

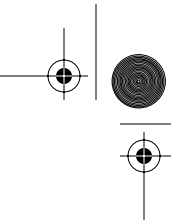
Internet Penetration 5

This chapter begins our discussion of the general process for performing penetration testing that we have developed during our experience. While the procedures discussed are not set in stone and we never cease to examine and refine our own techniques, we would like to stress that the approach laid out is both an efficient means of compromising a network and an effective means of evaluating the security posture of that network.

That is not to say it is the only means of examining the security posture of a network. Other security professionals have different and valid testing techniques. This process is one that has proven to be effective.

Having a defined, organized methodology provides for an efficient penetration test with a consistent level of detail. Professional consultants hired to perform penetration testing attempt to compromise the target network during a given time period, often a matter of weeks or even days. This is substantially different than hackers who can spend as much time as they want in attempting to gain root access to a network. Therefore, we need a well-defined methodology that allows us to systematically check for known vulnerabilities and pursue potential security holes in the time allotted. In addition, following a single methodology helps ensure a consistent level of reliability in results across multiple engagements.





The overall methodology for penetration testing can be broken into a three-step process.

1. *Network enumeration*: Discover as much as possible about the target.
2. *Vulnerability analysis*: Identify all potential avenues of attack.
3. *Exploitation*: Attempt to compromise the network by leveraging the results of the vulnerability analysis and following as many avenues identified as time allows.

Throughout our discussion of this process, we reference the tools we have found most useful for accomplishing these tasks.

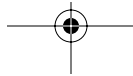
5.1 NETWORK ENUMERATION/DISCOVERY

Before we can gain unauthorized access to a network, we have to know the topology of the network. Every piece of information we can obtain about the target network adds a piece to the puzzle. We specifically scan the target network to obtain a list of live hosts, as well as to begin mapping the target to get a sense of its architecture and the kind of traffic (for example, TCP, UDP, IPX) that is allowed. The goal of discovery is to start with no information and gather as much data as possible about the target network and systems. We then use this information to identify potential exploits.

The process of discovering this information is called *network enumeration* and is the first step to an external penetration test. This step is performed largely over the Internet using readily available software and publicly accessible repositories of information. Most of the information we obtain in this step is freely available and legal to obtain. However, many companies monitor who tries to get this information since it may indicate a prelude to an attack.

5.1.1 WHOIS QUERY

Even before we begin the network scanning, we must determine the domain names and IP address ranges that belong to the target organization. To simulate the scenario of an external hacker, no prior information about the target organi-



zation should be provided to the consultant to best determine the amount of information a hacker could obtain. However, before moving to the second step of the process, all identified domain names and IP addresses should be verified with the target organization to ensure they are owned by the organization and are part of the scope of the exercise.

To determine the IP address ranges associated with the client, we perform an Internet whois query. The command can be run natively on most UNIX environments (check `man whois` for usage and version-specific syntax). For the Windows environment, Ws PingPro Pack and Sam Spade are two tools that can be used to perform whois queries. (These tools are discussed in Chapter 12.)

Whois queries can also be made over the Web from www.arin.net and www.networksolutions.com. Figure 5–1 shows the whois query from the Network Solutions site (without the domain servers) for the domain `klevinsky.com`.

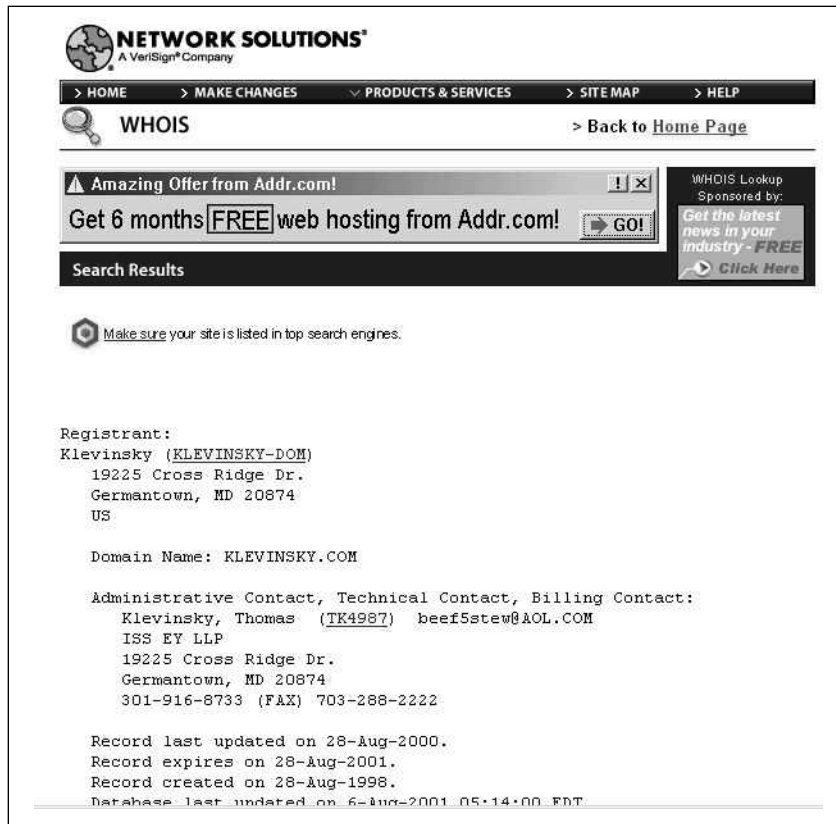
A whois query provides the administrative contact, billing contact, and address of the target network. The administrative and billing contact information can be useful for performing social engineering attacks on the employees of the target network (see Chapter 8).

