Hacking Techniques in Wired Networks

Qijun Gu, Pennsylvania State University, University Park
Peng Liu, Pennsylvania State University, University Park
Chao-Hsien Chu, Pennsylvania State University, University Park

Keywords: Wired network, Security, Cyber attack, Vulnerability, Hack, Worm, Virus, Internet infrastructure, End system, Enterprise network

Wired networks, especially the Internet, have already been indispensable in our daily activities. However, in the last 10 years, security "disasters" have challenged the design and development of networks and systems. Vulnerabilities in current systems are frequently exploited by hackers and attackers. Cyber attacks have become a more and more serious threat to our society. In order to better protect networks, this article gives an overview on a variety of hacking techniques. This article focuses on the objectives, principles, functionalities and characteristics of different types of hacking techniques in wired networks, and provides in-depth discussions on the common characteristics of cyber attacks, the structure and components of cyber attacks, and the relationships among cyber attacks. These discussions can help security professionals grasp the "soul" of a "new" cyber attack in an easier and quicker way.

INTRODUCTION

Nowadays, wired networks, especially the Internet, have already become a platform to support not only high-speed data communication, but also powerful distributed computing for a variety of personal and business processes every day. However, the principles for designing and

developing a network mainly targeted at providing connection and communication capabilities, until a series of security "disasters" happened on the Internet recently as shown in Figure 1. As a result, without making security an inherent part of the network design and development process, existing networks are very vulnerable to cyber attacks because of various security vulnerabilities. Such vulnerabilities, when being exploited by the hacker, can motivate the development of a variety of hacking techniques. These hacking techniques directly lead to cyber attacks; and these cyber attacks have become a more and more serious threat to our society.

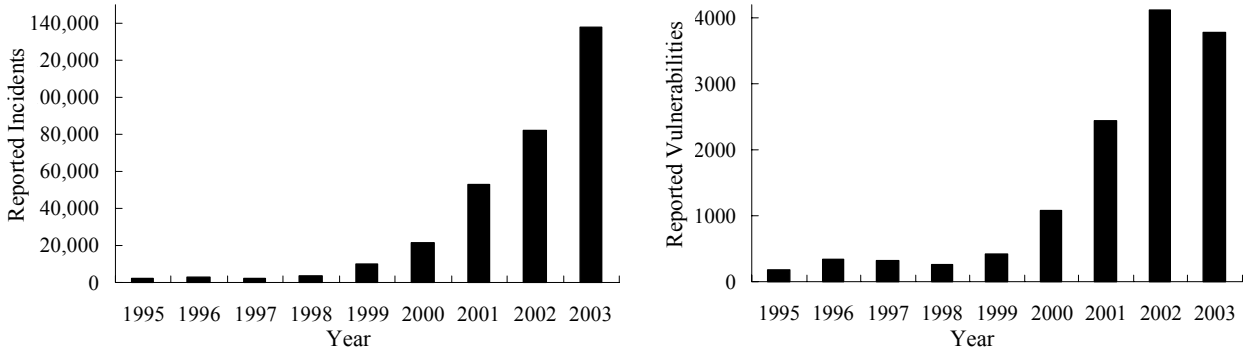

Figure 1. Reported Incidents and Vulnerabilities from 1995 to 2003 [11]

In order to better protect networks, this article tries to give an overview on a variety of hacking techniques. No wonder, the better we understand the hacker, the better networks can be protected. This article will focus on the objectives, principles, functionalities and characteristics of different types of hacking techniques in wired networks, but will not address detailed and in-depth hacking processes, which can be found in several other articles of this handbook. In addition, we only discuss well-known and published vulnerabilities and attacks. Most of these attacks have been prevented by the improved protocols and systems. Although it is not possible to identify all vulnerabilities and attacks, this article will provide in-depth discussions on the common characteristics of cyber attacks, the structure and components of cyber attacks, and the relationships among cyber attacks. These discussions can help security professionals grasp the "soul" of a "new" cyber attack in an easier and quicker way.

This article is organized as follows. In Section 2, the principles of hacking are summarized. We overview the common hacking procedures, review most used hacking toolkits, and illustrate how these tools are employed in hacking. In Section 3, we discuss how hacking techniques can be used to construct attacks on the Internet infrastructure. In Section 4, we discuss how hacking techniques can be used to construct attacks on end systems of the Internet. In Section 5, we discuss how hacking techniques can be used to construct attacks on enterprise network systems. Finally, in Section 6, we conclude this article.

PRINCIPLES OF HACKING

In this article, attacks and hacking techniques are two different concepts that are, nevertheless, closely related to each other. An *attack* typically goes through several steps or phases. In each phase, some attack *actions* will be carried out by the hacker, and these attack actions will typically involve the use of one or more hacking techniques. The hacking techniques involved in

different attack phases could be different. Moreover, an attack or hacking (software) tool may cover several phases of an attack and involve multiple hacking techniques.

Seven Steps of Hacking

No matter how to hack or attack a network, the attacker always takes certain procedures to accomplish his objectives. In general, these procedures fall in one of the following seven steps [3]: reconnaissance, probe, toehold, advancement, stealth, listening post, and takeover, where each step is enabled or helped by its previous steps and prepares for its following steps. These seven steps can serve as a procedural classification of hacking techniques because the hacking techniques used in each step are for the same purpose and share many common characteristics.

Reconnaissance

Reconnaissance is to gather information of the target system or network.

The information of interest may include host names, host addresses, host owners, host machine types, host operating systems, network owners, network configurations, hosts in the networks, list of users, etc. An intruder may start with searching the Internet for references to the target in order to find the domain information of the target. Then the intruder can obtain further information about other machines within that domain such as their host names and network addresses. For example, the intruder can analyze the target web pages to gather useful information about the users of the target system, because most web pages contain user information, such as contact emails or some personal information (name, address, phone number, etc). If the intruder obtains a user account in the target system, he can begin to guess the password. Sometimes, he can even directly contact a person through phone or E-mail to acquire the person's account information.

Probe

Probe is to detect the weaknesses of the target system in order to deploy the hacking tools.

After gathering enough information of the target, the intruder begins to probe the perimeter of the system for potential weaknesses. He can utilize remote exploit tools, which enable the intruder to conduct security surveys and automatically collect and report security-related vulnerabilities of remote hosts and networks. Using these hacking tools, the intruder can find out the remote services the target is providing, such as WWW, FTP, SMTP, finger, X server, etc., by scanning the hosts of the target network. In addition, the intruder can obtain such information as machine names, software names and version numbers. Then, he can refer to the known vulnerabilities of the detected services for further exploitation.

Toehold

Toehold is to exploit security weaknesses and gain entry into the system.

Once a vulnerability is found, the intruder will first exploit this vulnerability to build a connection (or session) between his machine and the target host, and then remotely execute hostile commands on the target. (For example, the intruder can generate an X terminal emulation on his own display.) In this way, a toehold into the target network has been established and the intruder can go further to compromise the system. Gaining entry into the system, the intruder can also search for more critical system information. If the current user identification (UID) is for a

privileged user, the intruder will jump to the stealth step; otherwise, he will get into the advancement phase.

