

Introduction

INFORMATION IN THIS CHAPTER

- Book Overview and Key Learning Points
- How This Book Is Organized

BOOK OVERVIEW AND KEY LEARNING POINTS

Security is heavily *contextual*; the effectiveness of any security measures depends on the context into which they are deployed. What if you give keys to the janitor, and he or she leaves them in his or her unlocked car? Further security is often *not incremental*; insecurity in one area can lead to insecurity in *all* areas. Hackers might break into your machines and steal your proposals and bidding information, so you carefully secure your network. Hackers might break into employees' home networks to steal passwords, e-mail accounts, or even hijack "secure" connections to break into your corporate network, so you institute policies about remote access. Hackers might park outside your building and "listen in" on your wireless network, so you encrypt it and use special measures to prevent the wireless signal from leaking outside the building. Hackers might use e-mail "phishing" and other "social engineering" attacks to gain access, so you add more policies and carefully train your staff and test them from time to time. Finally, comfortably secure and ready for anything, you unknowingly hire the hackers and fall victim to an "insider" attack. Life's tough.

What we think of as security is really a collection of policies and procedures that are, ultimately, about *giving out* information. Your employees (or even other parts of your infrastructure) *need* information to accomplish their mission. Security stands between your employees and accomplishing that mission. All too often serious security breaches start with some otherwise well-intentioned effort to get some useful work done. Sometimes, it is your employees who break your security; not necessarily because they have some evil purpose, but sometimes because they believe the mission is more important or that the security measures are unnecessary. The mission may be short term and absolutely critical. The effects of a security breach can take years to evolve or even to be detected.

It is late in the day and you have a very important bet-your-company deliverable due out in the morning. You desperately need Software X to run in order to finish the

deliverable, but Software X is being blocked by your firewall. You've tried adding rules to the firewall, you've tried calling the vendor, but nothing is working. Finally you disable the firewall, finish the deliverable, and ship. Will you remember to re-enable the firewall? Did you monitor your network while the firewall was down? The view that security is a collection of tradeoffs, or a series of calculated risks, assumes a *continuous* nature to security. The belief that you can trade *a little* insecurity for some other gain is often a misunderstanding of the nature of security. This is akin to saying you will allow anyone to withdraw money from your bank account but only as much as they can withdraw in 10 minutes. The mistake is that the two things (in this case money and time) are not directly related.

HOW THIS BOOK IS ORGANIZED

This book identifies seven classes of network attacks and discusses how the attack works, including tools to accomplish the attack, what are the risks of the attack, and how to defend against the attack. Seven attacks were chosen: denial of service, war dialing, penetration testing, protocol tunneling, spanning tree attacks, man-in-the-middle, and password replay. These are not mutually exclusive; you can exploit the spanning tree protocol, for example, to launch a denial-of-service attack. These were chosen because they help illustrate different aspects of network security; the principles on which they rely are unlikely to vanish any time soon, and they allow for the possibility of gaining something of interest to the attacker, from money to high-value data.

Chapter 1, “Denial of Service,” illustrates how even sophisticated networks can be crippled by a determined hacker with relatively few resources.

Chapter 2, “War Dialing,” illustrates how a hacker can circumvent the hardened security perimeter of a network to access “softer” targets.

Chapter 3, “Penetration ‘Testing,’” discusses the various tools and techniques used for penetration testing that are readily available to both the defenders *and* the attackers.

Chapter 4, “Protocol Tunneling,” presents a method for deliberately subverting your network perimeter to “tunnel” prohibited traffic into and out of your network.

Chapter 5, “Spanning Tree Attacks,” discusses the “layer 2” network responsible for knitting together your switches, routers, and other devices into a reliable network, and illustrates one way in which to exploit the weak security of this layer.

Chapter 6, “Man-in-the-Middle,” discusses a very common attack pattern and just what an attacker can accomplish once he or she has inserted himself or herself into your data stream.

Chapter 7, “Password Replay,” focuses on the security of passwords and other static security measures and how an attacker can use various techniques to gain unauthorized access.