The whois query provides IP address ranges that are associated with the name you enter. Some ranges may be returned that belong to a separate organization with a similar name. For example, the partial results of a whois query on *company* reveal registered IP addresses for a collection of firms whose names include the word *company* but may not be the target organization.

Of the multiple IP ranges that do belong to the client, a portion may belong to different divisions of the client's organization and lie outside the scope of the engagement. The targets for the engagement should be verified when this information is found.

Whois queries return only the first 50 items that match the query. This is implemented by Internic to limit the search time. As the listings of Internet domains grow, the task of searching all listings and returning all possible matches becomes more computationally intensive.

HACK I.T.



NETWORK SOLUTIONS[®]
A VeriSign[®] Company

> HOME > MAKE CHANGES > PRODUCTS & SERVICES > SITE MAP > HELP

WHOIS > Back to [Home Page](#)

Amazing Offer from Addr.com!
Get 6 months **FREE** web hosting from Addr.com! [GO!](#)

WHOIS Lookup Sponsored by:
Get the latest news in your industry - FREE [Click Here](#)

Search Results

Make sure your site is listed in top search engines.

Registrant:
Klevinsky (KLEVINSKY-DOM)
19225 Cross Ridge Dr.
Germantown, MD 20874
US

Domain Name: KLEVINSKY.COM

Administrative Contact, Technical Contact, Billing Contact:
Klevinsky, Thomas (TK4987) beef5stew@AOL.COM
ISS EY LLP
19225 Cross Ridge Dr.
Germantown, MD 20874
301-916-8733 (FAX) 703-288-2222

Record last updated on 28-Aug-2000.
Record expires on 28-Aug-2001.
Record created on 28-Aug-1998.
Database last updated on 6-Aug-2001 05:14:00 EDT

Figure 5-1 Whois query for klevinsky.com

If the target company has more than 50 listings that interest you, you may have to engage in some creative searching. One idea is to break up the names of the company or search for plurals or modified company names. Find the names of subsidiary organizations (press releases on the target company's Web site are a good place to look) and search for those names as well.

5.1.2 ZONE TRANSFER

A whois query also returns the list of domain name servers that provide the target network's host name and IP address mapping. (This information, along with

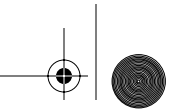
the contact information, is found by clicking on the Net Block name associated with the listing.) To obtain the network IP listing, we want to attempt a zone transfer against each system identified as a DNS server. A zone transfer requests the complete list of matched IP addresses and host names stored within a DNS for a specified domain.

A zone transfer can be performed with the `nslookup` command that is supported by both the UNIX and Windows platforms. Sam Spade, Ws PingPro Pack, and NetScan Tools on the Windows operating system all provide a graphical user interface (GUI) for performing a zone transfer. In order to perform a zone transfer, we have to use a DNS server that is authoritative for the domain of interest; therefore, we use the domain name servers identified through the whois query. Techniques for performing zone transfers are covered in Chapter 12.