Advancement

Advancement is to advance from an unprivileged account to a privileged one.

In this step, the intruder uses local exploit tools to obtain additional information of the target, such as configuration errors and known vulnerabilities of the operating system. Once finding a local vulnerability, the intruder can advance from an unprivileged UID to a root UID. Then, with the highest level of privileges, the intruder can fully control the target system, steal sensitive data, maliciously modify files, and even delete the entire file system.

Stealth

Stealth is to hide the penetration tracks.

During the probing phase, the intrusion actions are likely to be logged by intrusion detection systems, and during the phases of toehold and advancement, the intruder may leave his activities in the system log. Hence, in order to hide, the intruder will access the local log files and modify the corresponding log entries to remove the traces and avoid detection. He may further replace the system binary code with a malicious version in order to ensure future un-logged and undetected access to the compromised system.

Listening Post

Listening post is to install backdoors to establish a listening post.

In this step, the intruder inserts some malicious programs into the system, such as a stealth tool, a backdoor tool, and a sniffer. These programs ensure that his future activities will not be logged. They report false information on files, processes, and the status of the network interface to the administrators. They also allow the intruder to access the compromised system through the backdoor. With the sniffer tool, the intruder can capture the traffic on the network interfaces. By logging the interesting network traffic, the intruder can better monitor and control the compromised system.

Takeover

Takeover is to expand control (or infection) from a single host to other hosts of the network.

From the listening post, the intruder can sniff a lot of important information about other hosts of the network, such as user names and passwords. The intruder can also obtain information through several other ways. For example, he can check some specific configuration files (e.g., /.rhosts) of the compromised host and find mutually trusted hosts. With these information, the intruder can retake the previous steps to break into other hosts. In this way, he can expand his control to the whole network.

Overview of Hacking Toolkits

In broad sense, hacking toolkits include not only the softwares developed for attacks, but also the human activities for the collection of sensitive information and the penetration into the target system. In the following, we discuss fourteen types of representative hacking softwares and approaches.

Scanners

A scanner is a tool to obtain information about a host or a network. It is developed to probe the networks and report security related information. Serving for different purposes, a scanner is used by both security administrators for securing networks and systems, and hackers for breaking into. Scanners can be broken down into two categories: network auditing tools and host-based auditing tools. Network auditing tools are used to scan remote hosts [21,22,24]. For example, NMAP [22] is a free open source utility for network exploration and security auditing. It can rapidly scan large networks and single hosts. NMAP uses raw IP packets to determine what hosts are available on the network, what services those hosts are offering, what operating systems they are running, what type of packet filters/firewalls are in use, etc. Host-based auditing tools, working in a local system, are used to scan a local host and report its security vulnerabilities [12,27]. For example, the COPS package [12] can help identify file permission problems, easy-to-guess passwords, known vulnerable services and improperly configured services.

Sniffers and Snoopers

A sniffer monitors and logs network data [16]. The network traffic that passes through a host's network interface usually contains user name-password pairs as well as other system information that would be useful to an intruder. In a network where data is transmitted without encryption, an intruder with physical access to the network can plug in a sniffer to monitor the network traffic and obtain necessary information to access other hosts in the network. A snooper, also known as spyware, monitors a user's activities by snooping on a terminal emulator session, monitoring process memory, and logging a user's keystrokes [26]. By watching the user's actions, an intruder can obtain useful information to attack other users on the computer or even other systems in the network.

Spoofing Tools

In a network, a data packet always contains the source address field, which can expose the source of the intruder if he sends malicious packets. Hence, in order to hide and avoid detections, the intruder uses spoofing tools to forge another source address that is usually the address of another host or a nonexistent address. The spoofed address can be an IP address or a physical address, depending on the type of the network. Another usage of spoofing tools is to gain access to a network from outside. If the firewall of the target network is not configured to filter out incoming packets with source addresses belonging to the local domain, it is possible for an intruder to inject packets with spoofed inner addresses through the firewall.

Trojan Horse

The concept of Trojan horse comes from the legend in which the Greeks sneaked into the Trojan city by hiding in a huge, hollow wooden horse and defeated the Trojans. A Trojan horse in a computer system is thus defined as a malicious, security-breaking program, which is a piece of executable code hiding in a normal program. When the normal program is opened or executed, the hidden code will perform some malicious actions silently, such as deleting critical system files. The Trojan horse is spread in a disguised way. It presents itself as a game, a web page, or a script that attracts people. It may come from an Email with your friend as the sender or an online advertisement. But if the receiver opens it, the malicious code will commit the unsolicited actions.

Password Crackers

A password cracker is to find a user's password [17,23]. It is used by both computer crackers and system administrators for recovering unknown or lost passwords. There are three major types of crack approaches. The first type is the smart guessing cracker, which infers or guesses the password based on user's information, such as user name, birthday and phone number. The second is the dictionary-based cracker, which generates a large set of possible passwords, called *dictionary*, from a collection of words and phrases. These two types of crackers are smart and quick, but may not work if the password is randomly generated. The third type is to enumerate and test all possible passwords in a brute-force way. When the password is extremely long, the last type of password cracker will usually take a tremendous amount of time.

Denial of Service Tools

A DoS (Denial-of-Service) tool is used by an attacker to prevent legitimate users from using their subscribed services. DoS attacks aim at a variety of services and accomplish the objective through a variety of methods [14]. Attackers can flood the target network, thereby throttling legitimate network traffic; can disrupt connections between two machines, thereby denying access to the service; can prevent a particular individual from accessing the service; and can disrupt the service to a specific system or person. Different from inappropriate use of resources, DoS tools explicitly and intentionally generate attack packets or disrupt the connections. For example, they can consume scarce or non-renewable resources with a large number of ICMP echo packets, break network connectivity with SYN flooding, alter network configuration by changing the routing information, or even physically destroy network components.

Stealth and Backdoor Tools

Backdoors are programs furtively installed in the target system. They are malicious replacements of critical system programs that provide authentication and system reporting services. Backdoor programs provide continued and un-logged use of the system when being activated, hide suspicious processes and files from the users and system administrators, and report false system status to the users and system administrators. They may present themselves as an existing service, such as FTP, but implant a function to accept controls and execute commands from the intruder. They can also be a new service, which may be neglected because they hide their processes and do not generate noticeable network traffic.

Malicious Applets and Scripts

A malicious applet or script is a tiny piece of code, which is written in web compatible computer languages, such as Java, Jscript and Vbscript. The code is embedded in a web page, an email or a web-based application. When a person accesses the web page or opens the email, the code is downloaded to his personal computer and executed. The code may misuse the computer's resources, modify files on the hard disk, send fake e-mail, or steal passwords.

Logic Bombs

A logic bomb is a piece of code surreptitiously inserted into an application to perform some destructive or security-compromising activities when a set of specific conditions are met. A logic bomb lies dormant until being triggered by some event. The trigger can be a specific date, the number of execution times (of the code), a random number, or even a specific event such as

deletion of a specific file. When the logic bomb is triggered, it will usually do something unsolicited, such as deleting or changing files. Logic bombs may be the most insidious attack since they may do a lot of damage before being detected.