This book is intended to provide practical, usable information. However, the world of network security is evolving very rapidly, and the attack that works today may (hopefully) not work tomorrow. It is more important, then, to understand the principles on which the attacks and exploits are based in order to properly plan either a network attack or a network defense. The authors chose the contents of this book because we believe that, underlying the attacks presented here, there are important principles of network security. The attacks are deadly because they exploit principles, assumptions, and practices that are true today and that we believe are likely to remain true for the foreseeable future.

Increasingly sophisticated criminal organizations launch network attacks as a serious, for-profit enterprise. Similarly, well-funded governmental actors launch network attacks for political reasons or for intelligence gathering. Cyberspace is already a battlefield. Even if your network doesn't have high-value intelligence and you don't have deep pockets, you may be the target of a sophisticated attack because you have something else of value: machines and network access. An attacker may exploit your network to launch malware or to launch a network attack. Your Internet Protocol address may serve to give the attacker a level of plausible deniability. After all, would you want to launch the virus you just finished creating through your own Internet service provider connection? Attackers may use your machines for storage of information ranging from child pornography to stolen credit card numbers. Once these show up on your machines, it becomes your job to explain how they got there. Attackers can use compromised machines for command and control of deployed and distributed malware. This can result in your network being blacklisted or blocked as a distribution source for malware. Is this the company image you want your customers to see?

As networks grow and incorporate more sophisticated technologies, it can become difficult to maintain the necessary situational awareness. What were once "dumb" network nodes such as printers and network hardware may now have exploitable – and unexpected – vulnerabilities. These components are – in reality – just other computers on the network. Some of them have multiple interfaces that need to be considered, including Bluetooth, wireless, and wired connections. If one interface is well protected and another disabled, there may still be a third that is available. Network security requires considering the role and security concerns of each device, not just delivering the device and plugging it in.

There are many reasons why network security is hard, ranging from the fact that networks are increasingly sophisticated and complex to the fact that economic incentives can work against proper security. Network security is essentially *asymmetric warfare*; your adversaries can probe anywhere, but you have to defend everywhere. This creates a technological bias in favor of the attackers. Further, criminal organizations live in a target-rich environment. If they are unsuccessful with one attack, they can move on and attack a different organization.

The market for computer security products can – and does – fall prey to the *asymmetric information* problem. This is a case in which buyers of a product do not have as much information about the relative merits of the product as the sellers do. This creates a downward pressure on prices that, in turn, creates a downward pressure on quality.

Consider a used car market in which there are 100 good cars (the “plums”), worth \$3000 each, and 100 rather troublesome ones (the “lemons”), each of which is worth only \$1000. The vendors know which is which, but the buyers don’t. So what will be the equilibrium price of used cars?

If customers start off believing that the probability that they will get a plum is equal to the probability that they will get a lemon, then the market price will start off at \$2000. However, at that price only lemons will be offered for sale, and once the buyers observe this, the price will drop rapidly to \$1000 with no plums being sold at all.¹

CONCLUSION

Network security depends on many factors, and perfect network security is impossible. Network protocols can be inherently insecure in surprising ways. Cryptographic functions that are essential to network security can fall prey to sophisticated mathematical attacks. The algorithms that implement protocols or cryptography can contain bugs. Even otherwise correct code can fall prey to the effects of being run on a computer; errors exist in chip designs, and the use of finite-precision math on computers can result in unexpected effects that can be exploited. This is all good news for attackers—but not so much for defenders.

Of course, all is not lost. As a network administrator, you may have other factors on your side, including support by law enforcement, governmental agencies, and trusted third parties such as CERT^A and SANS.^B You have to control what you can. Stay educated on threats and responses. Make sure procedures support good security, and that personnel are properly trained. Make plans to deal with attacks. Most importantly, you need to understand how and why network attacks work. It is our hope that this book will contribute to that goal.