The zone transfer returns a listing of IP addresses and their corresponding host names. A typical listing may look something like this:

```
ls -d abc.com
[server.abc.com]
abc.com.                SOA      server.abc.com
                        (200000068 300 800 359100 4700)
admin.abc.com.          A        10.10.10.30
abc.com.                NS       server.abc.com
abc.com.                MX       10 mail.abc.com
business                A        10.10.10.11
application             A        10.10.10.32
mailsweeper             A        10.10.10.50
mimesweeper            CNAME    server4.abc.com
server4                 A        10.10.10.40
abc.com.                SOA      server.abc.com
admin.abc.com.          (200000068 300 800 359100 4700)
```

Machine host names often indicate the function of the machine. For instance, the corporate firewall machine is often called “firewall” or the name of the firewall running, such as “Gauntlet” or “Firewall1.” Similarly, we have seen some equally revealing machine names, such as “mail.companyname.com,” “smtp.companyname.com,” “ftp.companyname.com,” “dns01.companyname.com,” “ns01.companyname.com,” and “web03.companyname.com.”



HACK I.T.

These names not only offer strong evidence of their main function but also indicate the presence of other machines. For example, if there is a web03 machine on a particular network, there stands to reason that a web01 and a web02 may also exist. If there is an ns01 machine, there may also be ns and ns02 machines. In light of this, names of sports teams, famous people, and cartoon characters have been used as good machine names. They are easy to remember, and they do not give away any technical information.

When doing a zone transfer, keep in mind that often the DNS server does not have a complete listing for all the target network's hosts. Several machines may be using DHCP, and the company may use separate domain name servers for separate domains. Also, its DNS may not support zone transfer requests from unauthorized hosts, allowing them only from the backup name servers within the organization. Therefore, you should attempt zone transfers against all the target network's identified domain name servers. One may offer at least a partial listing.

We have also seen companies outsource the domain name function or use their ISP's DNS server. In our experience, performing a zone transfer against a DNS server or any machine belonging to an ISP or a third party is generally not received well by those third parties. In that case, we usually omit this step unless we have the written consent of both the target organization and the third party. In these situations, make sure the terms of the penetration test clearly state whether or not the hosted systems are within the scope of the engagement.

On the other hand, DNS machines that belong to the client organization but are not a part of the IP address range are specifically within scope and are valid targets of a zone transfer as long as there is a reasonable chance that that DNS will offer information regarding the within-scope target domain. This is because an Internet-based penetration relies on using information that lies in the public domain or is publicly accessible.

This usually occurs when the target comprises one or more domains within a large organization. The main DNS server for the organization will likely have a partial listing of the hosts in the target domain even if it lies outside that domain.



Unlike the whois query, a zone transfer is fairly indicative of hacker activity since there really is no need for the general user to have this information. Therefore, someone making this query against a DNS server is probably a potential attacker. For that reason, we suggest exercising good judgment before performing these queries. Zone transfers may indicate to the network staff the beginning of a penetration test against the network.

5.1.3 PING SWEEPS

Our next step is to ping the discovered IP addresses to see if they are “up” or “live.” There are a variety of ways to ping a set of IP addresses. The most commonly used is the traditional ICMP ping (with echo requests or echo replies messages), but gaining popularity is a TCP ping (with a full or half TCP handshake). Many sites have taken the security step of restricting ICMP traffic or blocking it at the border firewall and router, limiting their exposure to the traditional ping. However, a TCP ping may still be allowed on the network.

Over time, organizations have become more adept at blocking a ping sweep, and countermeasures are becoming more prevalent. While you can assume with some amount of confidence that a host that sends an ICMP response to an ICMP echo request is active, it is not always true that a host that fails to send such a response is necessarily down. The host may be down, or ICMP traffic to that host may be filtered and the ping request simply did not reach it. False responses can also be sent to ICMP echo requests by perimeter security devices.

Depending on the level of stealth you are seeking in your pinging activity, there are a variety of steps you can take to remain beneath the radar of an intrusion detection system that may be monitoring network traffic. While these steps are discussed in greater detail in the section on Nmap in Chapter 12, it is worth mentioning that randomizing the order of the IP addresses being pinged helps avoid detection, as do varying the time between sending ping packets and dividing the IP addresses into multiple groups (this is most helpful for large numbers of hosts, that is, over 100).

The ping utility exists natively on most operating systems and can be performed from a large collection of tools. One of the most popular is Nmap because of its

HACK I.T.

configuration, its ease of use, and the other features it includes (TCP ping, port scanning, OS identification). For the Windows environments, Pinger and Ws PingPro Pack are both effective tools for performing ping sweeps. (In addition, a Windows-compatible version on Nmap is currently under development.) Pinger strictly pings a set of IP addresses while Ws PingPro Pack provides additional functionality through a suite of tools.