Buffer Overflow

A buffer overflow tool launches attacks by inserting an oversized block of data into a program's input buffer and stack to enable an intruder to execute a piece of malicious code or destroy the memory structure [13]. When a program receives a block of input data, it puts the data into its input buffer. Without the boundary checking, the intruder can write data past the end of the buffer and overwrite some unknown space in the memory. At the same time, the intruder carries the malicious code in the oversized data block. If the unknown space is a part of the system stack that records the return addresses, the overwritten part may change the normal return address to the address pointing to the malicious code. Hence, when the return address is fetched for execution, the malicious code, instead of the original code, will be executed.

Bugs in Software

A piece of software is vulnerable once it is released. First, it typically contains unknown bugs. More complex it is, more bugs it may have. If an intruder finds a bug before it is fixed or patched, he can exploit it to hack a system. For example, the unchecked buffer size is a bug for possible buffer overflow attacks. Second, for the purpose of developing software, the developers usually write some codes for debugging. These debugging codes generally give the developers a lot of authorities. In case these codes are not removed from the released version, the intruder can utilize them for attack.

Holes in Trust Management

Trust management is crucial for a large-scale security system. Due to the complexity of trust management, mistakes in managing and configuring trust relationships may happen in many cases and leave holes for an intruder to gain an authorized access as an unauthorized user. For example, logic inconsistence could be such a hole. Assume that there are three parties, an intruder, a database, and a school. The database trusts the school, but does not trust the intruder. However, if the school trusts the intruder (maybe an adolescent student), the intruder can access the database through the school.

Social Engineering

Social engineering is a tactic to acquire access information through talking and persuasion. The target person is a user who can access the computer system desired by the intruder. The intruder may pretend to be a salesman, a consultant, a listener, a friend of the user, or whatever roles that the user does not suspect when they are chatting and exchanging information. The intruder thus can obtain valuable information, such as passwords, to gain access to the system.

Dumpster Diving

Trash is not trash in the eyes of a serious hacker. Trash usually contains shattered and incomplete information. The intruder can sift through garbage of a company to find and recover the original information so that he can break into the company's computers and networks. Sometimes, the information is used as an auxiliary to help intrusion, such as making social engineering more credible.

Classifications of Hacking Toolkits

Each of the hacking toolkits can help hackers to achieve certain objectives. They may be applied in different hacking phases, provide different information, and used in different attack scenarios. Accordingly, we classify them and illustrate how they may be used.

Procedural Classification

As shown in Table 1, a hacking toolkit can be used in one or several penetration steps, and different penetration steps usually need a different set of hacking toolkits. In the reconnaissance step, an intruder wants to gather information about the target system or network. He needs scanners to collect information of computers, user accounts, and services of the target. He may also apply social engineering and dumpster diving to facilitate the information collection. Then, in the second step, he probes the system for weaknesses. He uses scanners and sniffers to capture the activities of the target system and network and analyze possible security holes and vulnerabilities.

Table 1. Procedural Classification

Procedures	Toolkits
Reconnaissance	Scanners, Social engineering, Dumpster diving
Probe	Scanners, Sniffers
Toehold	Spoofing tools, Malicious applets and scripts, Buffer overflow tools, Password crackers, Software bugs, Trojan horses, Holes in trust management
Advancement	Password crackers, Software bugs
Stealth	Stealth and backdoor tools
Listening post	Stealth and backdoor tools, Sniffers and snoopers, Trojan horses
Takeover	Scanners, Sniffers, Spoofing tools, Malicious applets, Buffer overflow tools, Password crackers, etc.

Knowing the weaknesses, the intruder tries to gain entry into the system. In this step, the useful toolkits include spoofing tools, malicious applets, buffer overflow tools, password crackers, etc. These tools enable him to break into the system remotely or obtain authorized local access. Once getting inside the system, he tries to advance from an unprivileged account to a privileged account. In this step, he can first find some system files containing the information of privileged accounts, and then use password crackers to get the name-password pairs. He can also exploit the system bugs to advance his privileges.

Now the system is under control. The intruder hurries to hide his traces before the administrators find him. So he will use stealth and backdoor tools to remove his traces while continuing his access to the system. To keep monitoring the hacked system, the intruder establishes a listening post. He uses sniffers and backdoor tools to watch system activities and report crucial information, so that he can fully control the compromised system and prepare for further attacks.

Finally, the intruder expands his control from a single host to other hosts in the network. The previous tools will be used again. Scanners, sniffers, spoofing tools, malicious applets, buffer overflow tools and password crackers are all necessary tools for him to break into other hosts.

Functional Classification

According to the functions of the hacking toolkits, they can be broken into four categories, namely information gathering tools, remote exploit tools, local exploit tools, and DoS tools as shown in Table 2.

Table 2. Functional Classification

Functions	Toolkits
Information gathering	Scanners, Sniffers, Backdoors, Social engineering, Dumpster diving
Remote exploit	Spoofing tools, Malicious applets, Buffer overflow tools, Trojan horses, Holes in trust management
Local exploit	Password crackers, Software bugs
DoS	Denial of service tools

Information gathering tools are used to obtain the target's system information before and after attack. These tools include scanners, sniffers, backdoors, etc. Before attack, scanners and sniffers are mostly used to detect the target's vulnerabilities; while after attack, the intruder will monitor the compromised system's activities and keep the control of the victim by installing sniffers and backdoors.

To break into a system and obtain the desired privileges, the intruder needs either remote or local exploit tools. If the intruder does not have any account in the target system, he will use remote exploits tools, which enable the intruder to penetrate into a remote host. Spoofing tools, malicious applets and buffer overflow tools are mostly employed. These tools allow the intruder to compromise the target without much prior knowledge about the target.

If the intruder has already had a local account, he can use local exploit tools to gain unauthorized privileges on the computer. He can use password crackers to guess the password of the root account. If he succeeds, he can gain the root privilege. Another method is to exploit the system bugs or un-removed debugging codes. These system holes enable the intruder to execute programs with only an unprivileged account.

The fourth category is denial-of-service tools. DoS tools will typically apply some information gathering (or reconnaissance) techniques first. But instead of trying to break into the target system, as both remote exploit tools and local exploit tools want to do, DoS tools try to disrupt the services provided by the target system.

ATTACKS AGAINST THE INTERNET INFRASTRUCTURE

It is hard to give a precise meaning of the Internet infrastructure. In general, the infrastructure includes all hardware and protocols that support the communication between two hosts inter networks, such as routers, gateways, fibers and cables (as hardware) and TCP, ICMP and BGP (as protocol). In this section, we use several representative attacks to demonstrate the principles of infrastructure-oriented attacks, which may directly impact our daily usage of the Internet. Readers can identify other similar attacks against the Internet infrastructure.