Endnote

1. Anderson R. Why information security is hard – an economic perspective. Proceedings of the 17th Annual Computer Security Applications Conference (ACSAC); 2001 Dec.

^A See www.cert.org/

^B See www.sans.org/

Denial of Service

1

INFORMATION IN THIS CHAPTER

- How Denial of Service Works
- Dangers of Denial of Service
- Defense against Denial of Service
- The Future of Denial of Service

On April 26, 2007, the nation of Estonia was hit with a denial-of-service (DoS) attack. The attack lasted, off and on, until May 18th of the same year. The attack effectively cut off Internet access for much of the country. Members of the Parliament could not access their e-mail, people were unable to access their online banking accounts, Estonian news agencies could not communicate outside the country's borders, ATMs ceased to work, and citizens traveling abroad discovered their debit cards no longer worked.¹

Estonia was not overcome because of outdated infrastructure. It was (and is) one of the most “wired” countries in Europe, thanks to their Tiigrihüpe (Tiger's Leap) project. In Estonia, as in France and Greece, Internet access is regarded as a basic human right, and the Estonian government has invested heavily in information technology (IT).

One might also be tempted to dismiss an Internet outage as nothing serious. Of course, if your business depends on the Internet, you may feel differently. Estonia's largest bank, Hansabank, is estimated to have lost around \$1 million as the result of the attack. Banks are increasingly dependent on Internet banking and foreign money transfers, and thus an “always on” Internet. If the Internet *is* your business, as with Amazon.com and eBay, the effect can be disastrous.

Was this attack the result of careful planning by a foreign government? It now seems likely that the attack was organized and coordinated by one man: a 22-year-old Russian named Konstantin Goloskokov. He apparently carried out the attack in protest of the Estonian government's decision to move the Bronze Soldier, a war monument in Tallinn erected by the Soviet Union in 1947. At the time of

writing, the Estonian government has arrested and convicted just one person: Dmitri Galushkevich, who took part in the attack working from his laptop.

DoS attacks are on the rise and can be perpetrated by large-state actors, experienced hackers, or even by novices (“script kiddies”) following any of the “how-to” manuals found on the Internet. DoS attacks can be launched for any number of reasons, from political protest to espionage and even extortion. These attacks can be intentional, like the one just described, or unintentional, like the “Slashdot” effect.

As an example of unintentional DoS, suppose several aggregators, including Slashdot^A and Digg,^B pick up your essay on why Data was the best acting captain in *Star Trek* history. Now, thousands of people are visiting your site every minute, and the bandwidth allocated to you by your Internet service provider (ISP) is quickly used up. Now *nobody* can get to your site, not even you. Worst of all, you can’t post the adorable video of your cats dressed as the crew of the enterprise. You’ve been the victim of unintentional DoS. You may even get a bill from your ISP for the extra bandwidth.

This chapter will focus on intentional DoS – a denial-of-service *attack*. DoS attacks can be launched for a number of reasons; the Estonia case was a sort of protest but they can be used to damage competitors for financial gain. In 2004, businessman Saad “Jay” Echouafni allegedly hired computer hackers to launch a DoS attack on three of his competitors. Another application of DoS attacks for financial gain is extortion. A company receives a threat that they will be subjected to a DoS attack unless they wire money to an offshore account. In many cases, the company will simply pay. In 2004, Carnegie Mellon University surveyed 100 companies. They found that 17% of medium-size businesses had been the target of some form of cyber-extortion.^C

HOW DENIAL OF SERVICE WORKS

DoS requires two elements: a resource of finite capacity, and the means to acquire or “use up” the resource faster than it can be replenished. Although we generally think of these attacks in terms of computers, DoS attacks do not have to be network-based. It is possible to have “real-world” DoS attacks, provided you have the above two elements.

Real-world examples include the practice of “land blocking” where a company purchases the land around a store to prevent competitors from opening nearby, and many of the methods used by DeBeers to control the diamond market in the twentieth century.^D These are examples of a single entity that is powerful enough to consume

^A<http://slashdot.org/>

^B<http://digg.com/>

^CAs of August 2009, the full report is available online: http://heinz-racer.heinz.cmu.edu/whatsnew/images/CMU_Cyber_Extortion_Study.pdf.