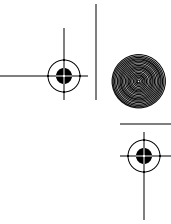
Ping sweeps are generally not considered to be evidence of harmful intent to hack a system. However, they can be irritating or destructive if they become excessive; for example, ping each box on a Class C network every 30 seconds for 8 hours and see how that affects bandwidth.

5.1.4 TRACEROUTE

In order to come up with a rough map of the client architecture, we trace the route to several of the live hosts. This is a tedious process, but it does help identify the routers, firewalls, load-balancing devices, and other border machines in place on the target network. In addition, it helps identify hosts that are on separate segments. Hosts on separate segments may be managed by different individuals and may have trust relationships that can be exploited to compromise the system.

A traceroute marks the path of ICMP packets from the local host (where the command is executed) to the destination host. It is available as a command line tool on both the UNIX (`traceroute`) and Windows (`tracert`) operating systems. In addition, the Windows-based tool VisualRoute performs this service as well as mapping the path over a map of the world. (VisualRoute is discussed in Chapter 12.)

We perform traceroutes on several IP addresses within the same Class C address block to see if the ICMP packets follow the same path. We are interested in seeing the hops just prior to the target. These hops may represent routers, firewalls, or other gateways. If several hosts have the same prior hop, it is probably a router or firewall. If there is a common host after which ICMP packets can no longer be seen, that too may be the firewall or filtering router. Also, a common



host in front of a bank of Web servers may be a load-balancing device or a Web redirector.

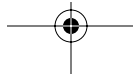
If you notice that packets to some hosts on the network segment follow an alternate path, you may have discovered new gateways into the target network. It is not uncommon for network segments to have multiple connections to the Internet—unknown to network managers. These can be developed on the fly for particular network tests or applications and simply forgotten. Such paths often lead to network compromises.

5.2 VULNERABILITY ANALYSIS

Vulnerability analysis, sometimes called vulnerability scanning, is the act of determining which security holes and vulnerabilities may be applicable to the target network. In order to do this, we examine identified machines within the target network to identify all open ports and the operating systems and applications the hosts are running (including version number, patch level, and service pack). In addition, we compare this information with several Internet vulnerability databases to ascertain what current vulnerabilities and exploits may be applicable to the target network.

Given the time constraint we may be under during an engagement and the number of hosts within scope, it may be necessary to focus initially on critical hosts. However, if any paring down of the target list needs to be done, it is usually done during the next step.

Note: It is important to take into consideration that the ping results do not authoritatively show that a host is down. In light of this, if there is any doubt as to whether the target(s) are effectively filtered or protected from ping or are actually down, we recommend continuing with a port scan. Keep the number of ports in such scans down as these scans tend to take a longer amount of time. If it is necessary to scan a large number of ports on unresponsive hosts, it is best to do this overnight.



HACK I.T.

At the end of this stage, we like to be able to document all target hosts (alive and otherwise) in a table along with the OS, IP address, running applications, any banner information available, and known vulnerabilities. This information is useful both during the exploitation stage and for presentation to the client so that the client becomes aware of the vulnerabilities on the network and the amount of information an outsider can gather prior to compromising the network.

5.2.1 OS IDENTIFICATION

By identifying the operating system, we can attempt to predict services that may be running on the host and tailor our port scans based on this information. Nmap, the leading tool used to perform OS identification, does this by analyzing the response of the target's TCP stack to the packets Nmap sends out. Various RFCs govern how the TCP stack should respond when queried. However, implementation details are left to the vendor. Therefore, differences in how vendors satisfy the RFCs allow them to be identified. While this method is not foolproof, Nmap's OS detection is fairly reliable and well accepted by industry. Changing a computer's OS signature is possible but not trivial, and it has not been our experience that companies perform this level of masking.

OS identification goes a long way in performing network enumeration and vulnerability scanning. As soon as we know the OS of a particular machine, we can begin to generate a list of potential holes and vulnerabilities—often from the vendor's own Web site. For example, as soon as we know a machine is Windows NT, we can check whether TCP port 139 is open and attempt a null connection to the IPC\$ share. If we identify a UNIX box, we can look for the X Windows ports (6000–6063).