Figure 2 shows a diagram of a daily activity in the Internet, e.g. browsing a webpage. In this browsing procedure, a user first puts the text-based URL (Uniform Resource Locator) of the web page into his browser. His computer then sends a DNS query to the corresponding DNS server to resolve the IP address of the web server, and starts a HTTP session with the web server to retrieve the webpage. The HTTP session is based on the TCP/IP communication, which ensure the feasibility and reliability of the browsing. The webpage is retrieved in a series of data packets, which are routed through a sequence of routers according to their embedded IP headers. In this process, three basic components of the Internet are involved, i.e. DNS, TCP/IP, and routing. Accordingly, in the following subsections, we discuss attacks against these components launched by attackers in different domains and networks.

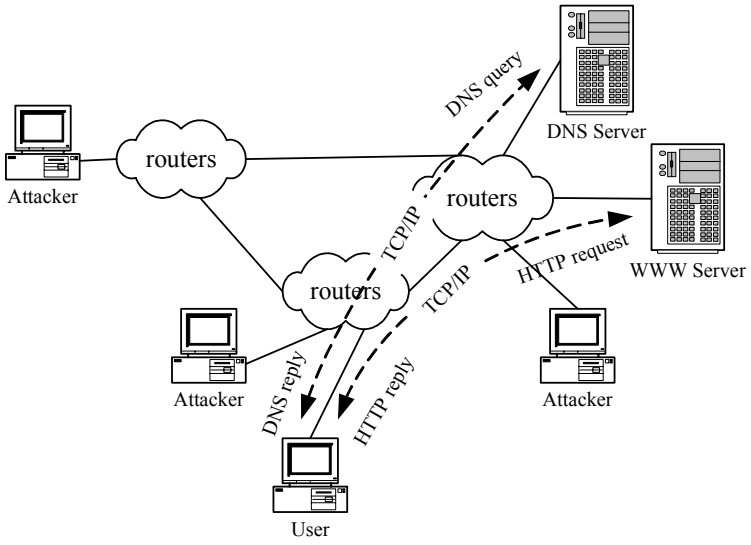

Figure 2. Surfing in the Internet

Attacks against DNS

The DNS (Domain Name System) is a distributed database to provide mapping between host names and IP addresses. A domain name is divided into a series of zones separated by periods, and all names form a name tree. For example, “www.mysite.com” is a domain name, in which “com” is one of the root zones of the name tree, and “mysite” is a branch of “com”, and “www” is a branch of “mysite”. A DNS server resides at a certain level of the name tree and contains name-address mapping information of some zones and the corresponding subzones.

Forward DNS mapping means that a host queries the DNS server for the address of a domain name. Inverse DNS mapping means address-to-name mapping, i.e., a host queries for the domain

name of an address. The response to a DNS query may contain the address or the name that is desired, a pointer to the proper DNS server if the information is not contained within the current zone, or an error indication if the record requested does not exist. The mapping can be multi-names to multi-addresses and vise versa. In general, hosts that use the DNS maintain a local cache to record returned DNS entries. All these records contain a Time-to-Live field set by the creator. At the end of that period, the cached record must be discarded.

In [2], a famous DNS attack is identified. The essence of the DNS attack lies in that the attacker controls a DNS server for the target zone and is able to make any malicious forward and inverse mapping. Consequently, the attacker can make the target host believe that a remote host is trusted. In the early Berkeley version UNIX, the attacker can exploit this attack to gain access to the target host from an untrusted host [2]. To illustrate, assume that the target host is "target.com" with IP address 190.190.190.190, the attacker's host is "attack.com" with IP address 180.180.180.180, and the target host trusts "trust.com". Before attack, the attacker changes the inverse mapping so that the attacker's IP address is associated with "trust.com". When the attacker attempts to "rlogin" to "target.com" from the attacker's machine, the target machine will try to validate the name of the attacker's machine, that is, it sends the DNS server a query with the attacker's IP address. However, because the DNS has been modified by the attacker, the DNS server will reply to the target host that 180.180.180.180 is associated with the domain name "trust.com". Hence the target host believes that one of its trusted hosts, i.e. "trust.com", is trying to connect. Thus the remote login is accepted and the attacker obtains the access. Forward DNS mapping can also fail because a compromised DNS server can return false IP addresses.

The attacker can also exploit the DNS attack to go inside a victim's network [15]. To illustrate, assume that "trust.com" and "target.com" are in the same network segment. The attacker first makes a name-to-address mapping so that "attack.com" has two IP addresses: the IP address of "target.com", namely 190.190.190.190, and its own IP address 180.180.180.180. If, on host "trust.com", the victim occasionally visits a web page on "attack.com", an embedded malicious applet may be downloaded to the victim's browser and run. The applet asks to create a network connection to "attack.com". The victim's Java virtual machine first looks up the address of "attack.com" to make sure that the applet does come from "attack.com". Not surprisingly, the Java virtual machine will get the IP address pair (190.190.190.190, 180.180.180.180) and it will compare this address pair with the address of the machine from which the applet came, i.e. 180.180.180.180. Since the pair includes the address 180.180.180.180, the Java virtual machine allows the connection. However, the Java virtual machine actually connects to the first address, namely 190.190.190.190 (i.e., "target.com"). Hence, the attacker now gets into the victim's network with a connection from an inside host "trust.com" to another inside host "target.com".

Attacks against TCP/IP

TCP/IP is the basis for data transmission in the Internet. The TCP/IP suite includes a set of protocols to control and guarantee worldwide information exchange, such as the TCP protocol, the Routing Information Protocol (RIP), and the Internet Control Message Protocol (ICMP). Despite inspiring features, the security flaws inherent in the TCP/IP suite are exploited by attackers to disrupt the Internet. Although we cannot enumerate all the attacks related to TCP/IP, we discuss several widely known TCP/IP vulnerabilities in this section.

At the TCP layer, an attacker can predict the TCP sequence number to construct a TCP packet without receiving any responses from the server and thus impersonate a trusted host [1]. A normal TCP connection is established according to the 3-way handshake protocol. The client sends to the server a SYN message which includes an initial sequence number SN_C. The server acknowledges it and replies a SYN message which includes its current sequence number SN_S and a piggybacking ACK with SN_C. The client acknowledges this reply by sending an ACK message with SN_S. If the procedure succeeds, a TCP connection is established and the client starts to send data packets. In case the value of SN_S is generated in a predictable way, for example, SN_S is increased by a constant amount every second, the intruder can impersonate a trusted host. In particular, the intruder first sends to the server a SYN message with a spoofed source address that belongs to a host trusted by the server. Although the server replies to the trusted host and the intruder may not receive the reply, the intruder can still forge the ACK message with the predicted SN_S. Hence, even if the trusted host does not request any TCP connection, the intruder can still successfully establish one connection in the name of the trusted host.

An attacker can also hijack a TCP connection to disconnect the target from the server [18]. First, the attacker sniffs for packets belonging to the connection between the client and the server. Thus he can obtain the corresponding IP addresses, port number and sequence numbers. Then, the attacker waits to get a packet whose ACK flag is set from the target to the server. The acknowledgement number SN_S in the packet is the sequence number of the next packet that the target is expecting. The attacker thus forges a packet using the server's address as the source address and the client's address as the destination address. In the packet, the RST flag is set, and the sequence number is set to SN_S. When the target receives the packet, the TCP connection is reset, and thus disconnected.

