9 The Enemy

For the past several years, the Honeynet Project has identified common tools, tactics, and motives shared by the blackhat community and has used this information to create a common methodology. Regardless of who you are and what systems you run, your organization is at risk. In this chapter, we discuss this methodology and how these threats apply to your organization. In Chapters 10 and 11, we review specific examples of honeypots compromised in the wild. By understanding the blackhat's methodologies, you will have a better idea of who your enemy is and the threat you face.

The Threat

A threat we all face is what is commonly known as the script kiddie methodology, the probing for and exploiting of the easy kill. The script kiddie methodology represents someone looking for the path of least resistance. The person's motives may be different, but the goal is the same: to gain control the easiest way possible, usually on as many systems as possible. The attacker does this by focusing on a small number of exploits and then searching the Internet for the given vulnerability, sooner or later finding targets.

Some of these blackhats are advanced users who develop their own tools and leave behind sophisticated backdoors. Others have no idea what they are doing

knowing only how to type `setup` at the command prompt. Regardless of their skill level, the blackhats share a common strategy: randomly search for a specific weakness and then exploit it. It is this random selection of targets that makes this strategy such a dangerous threat. Inevitably, your systems and networks will be probed; you cannot hide. We know administrators who were amazed to have their systems scanned when they had been up for only two days and no one knew about them. There is nothing amazing here. Most likely, their systems were scanned by a blackhat who happened to be sweeping that address block.

If this technique were limited to several individual scans, statistics would be in your favor. With millions of systems on the Internet, odds are that no one would find you. However, this is not the case. Most of these tools are easy to use and are widely distributed; anyone can use them. A rapidly growing number of people are obtaining these tools at an alarming rate; think of it as a type of Internet baby boom. Because the Internet knows no geographic bounds, this threat has quickly spread throughout the world. Suddenly, the law of numbers is turning against us. With so many users on the Internet using these tools, it is no longer a question of *whether* you will be probed but *when*. If your system has been connected to the Internet for more than 24 hours, you probably have already been probed.

This is an excellent example of why security through obscurity fails. You may believe that if no one knows about your systems, you are secure. Or, you may believe that your systems are of no value, so no one would probe them. Some organizations take security seriously and have highly secured systems and networks. However, all that needs to happen is a single mistake: a single system not patched, a misconfigured firewall rulebase, an intrusion detection system plugged into the wrong port, or a system that has an unsecured service accidentally started. It is these very systems that the script kiddies are searching for: the unprotected system that is easy to exploit, the easy kill.

THE TACTICS

Over the past several years, the Honeynet Project has consistently seen the same tactics used against the Honeynet. Alhough these tactics do not apply to the entire blackhat community, they are the ones most commonly used. You will

most likely see these tactics used against your organization. The tactic we have identified is a simple one. A majority of blackhats randomly scan the Internet for a specific weakness; when they find it, they exploit it. They focus on a specific vulnerability, perhaps the only one they know. Sometime, they use tools released for mass scanning and scan millions of systems until they find potential targets. Most of the tools are simple to use and automated, requiring little interaction. You launch the tool and come back several days later to obtain your results. The blackhat community even has a name for these types of tools: *autorooter*. No two tools are alike, just as no two exploits are alike. However, most of the tools are based on the same tactics. First, the blackhat develops a database of IP addresses that can be scanned: live systems that the blackhat can probe. The next step is to gain information on those IP addressees: what operating system they are using and any services or applications they are offering. Often, the version of the service or application must be determined. Once this information is obtained, either the blackhat or the tool will determine whether the remote system is vulnerable. Recently, however, it has become more and more common that blackhats do not even bother trying to determine whether the remote system is vulnerable. They just run the exploit against a wide range of systems and see whether they are successful.