^DFor an excellent history of the diamond market, see *The Diamond Ring: Business, Politics, and Precious Stones in South Africa, 1867–1947*, by Colin Newbury, Oxford University Press, 1990.

enough of the available resources (land, diamond production, or any resource your competition needs) to disrupt or starve others. In general, this requires deep pockets or significant resources, which makes it much less likely to occur than DoS attacks in the virtual world.

Distributed Denial of Service

To conduct a successful DoS attack, you usually need a lot of help. This is the origin of the *distributed* denial of service attack (DDoS). As an example of a DDoS attack for the physical world, consider the following: you admire the cool-headed reasoned approach of Captain Picard over the random cowboy style of Captain Kirk, and decide that what the world needs is a statue of Picard in San Francisco, the (future) home of Starfleet. To this end, you start a campaign to raise money, and people begin sending you checks.

Sadly your “friend” Mike does not agree and makes his mission to stop you. He recruits friends to send you hate mail, and soon your mailbox is stuffed with angry letters about why Kirk is better than Picard. Sorting through the mail takes longer and longer, and you only find a few checks in every batch of letters. Soon you’ve got friends involved to sort the mail, but sending the same angry letter multiple times is easier (and cheaper) than sending a new check, so the volume of hate mail far outstrips the volume of checks. You need more and more time to sort the mail, for fewer payoffs.

Everyday your mailbox is stuffed full. The post office begins to hold your mail because it cannot deliver it. Now you must drive to the post office to collect boxes of (mostly) photocopied hate mail, and you have to open every letter because you can’t easily tell which ones contain checks. Ultimately you may have to abandon your quest, noble though it may be.

Although a DDoS attack is hard to pull off in the physical world, DDoS is the most common – and disturbing – form of DoS attack in the virtual one. Utilizing DDoS techniques and the Internet, small groups (or even a single individual) can conduct massive DoS attacks. The rest of this chapter will focus solely on DDoS attacks.

Overview of a Denial of Service Attack

Suppose you want to conduct a network DDoS attack against a particular *victim*. You are taking the role commonly called the *intruder*. To conduct a DDoS attack, you need to be able to “use up” some resource needed by the victim. You can target any resource likely to interrupt your target. For instance, if you are targeting an online retailer, you might do any of the following.

- Overload the Web servers.
- Overload any network link.
- Crash servers.
- Attack a dependency.

In the last case you don't attack the victim directly, but you might attack their bank, their credit card clearing company, their accounting firm, or one of their suppliers, making it impossible for them to conduct business.

It is likely that your target, especially if it is a bank or online retailer, has a lot of capacity you must use up before you are successful. Your computer just cannot generate enough network traffic to launch a successful DDoS attack. You need help.

In some cases, it may be possible to use social engineering to accomplish your goal. For example, if you can start a successful Internet rumor about your victim, you may be able to get others to do your work for you. For example, you might post a long and official-looking "news" story about how your "friend" Mike was caught raising Dalmatian puppies for their fur. Soon activists are calling him at all hours, filling *his* mailbox with hate mail, and even stopping by his house. Mike can no longer go to the store, let alone continue his DDoS attack against you. Sadly, his friends may carry on in his absence.

Most often you will instead attempt to gain control of a large number of computers from which to conduct the attack. During the 1990s, you might have targeted universities because they had large numbers of always-connected machines with fast connections and (typically) low security. Today, university networks are better protected and monitored, but the rapid growth of the Internet means you can find a large number of always-connected machines with reasonably fast connections and low security in peoples' homes and small businesses. If you can compromise enough of these machines, you can launch your attack.