At the IP layer, the attacker can exploit the holes inside the ICMP protocol for attack [1]. One of such attacks is involved with the ICMP redirect message, which is used by gateways to advise hosts of better routes. If an intruder wants to set up a false route for the target, he first penetrates into or claims to be a secondary gateway available to the target. Then he sends a false TCP open-connection packet to the target with a spoofed source address, which is the IP address of a trusted host. The target will reply to the trusted host by finding a route through the primary gateway. During the process, the intruder sends a false redirect message, which refers to the bogus connection from the trusted host, through the secondary gateway, and to the target. This packet appears to be a legitimate ICMP control message, and thus the target will accept this routing change. If the target updates its routing table accordingly, future traffic from the target to the trusted host will be directed to the secondary gateway, which is under control of the intruder. Then, the intruder may proceed to spoof the trusted host, and establish connections to the target.

ICMP may also be used for Distributed Denial of Service attacks (DDoS) [4]. In an IP network, a packet can be directed to an individual machine or broadcasted to the entire network. When a packet is sent to an IP broadcast address from a machine in the local network, that packet will be delivered to all machines on that network. Hence, when an attacker sends an ICMP echo request packet to a network, in which the destination address is the network broadcast address and the source address is the target's address, any machine on the network will send an ICMP echo reply packet back to the target after receiving the broadcasted ICMP packet. Therefore, the amount of ICMP echo traffic directed to the target could be tremendous enough to degrade the network performance and cause DoS on the target.

Attacks against BGP

Internet routing is classified as intra-domain routing and inter-domain routing. The Border Gate Protocol (BGP) is the inter-domain routing protocol for routing between Autonomous Systems (AS), also known as routing domains. An AS uses the BGP protocol to announce its IP address ranges, which include the IP prefixes of its inner networks, to its neighbor ASs. Each AS also announces the IP ranges it learns from its neighbors.

As part of the TCP/IP protocol suite, BGP is subject to all the TCP/IP attacks, such as IP spoofing, session stealing, etc. Attacks against BGP mainly target at the provision of false routing information to the network [20]. Attackers can achieve this objective in multiple ways. Since BGP routing data is carried in clear text, it is very easy for attackers to obtain the content and thus take further steps. Since BGP uses TCP connections, the attacker can insert bogus but believable BGP messages into the communication channels between BGP peers. For example, to achieve this, an outside attacker can exploit the hacking technique of TCP sequence number prediction. Moreover, BGP speakers themselves can inject bogus routing information, either by masquerading as another legitimate BGP speaker, or by directly distributing unauthorized routing information. Because BGP does not have peer entity authentication, the man-in-the-middle attack can be easily done. Since BGP does not provide protection against deletion and modification of messages, attackers can change or delete inter-domain routing information. If an attacker removes the relevant routing information of a particular network from the information base maintained by the relevant BGP speakers, other networks of the Internet will not be able to reach this network because they can get only incomplete routing information. If an attacker alters the information so that the route to a network is changed, then packets destined for that network may be forwarded through the detoured route. As a consequence, traffic to that network could be delayed by a longer than expected period of time or even cannot reach the destination. If the route is detoured to the attacker, traffic will be forwarded to the adversary. The attacker can also announce a piece of false information saying that an AS originates a network, and then packets for that network may not be deliverable.

ATTACKS AGAINST END SYSTEMS OF THE INTERNET

In this section, we summarize most famous attacks that happened recently across the Internet as summarized in Table 3. Different from the previous sections where attacks are against the infrastructure of the Internet, the attacks in this section take advantage of the Internet infrastructure and target the hosts or end systems of the Internet. Most of these attacks exploit vulnerabilities in softwares, operating systems and protocols above the transport layer. They impacted hundreds and thousand of computers connected to the Internet.

Morris Worm

On the evening of November 2 1988, a self-replicating program was released to attack the Internet [25]. The program, known as the Morris worm later on, invaded VAX and Sun-3 computers running versions of Berkeley UNIX, and used their resources to attack more computers. Within the space of hours this program had spread across the U.S. infecting thousands of computers and making many of them unusable due to the burden of its activity. Although the worm was designed to spread itself to as many computers as possible and took only a tiny process to be unnoticeable, it did work strikingly due to mistakenly underestimating its

spreading power and overload. As time went on, some of these affected machines became so loaded with running processes (because they were repeatedly affected) that they were unable to continue any processing. Some machines failed completely with their swap space or process tables exhausted.

Table 3. Major Attacks against End Systems before 2004

Names	Date	Target systems
Morris	Nov. 2, 1988	VAX and Sun-3 running Berkeley UNIX
Melissa	Mar. 26, 1999	Systems running Microsoft Word 97 and Word 2000
Sadmind	May 8, 2001	Systems running unpatched Microsoft IIS or Solaris up to version 7
Code Red I	Jun. 19, 2001	Microsoft Windows NT 4.0, Windows 2000 and other systems with IIS 4.0 or IIS 5.0 enabled and Indexing services installed
Code Red II	Aug. 6, 2001	
Nimda	Sep. 18, 2001	Systems running Microsoft Windows 95, 98, ME, NT, and 2000 with IIS
SQL Slammer	Jan. 25, 2003	Systems running Microsoft SQL Server 2000 and Microsoft Desktop Engine (MSDE) 2000
W32/Blaster	Aug. 11, 2003	Systems running Microsoft Windows NT 4.0, Microsoft Windows 2000, Microsoft Windows XP, and Microsoft Windows Server 2003 with RPC

The worm has two parts, a main program and a vector program. At first, the worm starts stealthily on a VAX or Sun-3 computer. It gathers the information of authorized user accounts from the file “passwd”. Then it begins to break into user accounts. It chooses a possible password either based on the user information or a dictionary. It then encrypts the candidate password and compares the result with the list of encrypted passwords kept in the “passwd” file. If it succeeds in finding a username-password pair, it will go on to break into other hosts using these cracked user accounts.

To break into a remote host, the worm selects a host whose name is contained in a specific set of local files, such as “hosts.equiv”, “.rhosts” and “.forward”, and tries “rsh”, “fingerd”, or “sendmail” to execute a malicious process on the remote host. Through “rsh”, it establishes a connection from the remote host to the current host, copies the vector program to the remote host, and then compiles and runs it. The vector program in turn copies the whole worm to the remote host and thus generates a new worm on the remote host. For “fingerd”, it exploits the buffer overflow bug by sending a specific string of 536 bytes to the fingerd daemon. The string overflows the daemon’s input buffer and changes the stacked return address for executing the worm’s self-regeneration process. For “sendmail”, the worm takes advantage of a piece of unremoved debugging code. The worm sends to the remote host an email message with the DEBUG flag set and a carefully constructed recipient string. This string can pass the body of the

message to a command interpreter that will execute a malicious process embedded in the message body on the remote host.

Once the worm resides on a remote host, the whole process starts over again. In addition, the worm takes several measures to avoid local detection. For example, it sets its process name as "sh", forces the core dump size to zero, deletes the copied programs, etc.