For example, let's say that a blackhat has a tool that exploits a vulnerable version of `rpc.statd` on Linux systems, such as `statdx.c`. The blackhat may not know how the tool works or may not even know what `rpc.statd` is. Most likely, someone on IRC explained the exploit, or the blackhat downloaded a HOWTO that explains the tool step-by-step. However, the blachkhat does know that Linux systems running a vulnerable version, such as Red Hat 6.2, must be found. Often, the tools come preconfigured to be run against a specific operating system or vendor type. These are the systems and vulnerabilities the blackhat will look for. First, the attacker would develop a database of IP addresses that could be scanned: systems that are up and reachable. Another method would be to conduct a zone transfer of a domain's DNSs. Once this database of IP addresses is built, the user would want to determine which systems were running Linux. This can be done by looking at systems banners, such as from TELNET, or using more sophisticated scanning tools to determine the remote operating system type, such as Nmap or Queso. These tools create special packets that can remotely determine the operating system type of most systems, sometimes even the kernel

version or the patch level. Once the remote operating system type has been determined, the next step is to determine whether the service is running, in this case, `rpc.statd`. Port scanners, such as Nmap, or simple systems tools, such as rpcinfo, could then be used to determine which Linux hosts were running `rpc.statd`. All that is left now is to exploit those vulnerable systems.

These tactics are not limited to UNIX-based systems; we see the same tactics used against Windows-based systems also. Blackhats will randomly probe the Internet for specific Windows-based vulnerabilities and then, once identified, compromised them. For example, NetBIOS scans are one of the most aggressive scans we have seen. Blackhats on the Internet are aggressively scanning for systems with Windows SMB exposed shares. The Honeynet Project logged more than 500 such scans in a single month (see Appendix D). Other common probes are for NT IIS vulnerabilities, such as Unicode or RDS. Then, the blackhat community will quickly exploit these vulnerable systems. The blackhat community is not biased but will aggressively probe for and find any vulnerability. No system is safe.

Not every blackhat follows these tactics step-by-step. Often, only part of these tactics may be followed. For example, many blackhats become lazy and do not even bother building a database of IP addresses but instead just sequentially scan an entire network for a specific service, such as Washington University's FTP server daemon. If the blackhats find a system running FTP, they will not bother to determine which vendor or which version is running but instead will just launch the exploit. If it works, great. If not, they move on to the next system. They literally have millions of systems to try. As these tools are almost always automated, the numbers are in their favor. The blackhats can run these scans 24 hours a day, 7 days a week, at no cost to themselves.

You would think that all this scanning would be extremely noisy, attracting a great deal of attention. However, many people are not monitoring their systems and do not realize that they are being scanned or that their systems are being used to scan others. Also, many script kiddies quietly look for a single system to exploit. Once they have exploited a system, they use it as a launching pad, boldly scanning the entire Internet without fear of retribution. If their scans are detected, the system administrator, not the blackhat, will be held liable.

Blackhats often archive or share their scan results for use at a later date. For example, a user develops a database of what ports are open on reachable Linux systems in order to exploit the current image map vulnerability. However, let's say that a month from now, a new Linux exploit is identified on a different port. Instead of having to build a new database, which is the most time-consuming part, the user can quickly review the archived database and compromise the vulnerable systems. As an alternative, script kiddies share or even buy databases of vulnerable or compromised systems. (You will see examples of this in Chapter 11.) The script kiddie can then exploit your system without even scanning it. Just because your systems have not been scanned recently does not mean that you are secure.

Once systems have been compromised, the more sophisticated blackhats implement Trojans and backdoors. Backdoors allow easy, unnoticed access to the system. Even if the administrator changes system accounts or passwords, the blackhat still has remote access. System binaries are trojaned so that the blackhat's presence and activity are hidden. This is done by modifying system binaries to hide the blackhat's files, processes, and any other activity. The Trojans make the intruder undetectable, not showing up in any of the logs, systems processes, or file structure. More sophisticated Trojans modify system libraries or even load kernel modules, modifying the running kernel in memory. To automate this process and make it simpler, tools called rootkits have been developed and published. These kits automate the entire process of taking control of a system, including wiping system logs clean to hide the blackhat, replacing system binaries, implementing backdoors, and launching sniffers to capture system accounts and passwords. We have even recorded rootkits securing the compromised system so no other blackhats can find and exploit the same vulnerability. The blackhats build a comfortable and safe home from which to continue their activity.

