cracking

The phase "Distributed password cracking" refers to the process of spreading the workload of a classic password-cracking tool across several machines. Password cracking is a very fast guessing game. Thus, by distributing the workload across multiple machines at a time, cracking the passwords is much more efficient. Most passwords are predictable, such as user ids, dictionary words, and personal information, making password-cracking tools very successful.

How do password thieves work? Passwords are encrypted and stored in a database dependent upon the operating system. For instance, Windows systems store passwords in the SAM database, while passwords from Unix systems can be retrieved from /etc/passwd or /etc/shadow files. Password cracking tools automate the guessing game by using variations of dictionary and brute-force attacks to guess a password. [9, p.34] To combat this attack, administrators should enforce policies that require users to create "strong" passwords that resist password-cracking algorithms.

Types of Password Cracking

Here are some methods commonly used for cracking passwords.

- **Manual**. An intruder distributes the workload of the password-cracking tool to manually to several machines. For example, an intruder can break a password dictionary of 100 passwords into 10 files and place each file on a different machine. Then the intruder can run the tool on 10 machines distributing the workload.

- **Automatic**. Several new releases of password-cracking tools, such as LC3, automate the spreading of the workload as they coordinate the computing resources during the attack. [9,p. 35]

Types of Distributed Password Cracking Tools

Unfortunately, tools for cracking passwords are readily available on the Web. Some of the most popular such tools include:

- **Mio-Star**. A distributed Unix-based cracker that allows connections to multiple machines as long as perl script is installed.

- **Saltine Cracker**. Due to its cross-compatibility between NT HASH (MD4) and POSIX LibDES Crypt(3) passwords, it is possible to audit POSIX passwords with a Windows system and vice-versa.

- **Slurpie**. A "Slurpie" is a Unix password cracker that can automatically be run on multiple machines simultaneously.

- **John the Ripper**. This tool is a free, cross-platform dictionary-only cracker that has the ability to crack several encryption algorithms. It is mainly designed for Unix, but it can crack NT LanMan hashes. [10]

- **L0phtCrack (LC3)**. Probably the most widely known password tool is L0phtCrack. It can recover NT passwords from "SAM data imported from raw SAM files, from SAM_backup files, from a remote machine using Administrator access and the pwdump-like function, and by sniffing password hashes off the network." [3,p. 178] LC3 allows automated simultaneous connections to multiple computers on the same password audit.

Prevention of Password Cracking

The best defense against successful password cracking is strong passwords. Weak or "easy-to-guess" passwords should be eliminated from every system. Although a strong password policy isn't completely foolproof, as password tools

and dictionaries improve, it is among the best defenses available to the network administrator.

There are several published guides on password security. Here are a few common "best practices" to follow:

- Enforce password length. User passwords should be at least eight characters long.
- Enforce strong passwords. Passwords should include upper- and lower-case and non-alphanumeric characters. Refrain from using personal information, all words or numbers, recognizable words and hacker terms. Create a password that does not have to be written down.
- Enforce password history. Set the requirements to greater than 10.
- Enforce password minimum and maximum age. The minimum password age should be set to more than zero. Most companies set the maximum age from 30 to 90 days.
- Deploy password-filtering programs on authentication servers to enforce password policies, such as the passfilt.dll program from the Windows 2000 Resource Kit.
- Implement an awareness program to emphasize the importance of hard-to-guess passwords.
- Administration should routinely run a password-cracking tool against user passwords to identify weak passwords before a hacker does. [9,p.35-36:12]

Distributed Port Scanning

Port scanning is a tool that is useful to both the "good guys" and the "bad guys" for the same reason—it scans TCP and UPD ports to find open ports that services are running on or "listening." The types of ports open help the user determine the operating system and applications in use and exploit weaknesses. [3, p.43]

Detection of a port scan is more difficult to identify by a target system if several hosts on different networks are performing a scan at the same time. This practice is called distributed port scanning. A user controls the clients from a central server. [13]

Types of Port Scanning

There are several scanning techniques:

- Vanilla: The scan attempts to connect to all 65,535 ports.