Melissa

On March 26, 1999, a Microsoft Word 97 and Word 2000 macro virus, named Melissa, propagated widely via email attachments [5]. Its widespread attack affected a variety of sites throughout the Internet. Human actions, e.g., a user opening an infected Word document, are typically required for this virus to propagate. Nevertheless, it is possible that under some mailer configurations, a mailer might automatically open an infected document received in the form of an email attachment. This macro virus is not known to exploit any new vulnerability. While the primary propagation mechanism of this virus is via email, any way of transferring files can also propagate the virus.

In Melissa attack, the email attachment is a .DOT Word document that contains a piece of malicious macro code. If an infected document in Word97 or Word2000 is opened, the embedded macro code will infect the "Normal.dot" template and cause any documents referencing this template to be infected with this macro virus. If the infected document is opened by another user, the macro virus included in the document will be executed. Indirectly, this virus could cause serious denial of service on mail servers. Many large sites have reported performance degradation with their mail servers as a result of the propagation of this virus.

The Melissa macro virus propagates in the form of an email message containing an infected Word document as an attachment. The email message contains a specific subject header "Important Message From <name>", where <name> is the full name of the user sending the message. The body of the message contains two sections. The first section contains the deceiving text "Here is that document you asked for ... don't show anyone else ;-)". The next section is a word document, which contains references to some pornographic web sites.

The malicious code embedded in the attachment is actually a piece of VBA (Visual Basic for Applications) code associated with the "document.open" method. When a user opens an infected .doc file with Microsoft Word97 or Word2000, the macro virus is immediately executed if macros are enabled. Upon execution, the virus first lowers the macro security settings to permit all macros to run when documents are opened in the future. Then the macro checks the registry key to decide if the system is already infected. If not, the virus proceeds to propagate itself by sending an email message in the format described above to the first 50 entries in every Microsoft Outlook MAPI address book. For successful propagation, the infected machine should have Microsoft Outlook installed. The macro then infects the "Normal.dot" template file, and as a result, any newly created Word document will be infected. Since un-patched versions of Word97 may trust macros in templates, the virus may be executed without warning. Finally, if the minute of the hour matches the day of the month at this point, the macro inserts into the current document the message "Twenty-two points, plus triple-word-score, plus fifty points for using all my letters. Game's over. I'm outta here."

Melissa is not the only (macro) virus that propagates itself through email attachments. They are many other viruses like Melissa, such as the I Love You virus (in year 2000) and MyDoom (in year 2004).

Sadmind

On May 8, 2001, a self-propagating malicious worm, Sadmind/IIS, was created [7]. The worm uses two vulnerabilities to compromise systems and deface web pages. It affects systems running unpatched Microsoft IIS and systems running unpatched Solaris up to version 7. Intruders can use the vulnerabilities exploited by this worm to execute arbitrary code with root privileges on vulnerable Solaris systems, and arbitrary commands with the privileges of the IUSR_machinename account on vulnerable Windows systems.

To compromise the Solaris systems, the worm exploits a buffer overflow vulnerability in the Solstice sadmind program. It overwrites the stack pointer within a running sadmind process. Since sadmind is installed as root, it is possible for the attacher to execute arbitrary code with root privileges on a remote machine. Then the worm automatically propagates itself to other vulnerable Solaris systems. It adds "+ +" to the ".rhosts" file in the root user's home directory, and modifies the "index.html" file on the host Solaris system after compromising 2,000 IIS systems. It also establishes a root shell listening on TCP port 600, creates certain directories to hold the worm, and runs some processes.

After successfully compromising the Solaris systems, the worm installs software to attack Microsoft IIS web servers. It uses a vulnerability of the IIS systems, i.e. an intruder can encode the relative reference with certain Unicode characters to execute arbitrary commands with the privileges of the IUSR_machinename account on vulnerable Windows systems. If an IIS is compromised, the worm will modify the corresponding web pages.

Code Red I and Code Red II

In 2001, two worms exploited the same vulnerability to disturb the Internet, Code Red I on July 19 and Code Red II on August 6 [6]. Both of them exploited the vulnerability of buffer overflow bugs in Microsoft IIS Indexing Service DLL. They affected Microsoft Windows NT 4.0 with IIS 4.0 or IIS 5.0 enabled and Index Server 2.0 installed, Windows 2000 with IIS 4.0 or IIS 5.0 enabled and Indexing services installed, and other systems running IIS. Totally more than 250,000 hosts suffered from their attack.

Both worms attempt to connect to TCP port 80 on a randomly chosen host in order to find a web service. Upon a successful connection to port 80, the attacking host sends a crafted HTTP GET request to the victim, attempting to exploit a buffer overflow bug in the Indexing Service. If the exploit is successful, these two worms begin their execution on the victim host. Unpatched IIS 4.0 and 5.0 servers with Indexing service installed will almost certainly be compromised.

Code Red I changes all web pages requested from the victim server with a default language of English. Servers configured with a non-English language will not experience any change in the served content. The other worm activity occurs based on the day of the month of the system clock. Day 1-19: The infected host will attempt to connect to TCP port 80 of randomly chosen IP addresses in order to further propagate the worm. Day 20-27: A packet-flooding denial of service attack will be launched against a particular fixed IP address. Day 28-end of the month: The worm "sleeps" with no active connections or denial of service.

Compared with Code Red I, Code Red II goes beyond defacing web pages. It exploits the vulnerability to execute arbitrary code in the Local System security context. If the default system language is “Chinese”, 600 threads will be spawned to scan for 48 hours. Otherwise, 300 threads will be created which will scan for 24 hours. It copies “%SYSTEM%\CMD.EXE” to “root.exe” in the IIS scripts and MSADC folders. It places “CMD.EXE” in a publicly accessible directory to allow an intruder to execute arbitrary commands on the compromised machine with the privileges of the IIS server process. It creates a Trojan horse copy of “explorer.exe” and copies it to “C:\” and “D:\”. The Trojan horse “explorer.exe” calls the real “explorer.exe” to mask its existence, and creates a virtual mapping which exposes the “C:” and “D:” drives. Hence, Code Red II causes system level compromise and leaves a backdoor on machines running Windows 2000.

Nimda

On September 18, 2001, a worm, named W32/Nimda or Concept Virus (CV) v.5, propagated in the Internet to attack systems running Microsoft Windows 95, 98, ME, NT, and 2000 [8]. With the worm, intruders can execute arbitrary commands within the LocalSystem security context on machines running unpatched versions of IIS. In the case where a client is compromised, the worm will run with the same privileges as the user who triggered it. The infected computers may suffer from DoS caused by network scanning and email propagation.

Once running on the victim machine, the worm visits all directories in the system (including all those accessible through file sharing) and writes a MIME-encoded copy of itself to the disk using file names with “.eml” or “.nws” extensions (e.g., “readme.eml”). When a directory containing web content (e.g., HTML or ASP files) is found, a snippet of Javascript code is appended to every one of these web-related files for propagation. The worm also enables the sharing of the “C:” drive as C$, creates a Guest account on Windows NT and 2000 systems, adds this account to the Administrator group, and replaces existing binaries with their Trojan horse versions.