These attacks are not limited to a certain time of the day. Many administrators search their log entries for probes that happen late at night, believing that this is when blackhats attack. But they attack at any time. Remember, in most cases, it is automated programs, not manual methods, that break into systems. Scans take place 24 hours a day; you have no idea when the probe will happen. These attacks are also launched from throughout the world. Just as the Internet knows no geographical bounds, it knows no time zones. It may be midnight where the blackhat

is but 1 PM in your location. Expect your systems to be scanned and probed anytime, from anywhere.

THE TOOLS

The tools used are complex to develop but extremely simple to use. Developing the tools requires an intimate knowledge of low-level coding, such as assembler, and the inner workings of operating systems and application development. Only a small percentage of the blackhat community has such skills. Developing tools/techniques is not exclusively a blackhat activity; many whitehat or corporate products are abused and used for malicious purposes. However, the tools are often developed or modified so anyone can use them, with little or no knowledge of how they work. The result is a far larger number of individuals having access to extremely powerful tools that are extremely complex to develop but that are simple to use. Most tools are limited to a single purpose with few options, in part because limited functionality is faster and easier to code and to use. Some tools, however, are also starting to increase functionality, so instead of having to run five programs to perform a task, one program can be used.

First come the tools used to build an IP database. These tools are truly random, as they indiscriminantly scan the Internet. For example, many tools have a single option: A, B, or C. The letter selected determines the size of the network to be scanned. These tools then randomly select which IP network to scan. Other tools use a domain name (`zOne` is an excellent example of this). The tools build an IP database by conducting zone transfers of the domain name and all subdomains. Blackhats have built databases with more than 2 million IP addresses by scanning the entire `.com` or `.edu` domain. Once discovered, these addresses are then scanned by tools to determine vulnerabilities, such as the version of named operating system or services running on the system. These tools often probe for a single service, then determine the version of that service. Once the vulnerable systems have been identified, the blackhat strikes.

Tools have been developed to automate this entire process. The steps of probing, identifying, and attacking systems are all built into a single package. Once launched, these automated tools spend hours doing the work for the blackhat.

For example, one of our UNIX honeypots was compromised via the `rpc.statd` vulnerability. The blackhats then attempted to use the honeypot as a platform to scan and to exploit other systems on the Internet with the same vulnerability. Their weapon of choice was an *autorooter,* a tool that automated the entire process, sequentially scanning, probing, and exploiting thousands of systems. This tool even automated the process of downloading and installing a rootkit, ensuring ownership of the compromised system. In a four-hour period, we logged the tool attempting to scan more than 500,000 systems. All these attempts were blocked; however, these numbers indicate just how aggressive and truly random these tools can be. Following are the captured keystrokes of one such attempt. Here, we see the automated tool *luckgo* being called on to sequentially scan and compromise entire class B networks. If left unchecked, such activity can damage thousands of systems. This tool has also been included with the book's CD-ROM for you to analyze.

```
Feb 18 18:49:03 honeypot -bash: HISTORY: PID=1246 UID=0 tar -xzvf LUCKROOT.TAR
Feb 18 18:49:06 honeypot -bash: HISTORY: PID=1246 UID=0 cd luckroot
Feb 18 18:49:13 honeypot -bash: HISTORY: PID=1246 UID=0 ./luckgo 216 210
Feb 18 18:51:07 honeypot -bash: HISTORY: PID=1246 UID=0 ./luckgo 200 120
Feb 18 18:51:43 honeypot -bash: HISTORY: PID=1246 UID=0 ./luckgo 64 120
Feb 18 18:52:00 honeypot -bash: HISTORY: PID=1246 UID=0 .luckgo 216 200
Feb 18 18:52:06 honeypot -bash: HISTORY: PID=1246 UID=0 ./luckgo 216 200
Feb 18 18:54:37 honeypot -bash: HISTORY: PID=1246 UID=0 ./luckgo 200 120
Feb 18 18:55:26 honeypot -bash: HISTORY: PID=1246 UID=0 ./luckgo 63 1
Feb 18 18:56:06 honeypot -bash: HISTORY: PID=1246 UID=0 ./luckgo 216 10
Feb 18 19:06:04 honeypot -bash: HISTORY: PID=1246 UID=0 ./luckgo 210 120
Feb 18 19:07:03 honeypot -bash: HISTORY: PID=1246 UID=0 ./luckgo 64 1
Feb 18 19:07:34 honeypot -bash: HISTORY: PID=1246 UID=0 ./luckgo 216 1
Feb 18 19:09:41 honeypot -bash: HISTORY: PID=1246 UID=0 ./luckgo 194 1
Feb 18 19:10:53 honeypot -bash: HISTORY: PID=1246 UID=0 ./luckgo 216 1
Feb 18 19:12:13 honeypot -bash: HISTORY: PID=1246 UID=0 ./luckgo 210 128
```