- Strobe: Only a few ports are scanned, those ports that are well-known for being exploitable.
- Stealth-scan: Techniques such as SYN or FIN scans are used so the scanned computer cannot log the port scanning activities.
- FTP bounce scan: The scanner goes through an FTP server so that the scanning source cannot be determined.
- Fragmented packets: Fragmented packets are sent by the scanner to penetrate simple packet filter firewalls.
- UDP: UPD ports are scanned to see if they are open.
- Sweep: One port is scanned on multiple systems. [14]

Types of Distributed Port Scanning Tools

At this writing, a number of different port scanning tools are available, including the following:

- **NMAP**. NMAP is one of the most popular scanning tools available. It offers a variety of scanning techniques, such as UDP, TCP SYN, FTP Proxy, ICMP, and Null scan, among others. It also provides remote OS detection, fingerprinting, stealth scanning, decoy scanning, and port detection filtering.

- **NetScan Tools Pro 2000**. Considered to have one of the best Windows-based port scanners available, Netscan also does a lot more than mere port scanning. It offers several utilities like DNS queries, ping sweeps, whois, SNMP walks, and even multitasks between systems so a port scan can be run on one system and a ping sweep can be run on another.

- **SuperScan**. This TCP scanner is fast and flexible. Like Netscan, SuperScan allows for flexible specification of target IPs and port lists. In addition, it comes with some very extensive port lists. [3]

- **ISS Internet Scanner**. This commercial scanner provides several scanning techniques, including TCP, ICMP, and UDP. It offers NETBIOS and DNS utilities, identifies operating systems, and performs fingerprinting.

Port Scanning Prevention

"Federal law enforcement officials are generally in agreement that port scanning is not a crime" [15]. However, there is a fine line between port scanning and hacking. At this point, the best defense against distributed port scanning is to

disable all unneeded services on users' systems. Users should be familiar with the programs and applications running on their systems and configure their systems specifically to that environment's needs. Also, network administrators should make sure Intrusion Detection System and Intrusion Prevention System signatures deployed in their environment are up-to-date. [9, p. 36]

Active Sniffing

Sniffing is the act of capturing data passing by a computer's network interface. In simple terms, sniffing is eavesdropping. Administrators use sniffers to capture atypical data packets to help troubleshoot network issues. Hackers use sniffers to capture sensitive data that may be useful to the hacker, such as passwords, email, or files.

Traditional network sniffers are passive. They wait for network traffic to pass by so the data can be gathered before it arrives to its destination. However, LAN switching and encryption are commonly used to defend networks against network sniffers. [9, p. 39] Enter active sniffing.

Active sniffing is based on a variety of techniques to get around defenses for traditional sniffing tools. These techniques inject traffic into the network to allow an attacker to retrieve data. Several methods for injecting traffic into the network include MAC address flooding, spurious ARP traffic, fake DNS responses, and man-in-the-middle attacks against SSL. [3]

Types of Active Sniffing

Due to ever-increasing features available with sniffer tools, sniffing networks is one of the most dangerous attacks on a corporate network. Let's take a look at some of the common methods for implementing active sniffing.

One popular method is Media Access Control (MAC) flooding. The MAC address is a system's unique hardware ID. Network traffic is routed based on an Ethernet switch's MAC. The switch monitors the traffic to determine which plugs on the switch are associated to which MAC addresses. The switch also detects the MAC addresses for other network interfaces on the LAN and directs traffic appropriately.

MAC Addresses Flooding refers to the act of flooding the LAN with traffic that has bogus MAC addresses. As the switch tries to remember all of the MAC addresses as they attempt to pass, the switch's memory will be exhausted with bogus MAC addresses, causing the switch to fail-over and send traffic to all machines on the LAN. Once the traffic is open on the LAN, an attacker can use a traditional sniffer to recover data.

Another method of active sniffing is spurious Address Resolution Protocol (ARP) traffic. ARP is a protocol for mapping an IP address to a MAC address on a local area network. ARP responses and the mapping of the IP address and MAC address are stored in the ARP cache to minimize ARP traffic on the LAN. ARP supports "gratuitous ARP," which occurs when one computer sends an ARP response to other computers to update its own ARP cache for performance on the LAN. Spurious ARP traffic is a form of sniffing whereby the attacker sends gratuitous ARP messages to the victim's machine and tricks the victim's machine into sending data to the attacker's machine. To avoid suspicion, the attacker enables IP forwarding on his machine so that once he captures the victim's data, it is forwarded to the outside world. This attack requires that the attacker and victim be on the same LAN.