The worm spreads in multiple ways. First, it can propagate via an email that contains a base64-encoded attachment, named “readme.exe”. Because IE 5.5 SP1 or earlier renders will automatically run the enclosed attachment, the worm infects the receiving computer. The worm also contains code to resend the infected email messages every 10 days. The target email addresses are found from the user’s web cache and the emails retrieved via the MAPI service. Second, it propagates via open network sharing. A user on another system can trigger the worm in his own computer if he opens a copy of the worm or executes the Trojan horse versions of the legitimate applications in the shared folder. Third, it propagates via browsing compromised web sites. Because the worm adds a piece of Javasript code to all the web files it finds, a user browsing web content on the infected system may download a copy of the worm, which may be automatically executed and thus infects the system. Fourth, the worm exploits the Microsoft IIS 4.0 / 5.0 directory traversal vulnerabilities as the Code Red worm, and scans for the backdoors left by Code Red II and sadmind/IIS worms to propagate.

SQL Slammer

On January 25 2003, a worm, referred as SQL Slammer, W32.Slammer, or Sapphire, caused varied levels of network performance degradation across the Internet [9]. The worm affects Microsoft SQL Server 2000 and Microsoft Desktop Engine (MSDE) 2000. The high volume of UDP traffic generated by the infected hosts may lead to performance degradation against

Internet-connected hosts or those computers that stay on the same network of a compromised host.

The worm exploits the vulnerability of stack buffer overflow in the Resolution Service of Microsoft SQL Server 2000 and Microsoft Desktop Engine (MSDE) 2000, so that an intruder can execute arbitrary code with the same privileges as the SQL server. It may be possible for an attacker to subsequently leverage a local privilege escalation exploit in order to gain Administrator access to the victim system.

Once the worm compromises a machine, it will try to propagate itself. The worm will craft UDP packets of 376 bytes and send them to randomly chosen IP addresses on port 1434. If such packet is sent to a vulnerable machine, the victim machine will become infected for further propagation.

W32/Blaster

On August 11, 2003, a worm, named W32/Blaster, was launched [10]. It affects computers running Microsoft Windows NT 4.0, Microsoft Windows 2000, Microsoft Windows XP, and Microsoft Windows Server 2003. This worm exploits a vulnerability in the Microsoft Remote Procedure Call (RPC) Interface. This vulnerability affects a Distributed Component Object Model (DCOM) interface with RPC, which listens on TCP/IP port 135. This interface handles the DCOM object activation requests that are sent by client machines to the server. Due to incorrect handling of malformed messages exchanged over TCP/IP, an attacker can use buffer overflow to execute arbitrary code with System privileges or cause denial of service.

Upon successful execution, the worm attempts to retrieve a copy of the file “msblast.exe” from the compromising host. Once this file is retrieved, the compromised system runs it and begins scanning for other vulnerable systems to compromise in the same manner. In the course of propagation, the TCP session to port 135 is used to execute the attack. The worm also has the ability to launch a TCP SYN flood DoS attack against “windowsupdate.com”.

ATTACKS AGAINST ENTERPRISE NETWORK SYSTEMS

An enterprise network system refers to all hardware and software that construct the network infrastructure of a company to support its business. Generally, the network is composed of an internal network and some connections to the Internet. It can permit flexible communications among a wide range of individuals and enterprises, thereby enabling more convenient and efficient business operations. However, it can also expose the enterprise systems to a much wider variety of attacks than they previously faced [19]. If sensitive information is disclosed or improperly modified, or if critical services are denied to the enterprise or its customers, the security of the enterprise network system will be breached. In this section, we give an overview of the possible vulnerabilities of enterprise network systems and the corresponding security mechanisms.

Attacks against Private Networks

In this scenario, the network is mainly a LAN inside a company, which connects and supports communication among all divisions of the company, e.g., headquarter, sale, R&D, factory, etc. Business information from outside goes through leased telephone, post mail, or fax, and is entered (by human operators) into the system. Office operations and word processing are handled

with in-house servers and clients (PCs) that are connected by an in-house LAN. There are no external network connections to the LAN.

This type of network is typically hacked from inside. An authorized user or administrator can incorrectly configure authorized software, which results in misdelivery of messages. A user can load arbitrary software into a computer through a diskette and introduce malicious codes to the enterprise system. Users can attach their own modems to their PCs and open a backdoor to the private LAN through the leased telephone lines. It's also possible that an intruder gains physical access to the private network lines and eavesdrops on parts of the private network.

Attacks against Private Networks with Web Service

In this scenario, a web site is added to the private network as an interface for customers to place orders through the Internet. On the site, customers can review information about product specifications, new product announcements, company telephone numbers, and so on. In addition, the web site can also provide HTML forms to take orders. This entails implementation of interactive scripts on the existing web server, which can provide a path for the orders to be placed into the private network.

The web server, as long as it provides information, is vulnerable to denial of service attacks that originate in the Internet. If there is a bug in the web server that can be invoked by Internet users, such as a buffer overflow bug, the web server may be taken by attackers. If the web server is hosted by an ISP, attackers can penetrate into the ISP's system to take control of the server and modify the web content. Some vulnerabilities come from non technical operations in the WWW environment. For example, competitors or vandals could set up web sites with similar names and place bogus information on them.

Unless specific measures are implemented to authenticate customers and protect their communications with the server, the order information that the server sends to the order-taking staff may be forged, and outsiders may be able to forge order information that seems to come from the web site. Valid, but unsafe scripts can permit an outsider to "take over" a web site and thereby alter the information it provides, generate false requests, close it down, or initiate other malicious actions.

Attacks against Firewalls and Virtual Private Networks

In this scenario, to ensure security, firewalls are added to protect the internal network. However, this is not absolutely safe. If the firewall is not carefully configured, it may provide a false sense of security, and permit outsiders to hack internal systems. An inadequately configured firewall can make internal hosts visible to the outside world, may pass traffic from untrusted hosts and ports that are supposed to be blocked, and may provide an incorrect proxy server that lets malicious traffic into the internal network. Insiders can invoke malicious software to leak information or import malicious codes. Administrators and users may install tools on systems so that they can work remotely and conveniently. These tools may become backdoors for outside intruders.

A company may have several private networks allocated across separated locations. A Virtual Private Network (VPN) consists of a set of corporate sites and internal networks. Each site manages its own Internet connection and runs an encrypting firewall so that traffic between any pair of sites is encrypted, and the Internet works as a transmission medium. The network is virtually private in that only the packet headers containing routing information are exposed to

public view; however, it is still not private in the sense that it contains no leased private lines. Although communication over the Internet can be flexible, the Internet does not guarantee quality and security. Depending on the ISP and the Internet backbone, delays may occur in delivering important messages or cause serious degradation of performance. Furthermore, more internal corporate information is flowing over the Internet and open to interception, although they are encrypted. Without careful configuration and usage of cryptographic techniques, such as weak encryption and short keys, critical information may still be decrypted by a malicious third party.