The blackhat community has developed advanced means of distributing these tools and teaching others how to use them. Two extremely common methods are Web sites and IRC channels. Blackhats set up Web sites to distribute these tools, so anyone on the Internet can easily access them. These underground Web sites are often set up on compromised systems. Little do administrators know it, but often their compromised systems are being used to distribute gigabytes of data to the blackhat community. Publicly released tools can also be found on such sites

as Bugtraq (*http://www.securityfocus.com*). To use the tools, often very simple and detailed HOWTOs are published, explaining to even the most novice users how to exploit vulnerable systems. One example is the Named NXT HOWTO distributed by the blackhat community (see Appendix C). These HOWTOs are commonly distributed with the tools themselves. Another means of communication is IRC, or Internet relay chat. IRC gives the blackhat community realtime communication. This is where the more experienced blackhats teach the beginners how to use the tools or the accounts of compromised systems. IRC also allows the transfer of files in realtime. Blackhats can quickly communicate and share the latest vulnerabilities and exploits. Chapter 11 provides examples of how blackhats use IRC to exchange tools and tactics. Another means of communication and distribution are publications. Electronic publications, such as "Phrack" (*http://www.phrack.com*), detail cutting-edge technologies. Some of these publications are also released in print, such as *2600* (*http://www.2600.com*) magazine.

THE MOTIVES

The motives vary for randomly exploiting vulnerable systems. Every time one of our honeypots is compromised, we learn the tools and tactics used, and we often also learn why the honeypot was attacked. This information can often be the most interesting and helpful.

One motive may be denial-of-service attacks. Recently, new denial-of-service attacks have been reported: DDoS (distributed denial of service). With these attacks, a single user controls hundreds, if not thousands, of compromised systems throughout the world. These compromised systems are then remotely coordinated to execute denial-of-service attacks against one or more victims. Because multiple compromised systems are used, it is extremely difficult to defend against and to identify the source of the attack. For such an attack to work, a blackhat needs access to hundreds, if not thousands, of compromised systems. To gain access to such a large number of systems, the blackhat randomly identifies vulnerable systems and then compromises them to be used as DDoS launching pads. The more systems compromised, the more powerful the DDoS attack. We saw this in Chapter 6, where the honeypot we analyzed was compromised to be used as a Trinoo client, one version of a DDoS tool. To learn more about DdoS

attacks and how to protect yourself, check out Dave Dittrich's site at *http://staff.washington.edu/dittrich/misc/ddos/*.