EPIC FAIL

If you aren't careful, you can launch a DDoS attack on yourself. In 2003, the University of Wisconsin at Madison was flooded by as many as 280,000 Network Time Protocol (NTP) packets per second. The source? Netgear routers (about 700,000 of them) were sending requests for the current time. These routers were configured such that, if no answer was returned within 1 s, they sent another request. If a router got a satisfactory reply, it would then wait for a period of time and update again. Once the system at Madison became congested, this approach (try again every second) just made matters worse, and even machines with a satisfactory reply would be asking again before the queue had been cleared.²

At the time of the writing of this book, the largest networks of compromised machines are believed to be the Conficker and Srizbi *botnets*, containing around 1 million and 450,000 compromised computers, respectively. Its primary use is sending spam; a different sort of DDoS attack that slows your ability to read your e-mail.

Recruitment

This initial phase of collecting machines you can control, or *zombies*, is called *recruitment*. During the recruitment phase, you need to find machines you can take over. The process of looking for vulnerable machines is called *scanning*. While scanning can be done manually, it is obviously better to let computers do the work.

Several network scanners exist to find vulnerable machines, including Nmap,^E Nessus,^F and SAINT.^G You can specify particular machines to scan, scan entire blocks of the network, or scan randomly for vulnerable machines. These tools can help you discover vulnerable machines on your network so you can patch or remove them. Alternately, they can be used to find vulnerable machines on *other people's* networks so that you can make them into zombies.

Figure 1.1 shows the result of a network scan using Nmap. The Zenmap graphical user interface is shown. Figure 1.2 shows a vulnerability detected using Nessus.

Exploitation

Once you've found machines that are vulnerable, you can move on to *exploit*, or gain access to, the machines. There are many ways to exploit a machine, depending on what operating system and patches are installed. Once again, this is a task best turned

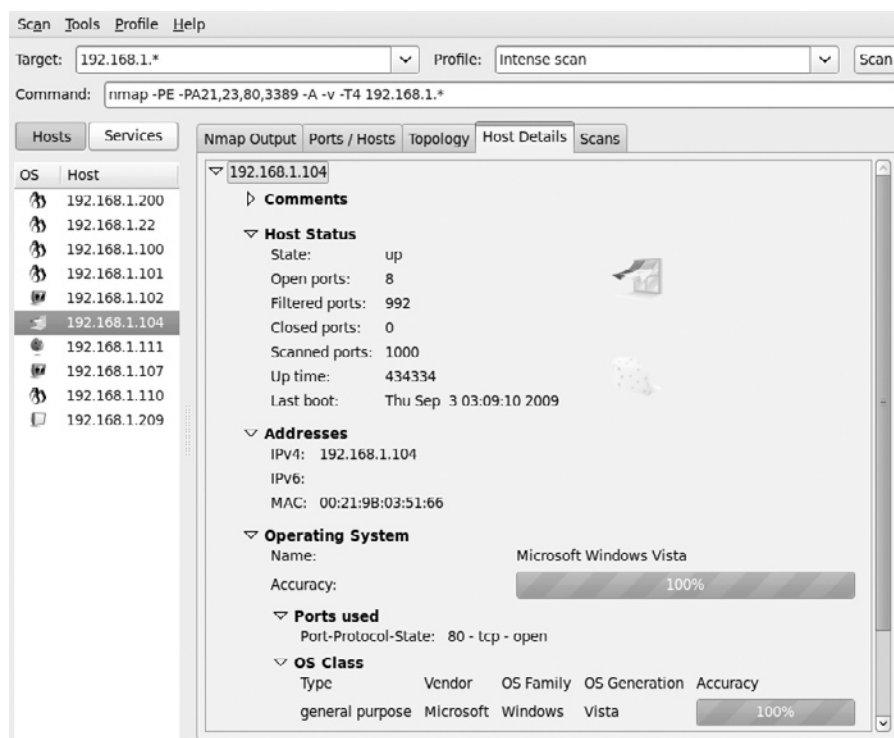


FIGURE 1.1

Scanning a Network with Nmap

^E<http://nmap.org/>

^FNessus is a registered trademark of Tenable Network Security: www.nessus.org/nessus/. An open-source version of this tool called OpenVAS also exists: <http://openvas.org/>.