Another technique for injecting packets into a network to sniff involves fake DNS responses. DNS (Domain Name Service) is a very complicated service used to resolve domain names and service location. Attackers can re-route traffic, like ARP traffic, by sending fake DNS responses to a victim's machine. Instead of having to exist on the same LAN as the victim in spurious ARP traffic, the attacker just has to sit on a network between the victim and the DNS server in question. Once a client sends a request to a DNS server on the Internet, the attacker can circumvent the victim's request, sending a fake DNS response with the attacker's IP address to the victim's machine. The victim will then, without knowing it, send data to the attacker's machine; that information can then be viewed before being forwarded to the true destination.

Finally, man-in-the-middle attacks against cryptographic protocols have began to surface due to the rise in companies sending data over secure protocols. These attacks occur when fake DNS responses are sent to a victim's machine by an attacker so a new secure session (SSL, HTTPS, etc.) is established through the attacker's machine, frequently by a tool such as Webmitm. Once the attacker circumvents a victim's DNS request to an outside host, he establishes a secure session with the victim. A tool such as Webmitm can also establish a secure session for the attacker on the actual outside source that the victim wants to access. The tool is able to decrypt traffic for the attacker as the victim sends data to the outside source and encrypts it back before reaching its true destination. The biggest trick for the attacker is that he must send a digital certificate that the victim must accept. Because most users do not have the knowledge to understand the warning pop-ups on certificates, they generally will accept the certificate without thought. [9, p. 40-45]

Active Sniffing Tools

There are many good sniffing tools available for network administration. Dsniff, TCPDump, and Snort are among the most commonly used tools.

- Dsniff is a Unix-based collection of utilities used for auditing and penetration testing. Three utilities provided that can be used to actively retrieve network traffic are arpspoof, dnsspoof, and macof. [10]
- TCPDump is a popular packet sniffing tool that is supported on Unix and Windows.
- Snort is a real-time packet analysis and logging tool. [3, p.30]

Defenses Against Active Sniffing

There are several measures that administrators can take to proactively defend their networks against active sniffing. Unfortunately, however, none are foolproof. Here are some of the rules of thumb.

At the top of the list, administrators should encrypt data, at least sensitive data, and implement secure protocols such as SSH, HTTPS, and IPSec. Whenever possible, do not telnet to perimeter defenses, sensitive servers, or PKI systems. Never use untrusted certificates when sending data across the network with an SSL session. Although it may not be possible financially, it is a good practice to use switches instead of hubs for security's sake. If switches are used on one's network, enable port-level security on network housing sensitive data. Finally, sensitive machines (i.e. DMZ servers) should be implemented with static ARP tables. [9, p. 45-46]

Kernel-Level Rootkits

Rootkits are a collection of tools that a hacker uses to attack an operating system. After obtaining user access to a system, the hacker installs the rootkit on the operating system. This toolkit often consists of utilities that monitor traffic and keystrokes, maintain backdoors, alter log files, and attack other systems on the network. Kernel-level rootkits actually alter the kernel itself instead of just taking advantage of application-level programs. The kernel is the brain of the operating system controlling resources like disk, system processor and memory.

Programs like Tripwire can discover traditional rootkits since, they rely on the kernel to check the integrity of application programs, this isn't really the case with kernel-level rootkits. They corrupt the kernel, providing backdoor access to the system while hiding the hacker's identity. Most kernel-level rootkits provide execution redirection, file hiding, and process hiding, techniques which provide the hacker the complete ability to manipulate the machine. [9, p. 47-48]

Kernel-Level Rootkit Programs

Although the availability of kernel-level rootkits does not equal that of traditional rootkits, their popularity is increasing. Here are two of the most common toolkits:

- **Knark**. Developed for Linux 2.2 kernels, Knark offers utilities to hide or unhide files, exec-redirection, execute commands remotely, gain root access and hide strings in /proc/net/tcp and proc/net/udp.