Another motive is for blackhats to hide or to obscure their source and identities. When blackhats attack a specific system, they do not want the attack to be traced back to them. Blackhats can obscure their true identities by compromising a system from a chain of previously compromised systems. Instead of directly attacking a system from their own location, the blackhats will compromise systems in a series of hops. After compromising one system, the blackhats hop from that system to another, and so on, continuing this series of hops until they achieve their final goal. This makes it extremely difficult to trace back to the blackhat, as one must go through a series of compromised systems. Most likely, somewhere along the line, the blackhat has effectively cleaned any tracks. To make this tracing more difficult, blackhats can compromise systems in various countries having different time zones, languages, and government structures. This makes it far more difficult for administrators and law enforcement to trace an attack. Language barriers, time zones, and political systems can make it impossible to follow the chain of compromised systems. To create such a chain, a blackhat must have access to a large number of systems.

Another motive for randomly compromising systems is IRC, or Internet relay chat. Often, blackhats want to maintain administrative rights (sys ops) on their IRC channel. To maintain such rights, the blackhats have to maintain a presence on the channel. An automated tool, *bots*, allows them to keep these rights at all times. However, *bots* can die or be taken out by other blackhats. So a common tactic is to compromise as many systems as possible and to launch automated *bots* from the compromised systems. The more systems compromised, the more *bots* the blackhats have. The more *bots* the blackhats have, the more power they have on the IRC channels. These same systems are also used to launch denial-of-service attacks against other blackhats to kill their *bots* or to remove them from IRC channels.

Also, these same IRC channels are a primary means of communication among blackhats. The Honeynet Project has repeatedly had honeypots compromised to facilitate such communication. In one situation, not only IRC *bots* were installed, but also *BNC*, a utility allowing blackhats to proxy connections through the system.

For more information on IRC and how the blackhat community uses it for communication, we highly recommend the paper "Tracking Hackers on IRC" by David Brumley, available at *http://theorygroup.com/Theory/irc.html*.

Another motivation to win is bragging rights. Many blackhats like to brag about how many systems they have compromised. It does not matter which sites they compromise, just as long it is more than everyone else. Often, blackhats advertise these acts by compromising Web sites and then modifying them to brag. Also, compromised systems can become a form of currency. Blackhats can exchange the accounts of compromised systems for things of value, such as stolen credit cards. We see these motivations in Chapter 11, in our review of the communications of several blackhats.

Compromised sites can also be used as storage and distribution centers. Blackhats will often set up websites to distribute tools, documents, cracked software—often called Warez—music, photographs, and other assorted files. Why should blackhats pay for such resources when they can use someone else's?

The motives are as varied as the blackhats themselves. There is no single, common motivation. Often, blackhats will attempt to justify their actions by claiming that their activity is politically justified, such as retaliation against an "unjust" political system or specific corporations. In Chapter 11, we see blackhats who state that they have a political agenda but appear to be out for a joyride. The Web site *http://www.attrition.org* lists Web sites that have been compromised. Spend time reviewing these compromised sites and the Web pages vandalized by script kiddies. They often post messages of political motivation. However, these justifications tend to be nothing more then conjured-up reasons for the blackhats to satisfy their own personal motives.

CHANGING TRENDS

Over the past several years we have noticed several changes in blackhats' tools and tactics. These changes pose a growing threat to the security community. Four of the most dramatic changes are in their scanning tactics, use of encryption, sophisticated rootkits, and worms.

Scanning tactics are becoming increasingly aggressive. Traditionally, blackhats took the time to identify systems vulnerable to a specific exploit before attempting to exploit them. Now, however, the trend is for blackhats not to even bother identifying such systems; they just identify a service and attempt to exploit it, regardless of the operating system or version. For example, we maintain default installations of both Linux and Solaris honeypots, both systems running `rpc.statd` service. On average, these systems were scanned one to three times a day, often forRPC. We would then log blackhats' determining whether `rpc.statd` was running on these systems (rpcinfo query). Then the blackhats simply launched their exploit script. However, the same exploit script was run against both the Linux Intel system and the SPARC Solaris system, even though the exploit works only against Linux systems.