^GSAINT is a registered trademark of SAINT Corporation: www.saintcorporation.com/.

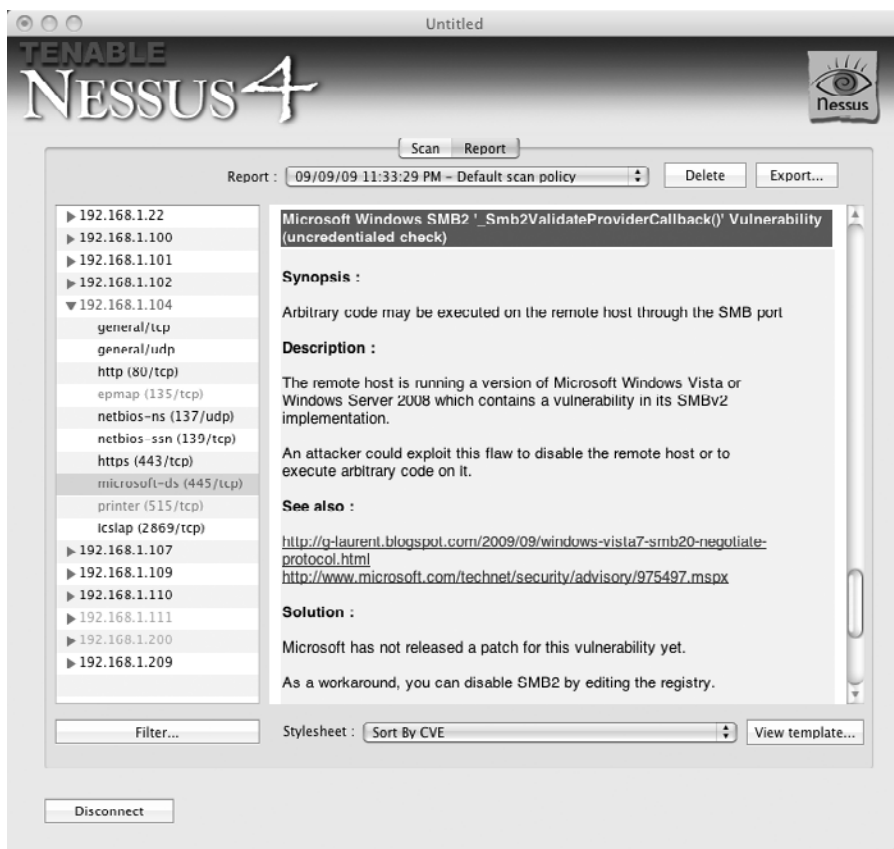


FIGURE 1.2
Detecting a Vulnerability Using Nessus

over to the computers themselves. The Metasploit Framework^H is a tool specifically for developing and executing exploit code against a remote machine, and its use is described in Chapter 3, “Penetration “Testing””.

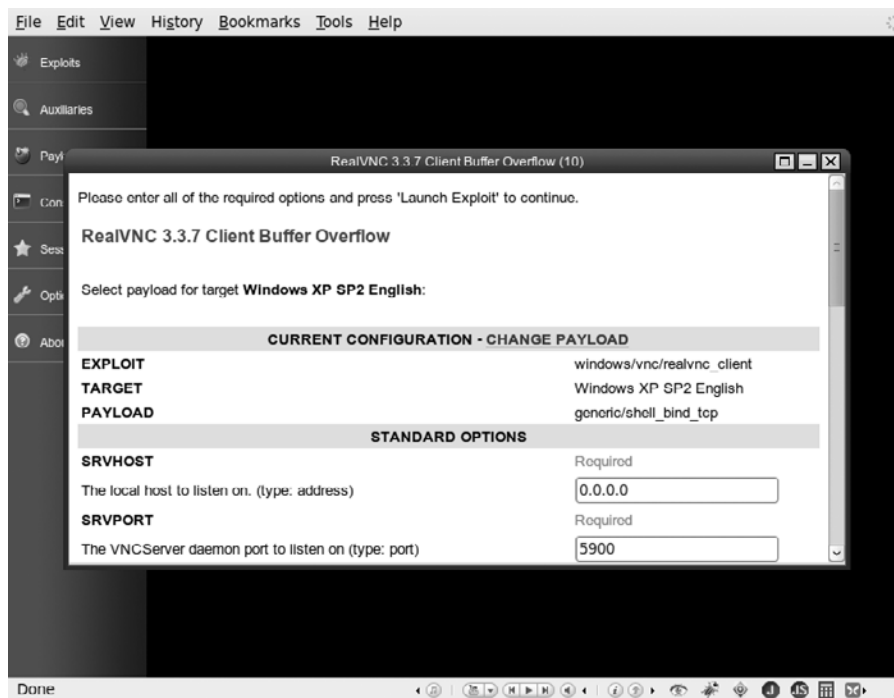
Figure 1.3 shows a configuration page for an exploit, using Metasploit’s Web interface.