5.2.2 PORT SCANNING

Port scans attempt to determine whether a listening service is listening on a given port. There are many variations in performing a port scan. We describe those that have been the most useful for us. Depending on the level of stealth you want to maintain, there are a variety of scan types you can use, TCP SYN scan is

the most popular of the stealthy port scans and is covered in more depth under the Nmap section of Chapter 13.

A scan of all possible ports (1–65535) is the most comprehensive and offers the most information on the target, but it also is the most time consuming and maximizes the chances of being identified by the target. Such a port scan is usually performed only by beginner hackers. If you do elect to scan all ports, we strongly recommend that the scan be performed over several sessions, each with a smaller port range. Often we perform a comprehensive scan at the end of the engagement when stealth is no longer necessary. This helps identify any services that we may have missed during the surgical port scans.

If you are not attempting to avoid detection and are just trying to identify weaknesses in the target system (for example, if your client's security staff is fully aware of your penetration testing efforts), then there is no problem with scanning all ports at once. However, it will take a long time to return information. The test becomes more efficient if you can review results while simultaneously scanning new systems.

Luckily, there are several alternatives to a full port scan. We can stick to the basic known ports, 1–1024, and add a few other ports we know to be relevant to the client, such as the X Windows ports (6000–6063) on a UNIX machine. Reviewing the `/etc/services` file on a UNIX machine also provides a good listing of ports to scan. We can also create a list of ports that support applications with known vulnerabilities we may wish to exploit, such as FTP, telnet, and RealSecure (ports 21, 23, and 2998, respectively). Most scanners give you the ability to scan both TCP and UDP ports. Often, UDP ports are simply ignored since they are less common, but they can be just as vulnerable. Since UDP is a connectionless protocol, UDP port scans are generally considered less reliable.

Ultimately, you should develop a port list that you are comfortable using on any given network and modify the list to fit the particular network you are going to scan. Specifically, create a generic list, and remove NT-specific ports for a UNIX network or vice versa. Nmap is distributed with a port list of several known ports and can serve as a starting point for such a generic list. (Additional port lists are also available on various hacker sites; see Chapter 22.) To this list, add ports

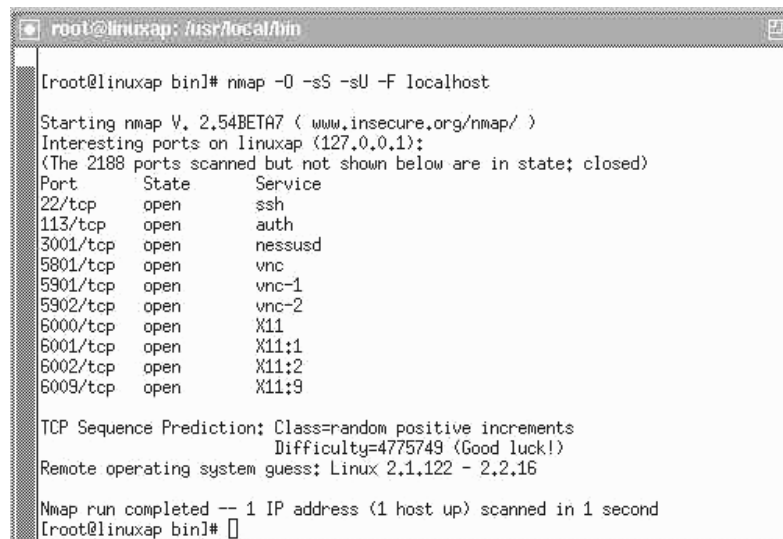
HACK I.T.

each time you find another one that is associated with an application featuring a vulnerability you are aware of or a security hole you can exploit. Remove ports that are not related to weaknesses, vulnerabilities, or information gathering. While maintaining such a list demands continuous testing, the more port scanning you do, the more relevant the information your own list will return.

As previously mentioned, on UNIX environments, Nmap is the tool of our choice for port scanning and is considered to be the premier port scanner available (as well as a reliable OS detection tool). SuperScan and 7th Sphere are effective port scanners for the NT environment but do not include OS detection. (As mentioned, an Nmap for the Windows environment is under development.)

The use of these tools is discussed in depth in Chapter 13, so we will avoid repetition here. However, Figure 5–2 displays Nmap results for a scan of TCP and UDP ports on a single Linux host.