CONCLUSION

In this article, we discussed a variety of hacking techniques. From the functionalities, objectives, and principles of different hacking techniques, we can summarize that vulnerabilities of a network or system always come from two major factors, technical factor and human factor. The technical factor refers to those imperfect designs of networks and systems, such as unencrypted data, unprotected communications, buffer overflow problems and software bugs. These deficiencies provide holes through which intruders can penetrate into the system. The human factor is another important perspective. For example, users' incautious talk can become the source to disclose critical information about network and system. Inappropriate use of the system may let attackers sneak in. Insiders may be the most serious threats to the system.

Glossary

Autonomous Systems (AS) A collection of routers under a single administrative authority, using a common Interior Gateway Protocol for routing packets.

Border Gate Protocol (BGP) A protocol that distributes routing information to the routers, which connect autonomous systems.

Domain name A series of alphanumeric strings separated by periods that is used to name organizations and computers and addresses on the Internet.

Domain Name System (DNS) A general-purpose distributed, replicated, data query service chiefly used on Internet for translating hostnames into Internet addresses.

Firewall A router or computer software that prevents unauthorized access to private data (as on a company's local area network or intranet) by outside computer users (as of the Internet).

Hypertext Transfer Protocol (HTTP) A protocol used to request and transmit files, especially webpages and webpage components, over the Internet or other computer network.

Internet Control Message Protocol (ICMP) One of the Internet protocols that allows for the generation of error messages, test packets, and informational messages related to IP.

Internet Protocol (IP) A connectionless, best-effort packet switching protocol that provides packet routing, fragmentation and re-assembly through the data link layer.

Internet Service Provider (ISP) A company that provides other companies or individuals with access to, or presence on, the Internet.

Listening post A center for monitoring electronic communications (as of an enemy).

Plaintext The unencrypted form of an encrypted message.

Private network A network composed of point-to-point leased lines between sites.

Router A device that forwards packets between networks based on network layer information and routing tables, which often constructed by routing protocols.

Routing Information Protocol (RIP) A distance vector routing protocol that distributes routing information to the routers within an autonomous system.

Transmission Control Protocol (TCP) A protocol for the internet to get data from one network device to another by using a retransmission strategy to insure that data will not be lost in transmission.

Uniform Resource Locator (URL) A way of specifying the location of an object, typically a web page, on the Internet. It has two parts separated by a colon. The part before the first colon specifies the protocol. The part after the colon is the pathname of a file on the server.

User Datagram Protocol (UDP) A connectionless protocol in the transport layer layered on top of IP protocol that provides simple but unreliable datagram services.

Virtual Private Network (VPN) A network composed of several sub private networks connected through a public network (as of the Internet). The network traffic is encrypted in the IP layer so that secure connections among the sub private networks are provided through the insecure public network.

Cross References

References

[1] Bellovin, S. M. (1989). Security problems in the TCP/IP protocol suite, *ACM SIGCOMM Computer Communication Review*, vol. 19, pp. 32-48.

[2] Bellovin, S. (1995). Using the Domain Name System for System Break-ins, *Proceeding of the 5th UNIX Security Symposium*, pp.199-208.

[3] Boulanger, A. (1998). Catapults and grappling hooks: The tools and techniques of Information warfare, *IBM System Journal*, vol. 37, no 1, pp. 106-114.

[4] CERT® Advisory CA-1998-01 Smurf IP Denial-of-Service Attacks (2000). Retrieved May 20, 2004, from http://www.cert.org/advisories/CA-1998-01.html.

[5] CERT® Advisory CA-1999-04 Melissa Macro Virus (1999). Retrieved May 20, 2004, from http://www.cert.org/advisories/CA-1999-04.html.

[6] CERT® Advisory CA-2001-19 “Code Red” Worm Exploiting Buffer Overflow In IIS Indexing Service DLL (2002). Retrieved May 20, 2004, from http://www.cert.org/advisories/CA-2001-19.html.

[7] CERT® Advisory CA-1999-16 Buffer Overflow in Sun Solstice AdminSuite Daemon sadmind (2000). Retrieved May 20, 2004, from http://www.cert.org/advisories/CA-1999-16.html.

[8] CERT® Advisory CA-2001-26 Nimda Worm (2001). Retrieved May 20, 2004, from http://www.cert.org/advisories/CA-2001-26.html.

[9] CERT® Advisory CA-2003-04 MS-SQL Server Worm (2003). Retrieved May 20, 2004, from http://www.cert.org/advisories/CA-2003-04.html.

[10] CERT® Advisory CA-2003-20 W32/Blaster worm (2003). Retrieved May 20, 2004, from http://www.cert.org/advisories/CA-2003-20.html.

[11] CERT/CC Statistics 1988-2003 (2004). Retrieved May 20, 2004, from http://www.cert.org/stats/cert_stats.html.

[12] Computer Oracle and Password System (1993). Retrieved May 20, 2004, from http://www.fish.com/cops/.

[13] Cowan, C., Wagle, F., Pu, C., Beattie, S., and Walpole, J. (2000). Buffer overflows: attacks and defenses for the vulnerability of the decade, *Proceeding of DARPA Information Survivability Conference and Exposition*, pp. 119-129.

[14] Denial of Service Attacks (2001). Retrieved May 20, 2004, from http://www.cert.org/tech_tips/denial_of_service.html.

[15] DNS Attack Scenario (1996). Retrieved May 20, 2004, from http://www.cs.princeton.edu/sip/news/dns-scenario.html.

[16] ETHEREAL (2004). Retrieved May 20, 2004, from http://www.ethereal.com/.

[17] John the Ripper password cracker (2004). Retrieved May 20, 2004, from http://www.openwall.com/john/.

[18] Joncheray, L. (1995). Simple Active Attack Against TCP, *Proceedings of the 5th USENIX UNIX Security Symposium*.

[19] Landwehr, C. E. and Goldschlag, D. M. (1997). Security issues in networks with Internet access, *Proceedings of the IEEE*, vol. 85, pp. 2034-2051.

[20] Murphy, S. (2003). BGP Security Vulnerabilities Analysis, Retrieved May 20, 2004, from http://www.ietf.org/internet-drafts/draft-ietf-idr-bgp-vuln-00.txt.

[21] NESSUS (2004). Retrieved May 20, 2004, from http://www.nessus.org.

[22] NMAP (2004). Retrieved May 20, 2004, from http://www.insecure.org/nmap/.

[23] Password cracker (2004). Retrieved May 20, 2004, from http://www.pwcrack.com/index.shtml.

[24] Security Administrator Tool for Analyzing Networks (1995). Retrieved May 20, 2004, from http://www.fish.com/satan/.

[25] Seeley, D. (1990). The Internet Worm of 1988, Retrieved May 20, 2004, from http://world.std.com/~franl/worm.html.

[26] SPYTECH (2004). Retrieved May 20, 2004, from http://www.spytech-web.com/.

[27] TIGER (1994). Retrieved May 20, 2004, from http://savannah.nongnu.org/projects/tiger/.