Installation

Now that you have found and exploited a vulnerable machine, you need to install the *payload* software. This software typically includes several items.

- Task-oriented software to perform the actual DDoS attack.
- Command-and-control software so you can control, coordinate, and launch the attack.

^Hwww.metasploit.com/

**FIGURE 1.3**

Configuring an Exploit with Metasploit

- Antidetection software to prevent your software from being discovered and removed.
- Update software so you can remotely update the machine.

Control

The software to control the machine is typically called a *bot* (short for “robot”), and the resulting network of machines is called a *botnet*. The bot listens for specific control strings on an Internet Relay Chat (IRC) channel and responds on the channel or takes other actions. It is not uncommon for compromised machines to act as IRC “routers” (to pass along messages) and “proxies” (to hide the fact that IRC is being used).

More recently, some botnets have been found that use instant messaging (IM) and even Twitter¹ for command and control. Any lightweight, decentralized network will do. IM can even be used for the infection phase, by sending a malicious link for a user to click, or by exploiting a vulnerability in an IM client.

¹<http://twitter.com/>

Automation: Worms

You can proceed to compromise more machines, but one approach is to take the above process and embed it in a worm. A *worm* is a program that *automatically* propagates from one vulnerable machine to another. The worm automates the scan-exploit-install process, copying itself to each exploited machine, and then using that machine to launch further attacks. This gives several advantages:

- The scanning and exploitation of machines is parallelized, so you can grow your botnet much faster.
- You may find helpful information on the machine itself, such as the addresses of other machines in a local network or log files containing other addresses you can scan.
- The path used to exploit a machine is very difficult to trace back, making it harder to find you.

Propagation

The worm may or may not contain the payload. Some worms “phone home” after exploiting a machine to obtain the payload. This is convenient because it allows the intruder to manage the payload centrally. This is convenient for the defender (your intended victim) because they can potentially detect or defeat you by watching for or by blocking communication with your host. The SoBig worm used this method; it was programmed with 20 different addresses to contact and download software.¹

Other worms may use a “back-chaining” method, where a worm propagating from host A to host B first exploits host B, and then later pulls the payload from host A. This can be useful if the payload is large, and the transfer may be interrupted, perhaps by a network outage. The worm can resume the transfer later when the network is again available. This also gives the worm a chance to disable scanning and antivirus programs before installing the payload, perhaps reducing the chances of detection. The Ramen worm that infected Linux machines used this method. Finally, the worm may act as a single program, containing both the propagation software and the payload. The tiny, but virulent SQL Slammer worm used this method, as did the Morris worm.

NOTE

Many regard the Morris worm as the first computer worm. A mistake in the worm's propagation code resulted in a severe, though unintended, DoS attack. This led DARPA (the United States Department of Defense Advanced Research Projects Agency) to fund the creation of the CERT/CC at Carnegie Mellon University to serve as a central point for coordinating responses to network emergencies. The cost of removing the Morris worm has been estimated at between \$10 million and \$100 million.