Port scanners generally ping hosts before scanning and only scan those hosts that respond to pings. If there is any doubt as to the validity of the ping results, then



```
root@linuxap: /usr/local/bin
[root@linuxap bin]# nmap -O -sS -sU -F localhost
Starting nmap V. 2.54BETA7 ( www.insecure.org/nmap/ )
Interesting ports on linuxap (127.0.0.1):
(The 2188 ports scanned but not shown below are in state: closed)
Port      State      Service
22/tcp    open       ssh
113/tcp    open       auth
3001/tcp   open       nessusd
5801/tcp   open       vnc
5901/tcp   open       vnc-1
5902/tcp   open       vnc-2
6000/tcp   open       X11
6001/tcp   open       X11:1
6002/tcp   open       X11:2
6009/tcp   open       X11:9

TCP Sequence Prediction: Class=random positive increments
                        Difficulty=4775749 (Good luck!)
Remote operating system guess: Linux 2.1.122 - 2.2.16

Nmap run completed -- 1 IP address (1 host up) scanned in 1 second
[root@linuxap bin]#
```

Figure 5–2 Sample Nmap results

set the port scanners to scan hosts unresponsive to ping. As a consequence, the port scanning will take more time.

The legality of port scanning has long been a topic of discussion in the security community. Some professionals have indicated port scanning is no more than driving down a street looking at houses and noticing which windows are open. However, port scanning without permission is clearly unethical and will always be alarming as a possible prelude to an attack.

5.2.3 APPLICATION ENUMERATION

From the results of port scanning, we gain a list of open ports on the target machines. An open port does not entirely indicate what listening service may be active. Ports below 1024 have been assigned to various services and if these are found open, they generally indicate the assigned service. Additionally, other applications have been run on certain ports for so long that they have become the de facto standard, such as port 65301 for pcAnywhere and 26000 for Quake. Of course system administrators can change the port a service runs on in an attempt to “hide it” (an example of security through obscurity). Therefore, we attempt to connect to the open port and grab a banner to verify the service running.

Knowing which applications the target hosts are running goes a long way toward performing vulnerability analysis. Just as with knowing the OS, we can run the list of applications through the Internet and find a list of known vulnerabilities and exploits for these applications—again, often from the vendors themselves.

Application enumeration also involves banner grabbing. Several applications (including telnet, FTP, HTTP, SNMP, and a host of others) identify themselves and their version in their user name/password challenge screen. This information is called a *banner* and is very helpful in identifying running applications. We generally record any banners we come across during our penetration testing. This can be done with many applications, including netcat, which runs on either the UNIX or Windows command line; telnet; and What’s Running, a Windows GUI tool that is covered in Chapter 12 and shown in Figure 5–3.

HACK I.T.

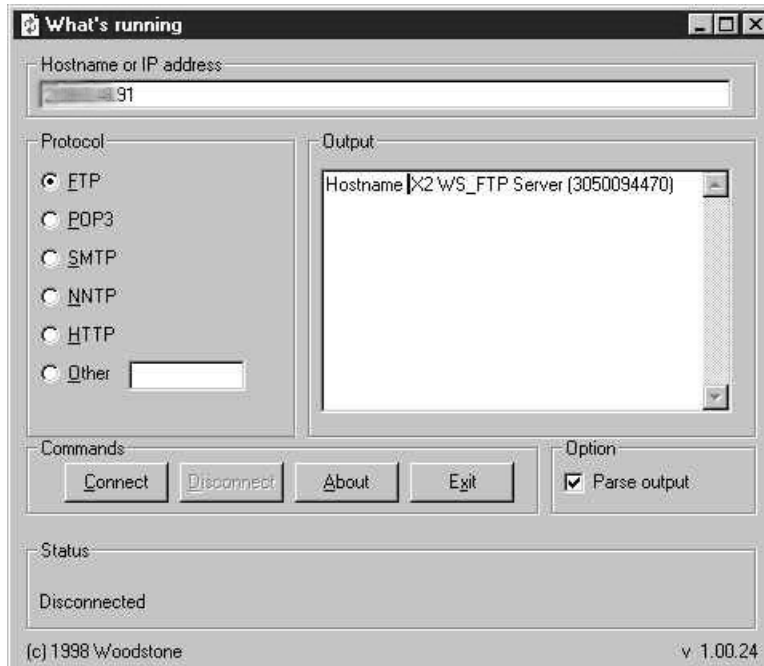


Figure 5-3 Screen shot of What's Running

A benefit of What's Running is that the banner is placed in a window from which it can be copied into a file or edited.