- **Windows NT kernel-level Rootkit**. This is a Windows-based kernel-level rootkit that offers registry key hiding and EXE redirection. [10]

Defenses Against Kernel-Level Rootkits

The first and best defense against kernel-level rootkits is to keep attackers from gaining administrative or superuser access to one's systems. Attackers need this high-level access to install the rootkit, thus, without that kind of access, the installation is impossible. It is important to disable all unneeded services and apply relevant security patches as necessary. Lastly, one can deploy kernels that do not support loadable kernel modules (LKMs) which allow kernels to be dynamically modified. Unix can be built without this support, but Solaris systems cannot. [3, p. 559-560]

Conclusion

This paper presents a general overview of newer hacks, tools used to perform these hacks, and methods of prevention. As long as the Internet is available, external security threats will continue to grow. While the type of attack methods may not change tomorrow, the architecture behind them will evolve as more sophisticated and easier to use.

As long as new protocols, hardware, and operating systems are developed, hackers will continue to identify vulnerabilities in networks and systems and plan attacks that will range from the harmless to the destructive. The best defense against these attacks by administrators will be awareness and prevention.

In order to provide optimum protection for corporate IT assets, it is imperative that administrators set aside time to familiarize themselves with the latest vulnerabilities specific to their environments. There are many security solutions available to harden networks and their systems. Administrators should routinely assess their systems and implement appropriate safeguards to ensure the

systems are up-to-date and protected against new vulnerabilities. Those safeguards include, at a minimum applying patches hotfixes in a timely manner, implementing defense programs, better network designs, and security awareness programs.

References

[1] CERT Coordination Center. "Overview of Attack Trends." Carnegie Mellon University. 2002. URL: http://www.cert.org/archive/pdf/attack_trends.pdf. (May 26, 2003).

[2] Tanase, Matt. "Barbarians at the Gate: An Introduction to Distributed Denial of Service Attacks." December 3, 2002. URL: http://www.securityfocus.com/infocus/1647 (May 20, 2003).

[3] Joel Scamby, Stuart McClure, and George Kurtz. Hacking Exposed: Network Security Secrets & Solutions 2nd Edition. Berkeley: Osborne/McGraw-Hill, 2001.

[4] "Denial of Service Attack Threat Analyses." 2002. UK Security Online Ltd. URL: http://www.uksecurityonline.com/threat/dos.php. (May 15, 2003).

[5] CERT Coordination Center. "Distributed Denial of Service Tools." Carnegie Mellon University. 2001. URL: http://www.cert.org/incident_notes/IN-99-07.html. (May 15, 2003).

[6] Dittrich, David. "The "Stacheldraht" Distributed Denial of Service Attack Tool." University of Washington. December 31,1999. URL: http://staff.washington.edu/dittrich/misc/stacheldraht.analysis. (May 20, 2003).

[7] Robichaux, Paul. "Distributed Denial-of-Service Attacks and You." Microsoft Technet. 2003. URL: http://microsoft.com/technet/security/bestprac/ddosatku.asp?frame=true. (May 16, 2003).

[8] Dunne, Danielle. "What is a Denial-of-Service Attack?" CXO Media, Inc., June 14, 2001. URL: http://www.darwinmag.com/learn/curve/column.html?ArticleID=115. (May 20, 2003).

[9] Harold Tipton and Micki Krause. Information Security Management Handbook 4th Edition ,Volume 3. Boca Raton: Auerbach Publications, 2002.

[10] Archive Search. URL: http://www.packetstormsecurity.org. (May 25, 2003).

[11] SearchTechTarget. URL: http://searchsecurity.techtarget.com/sDefinition/0,,sid14_gci213306,00.html. 2001-2003. (May 20, 2003).

[12] “Red Hat Linux 8.0: The Official Red Hat Linux Security Guide Chapter 4. Workstation Security.” URL: http://www.redhat.com/docs/manuals/linux/RHL-8.0-Manual/security-guide/s1-wstation-pass.html. (May 15, 2003).

[13] Bechberger, Chris. “Distributed Port Scanner.” Beyond Security Ltd., December 12, 2000. URL: http://www.securiteam.com/tools/6A00H0K0KC.html. (May 20, 2003).

[14] ISS.Net AdvICE. URL: http://www.iss.net/security_center/advice/Underground/Hacking/Methods/Technical/Port_Scan/default.htm. (May 20, 2005).

[15] Poulsen, Kevin. “Port scans legal, judge says.” Security Focus. June 18, 2000. URL: http://www.securityfocus.com/news/126. (May 20, 2005).

[16] Skoudis, Edward. “Cracker Tools and Techniques.” InfoSecurity Magazine. July 2002. URL: http://www.infosecuritymag.com/2002/jul/faster.shtml. (May 15, 2003).