During January 2001, 19 `rpc.statd` exploits were ran against the Solaris honeypot, even though the exploit would not work on them. This indicates that the blackhats were not taking the time to positively identify vulnerable systems. When in doubt, they simply launch their exploits and move on to the next system. These aggressive tactics could potentially harm systems by crashing services or even the system. Also, this proves that "security through obscurity" does not work. Security through obscurity is a common belief that if the vulnerability is hidden or unknown, the system is secure. For example, organizations change the version number on applications to make a nonsecure application appear secure. Organizations also modify the application so that it does not reveal the version number. Organizations believing that they there are secure using these methods are fooling only themselves. Keep in mind that blackhats often do not even bother verifying versions; they simply attack systems and move on to the next one. The Honeynet Project has seen these tactics used again and again.

The second trend, encryption, makes tracking blackhats more difficult. Traditionally, the Honeynet Project tracked blackhat activity by capturing their keystrokes, by sniffing the network. However, this method is no longer valid, as blackhats are using encryption to communicate with compromised systems. Many operating systems, such as Linux or OpenBSD, come with *ssh*. Once a system is compromised, blackhats will use *ssh* instead of TELNET to control the exploited system. *Ssh* encrypts all blackhat traffic, protecting it from intrusion

detection systems or sniffing on the network. Even if encryption utilities are not installed, the blackhats will install their own. In the past five attacks on our honeypots, blackhats uploaded and installed their own encryption utilities to ensure that their actions could not be monitored. In all cases, trojaned versions of *ssh* were installed, not only encrypting their activity but also putting a backdoor into the system. Encryption makes tracking blackhats far more difficult. In response, we have been monitoring blackhat activity at the system level, such as trojaned shells or kernel drivers that capture keystrokes, and forwarding that activity to a trusted system.

The third development the Honeynet Project has seen is the use of more advanced rootkits. Traditional rootkits replaced system binaries, hiding the blackhat's activities and implementing backdoors. Recently, we have seen more advanced rootkits used, loadable kernel module rootkits, such as Adore, which modify the kernel of the operating system. Then no activity on the system can be trusted. Even if you upload trusted binaries on the system, such as *ls* or *find*, their output cannot be trusted, as the kernel cannot be trusted. Blackhats are becoming more and more difficult to track once they compromise a system. What is significant about these kernel-level rootkits is that the binaries on the system are not modified. With traditional rootkits, an attacker modifies the binaries, such as *ls* or *whois*, which means that programs like *Tripwire* can detect that the file has been modified. But because the modifications are being done at the kernel, the binary files do not change, which means that programs like *Tripwire* can no longer detect that a rootkit has been installed. These kernel-level rootkits are both very powerful and very difficult to detect.

The fourth trend we have seen is one of the most frightening. Blackhats have created worms that not only automate the probing and attacking but also are self-replicating. This means that systems can be exponentially exploited by the blackhat community, with little or no involvement on their part. After compromising a system, the worm then uses that system as a base to replicate itself by scanning and exploiting other systems. The worm continues this process, gaining control of as many systems as possible. We see an example of one such worm in the next chapter. Traditionally, worms have been limited to Windows-based systems. However, in early 2001, we witnessed a growing number of worms, such as

Ramen, Lion, or *Sadmind/IIS*[1] that attack UNIX systems. These worms are based on the same tools and vulnerabilities we have discussed so far; it is the fact that they are self-replicating that makes them so dangerous. You can find team member Max Vision's detailed writeup of the worm *Lion* at Vision at *http://www.whitehats.com/library/worms/lion/index.html,* or on the CD-ROM that accompanies this book.

SUMMARY

We have completed our overview of the blackhat's motives. This in no way means that all blackhats operate and think in the ways we have described; what we have just described is a generalization. However, these are the common tools, tactics, and motives the Honeynet Project has encountered over the past several years. This is also a common threat that we all face, regardless of what type of connection or organization you represent. This threat is also continually growing, changing, and improving all the time. It is almost certain that your organization will have to deal with this enemy.

1. Ramen: *http://www.cert.org/incident-notes/IN-2001-01.html.* Lion: *http://www.cert.org/incident-notes/IN-2001-03.html.* Sadmind/ITS: *http://www.cert.org/advisories/CA-2001-11.html.*