5.2.4 INTERNET RESEARCH

Once the list of applications is known, the next step is to research the list and determine what vulnerabilities exist. As you perform penetration tests, you become familiar with certain popular vulnerabilities and can quickly determine whether an application is vulnerable. However, it is important to keep in mind that new vulnerabilities are posted on a daily basis, and you should check your favorite vulnerability databases for all the applications, services, and operating systems you find on each engagement.

Chapter 22 contains a list of Internet sites with vulnerability databases. The popular ones are the Bugtraq lists, Packetstorm (www.packetstormsecurity.org), and SecurityFocus (www.securityfocus.com). You will become comfortable with these and other sites that offer the ability to quickly search for vulnerabilities. Even though there is some amount of duplication among them, you may want to check multiple sites since no one repository is entirely comprehensive. Each site has its own network of users and supporters who update the databases.

Once we have identified vulnerabilities and documented them in our table, we download the exploit code (if applicable) to use in the next phase of the penetration test.

5.3 EXPLOITATION

Once the list of vulnerabilities has been identified, the next step is to proceed to exploit the vulnerabilities in an attempt to gain root or admin-level access to the target systems.

Our general procedure is determined in part by the results of our enumeration and information gathering. We examine the list of known vulnerabilities and potential security holes on the various target hosts and determine which are most likely to be fruitful. Next we pursue exploiting those vulnerabilities to gain root access on the target system.

Primary targets are open ports and potentially vulnerable applications. Our approach is to review the list of vulnerabilities collected in the previous stage and sort them by likelihood of success and potential harm to the target network to see which may be helpful in our exploitation efforts. For instance, a buffer overflow or denial-of-service attack may well be successful on the target but also dangerous to the system. Unless the contract specifically calls for DoS attacks to be performed as part of the test, they should not be attempted.

Among the common exploits performed are Web-server hacks. This is fast becoming a popular way to compromise target networks. One popular Web-based hack is the Microsoft IIS MDAC/RDS hack. This is an exploitation of the IIS

HACK I.T.

Web server through the `msadcs.dll` file and the Remote Data Service (RDS). It allows the attacker to execute a single command on the target host. The command can be reused to execute a succession of individual commands that when taken together can be used to achieve a variety of results, such as retrieving sensitive files on the target host and making connections to other hosts. Also, when used in conjunction with other tools, such as the `ntuser` command, it can allow a user to be placed into the local administrator's group.

A Perl script, `msdacExploit.pl`, coded by rain forest puppy, can remotely exploit this vulnerability and is widely available. (The `msdacExploit.pl` file is not the only file coded to exploit this hole.) In order to perform this exploit, simply run the following command against the target host.

```
C:\> perl -x msdacExploit.pl -h <target host>
```

(You do not necessarily have to be in the C drive.) A command prompt from the target host should appear on your machine and allow you to execute one command. To run multiple commands, the exploit must be run multiple times.

Once we have obtained unauthorized access to a remote system through either the ability to execute a command on a target host or direct access to an actual user account, we immediately document all relevant information, including the host and directory or share name to which we have gained access, the host from which we gained access, the date and time, and the level of access. Also, we specify the hole(s) that we exploited to gain access. Next, we share this information with the target organization. This serves two purposes: (1) to alert the organization to the hole(s) we have identified and exploited so that the company can begin to address the issue and (2) to cover ourselves as penetration testers from a litigation standpoint. Even in the case of an unannounced test, our point of contact (who is aware of our activities) should know when we have gained access so if we are detected the matter is not escalated to law enforcement authorities.

Having gained access to one machine is not necessarily the end of our penetration test. If additional work is within scope, we can continue by installing a tool kit comprised of the tools we can use to test other systems from the exploited box. This is different from the “root kit” used within the hacker community to represent a collection of tools and exploits used to either compromise the same system again in the future, by creating back doors or Trojaning system files, or to launch attacks against other hosts, such as distributed denial-of-service daemons.

This tool kit is tailored to the operating system of the target machine and of the machines we may encounter during the penetration test. Generally, we include netcat, password crackers, remote control software, sniffers, and discovery tools. Often, due to the connection, command line tools are preferred. GUI tools can be used if a remote control program such as pcAnywhere or Virtual Network Computing (VNC) is first installed. Otherwise, having the target send a GUI back to our box can be tricky in that it may still be blocked at the firewall or by the host itself. Additionally, it can sometimes display the GUI on the local machine, alerting the machine’s user of our presence and activities.

The tool kit can be copied over with FTP or TFTP, but other means are possible as well. Once the kit is installed, we can begin penetration testing other machines. At this point, the methodology we use closely follows the internal testing method since we are essentially located on the target network. Refer to Chapter 7 for information on how to proceed with testing of additional systems.

Some of the things we do include are sniffers and keystroke capture utilities through which we can capture client traffic. We are looking specifically for user names and passwords that can be used to attempt access on other hosts, dial-in systems, or listening services on the network. Sniffers are discussed further in Chapter 14.

We also try remote control tools that allow us to control the system. There is tremendous potential for further network compromise once we have taken over one machine. We may capture the UNIX password file (along with the shadow password file) or the Windows registry (often through the version stored in the repair directory) to obtain passwords for all users on the machine and possibly the ad-

HACK I.T.

min account(s), which likely give us access to additional machines on the network. Remote control tools and usage are discussed further in Chapter 18.

In any case, we load whatever tools will help us to use the compromised system as a platform for exploiting additional systems. However, as we load these tools, we keep careful track of what was loaded and where so we can return the system to normal after testing.

CASE STUDY**DUAL-HOMED HOSTS**

As mentioned in Chapter 4, dual-homed hosts introduce a significant security hole into the network architecture since they can give users with access rights and privileges on one network or domain the rights and privileges they perhaps are not intended to have on a separate domain. This vulnerability usually appears as a corporate desktop machine connected to the organization's internal LAN and simultaneously connected through a modem line to a local ISP. In such a configuration, anyone on the Internet may be able to access the corporate network through the dial-up connection. However, there are other configurations in which this vulnerability can occur.

For example, on one engagement in particular, the client was an ISP that also provided Web-hosting services for thousands of companies. The hosting facility consisted of a large number (in the hundreds) of UNIX-based hosts, with identical configuration, running the Netscape Web servers.

The ISP's model, in place of providing full management, was to maintain the machines but allow the clients to manage the Web servers themselves.

Included in the Web-hosting package was the tcl scripting language, which allowed remote management of the Web servers. What perhaps was unknown to the ISP is that through the tcl scripting language, knowledgeable clients and even visitors to the hosted Web sites would be able to do more than basic administration. It was possible to use the Web server, which was running with root

privileges, to gain root access on the machines through various specially crafted URL strings. This is an input-validation attack against the Web server.

This led to the compromise of the host machine, in much the same way misconfigured Microsoft IIS servers can lead to the compromise of the host machine. However, this did not turn out to be the worst exposure on the network.

Once a machine on the Web-hosting network was compromised (for example, root access was achieved), a hacker tool kit could be loaded onto that machine, including tools to crack passwords. Once having gained root access on one machine, we were able to determine that the network was connected to a second network used to support various business units of the ISP. Further, we found that some users on the Web-hosting network had accounts on the second network as well and used the same passwords.

At this point, access to this second network was achieved simply by the existence of accounts with the same user name and password on both networks, and the hacker toolkit could again be copied and installed.

We were able to determine that a machine on this second network was also homed on a third network. This third network was the corporate, internal network used to support payroll and accounting functionality and to maintain client databases and other such valuable assets. This network was intended to be a self-standing, internal network. One machine was mistakenly left dual-homed.

This machine was discovered by identifying that it had two NIC cards with IP addresses belonging to two separate address ranges. Therefore, user accounts (and the root account) on this box had rights on both networks. As can be expected, the root account had the same password on all hosts in the second network, and therefore, we gained root access to the organization's core, internal network.

In summary, it was possible to gain root access to a machine on the Web-hosting network using software existing on the Web servers themselves, to jump to a second network through user accounts with the same user name/password pairs, and finally, after discovering a dual-homed box, to gain unauthorized access to the

HACK I.T.

internal corporate network. Actually, given that valid access rights had been attained, this access *was* authorized in the sense that access control mechanisms did not stop it or identify it as being unnecessary.

After the company managers realized they had inadvertently left a machine on their internal, private network dual-homed on a network that had connections to the outside world, and thereby damaged the integrity and confidentiality of the company's critical data assets and client information, they were understandably shocked and mortified.

LESSONS LEARNED

We have seen several cases where organizations were unaware that a dual-homed machine existed or the organization had used a dual-homed host as an easy solution to fix problems with certain applications communicating through firewalls. The moral of the story is that close attention needs to be paid to an organization's network architecture. After designing and implementing a secure architecture, including both host configuration and overall network topology, any changes must go through a change-control mechanism to help prevent security exposures such as the dual-homed scenario from sneaking into the environment.