¹SoBig illustrates some difficulties in classifying malware. It can propagate on its own, so it is a worm, but it can also “pretend” to be an innocuous e-mail attachment, so it may be classified as a Trojan horse.

Different variants of the Conficker worm use different propagation methods. Interestingly, some variants select domain names based on the date, and then try to contact each of them. This creates many potential rendezvous points for the intruder to control and update the worm, while making it hard to defeat the worm by blacklisting or detecting addresses.^K

The worm may find new machines to infect by randomly trying addresses, but it is also possible to provide the worm with a “hit list” of address blocks to try, so that the propagation is not completely random. For instance, you may want to avoid scanning certain targets (such as network security sites), since the worm may be detected. You might also want to avoid certain hosts for legal reasons. Some malware avoids infecting machines in certain countries, possibly so the author can avoid prosecution by the local authorities. Swizzor will not infect machines using the Russian language version of Windows, and Conficker does not infect machines using a Ukrainian keyboard layout.

Launching the Attack

Now that you have a botnet, you are ready to launch the attack. At this point, there are several methods open to you:

- You can preschedule your attack, embedding it in the worm itself. In this case, you do not need to communicate with the infected machines, possibly reducing the chances you will be discovered. On the other hand, this approach is far less flexible.
- You can send a command to your botnet to initiate an attack, perhaps of a specific duration and type. This allows you to collect information about the result of the attack, and then plan the next phase.

The only remaining item is what you specifically do to the victim. Again, you have several options:

- **Crash machines** By exploiting vulnerability in operating system software, you may be able to cause a machine to fail or reboot. The *teardrop* attack is an example of this kind of attack.^L
- **“Break” something** It may even be possible in some instances to corrupt the machine’s code so that it will not successfully reboot. This is sometimes called *permanent* DoS or “phlashing.” If you can corrupt the firmware (the most basic programming a machine uses to start itself) or the boot sector of a hard disk (the code used to load the operating system when the machine boots up), you may be able to “brick” the machine, or permanently disable it.
- **Overwhelm something** This is the classic distributed DoS and is discussed in more detail below.

^KFor a lot of analysis of Conficker, visit the Conficker Working Group on the Web, at www.confickerworkinggroup.org/.

^LLong messages sent on the Internet must be broken into a sequence of packets. These packets are then sent, and reassembled into the original message at the destination. The *teardrop* attack exploits this by creating and sending a series of bogus packets that overlap, are too large, or otherwise do not constitute a valid message when reassembled. This exploits bugs in the reassembly algorithm; when the receiver tries to assemble the packets into a complete message, it crashes.

As with scanning and exploitation, there are ready-made toolkits to get you started. Agobot is an open-source worm that requires very little programming skill to use. It has been released under an open-source license, has since been specialized with many variants, and includes a variety of features for propagation and for conducting DDoS attacks.^M There are many other well-known DDoS tools, including Trinoo, Shaft, and Stacheldraht.^N

Ping Floods

There are many ways to overwhelm your victim. Most of these rely on exploiting Internet protocols, but it is often possible to overwhelm a host by simply sending lots of otherwise legitimate messages.

Suppose you are headed to a *Star Trek* convention where you will have the chance to actually meet Patrick Stewart. Sadly, if anyone says hello to you and offers to shake your hand, you are obligated to say “hello” back and shake their hand. Upon arriving you are greeted by your “friend” Mike and 300 or so people he rounded up for this event. You spend the entire time saying hello and shaking hands, and never get to see Patrick Stewart. You’ve just been the victim of a *ping flood*.

Internet Control Message Protocol (ICMP) is one of the foundational protocols on which the Internet is built. To determine whether a particular host is accessible and working or to determine the *latency*, or time to communicate with a host, ICMP provides a special kind of message called an *echo request*, or more commonly, *ping*.^O Ping messages are used to measure the round-trip time for a message to reach a destination and return, as well as to gauge the quality of a connection.

When a machine receives an ICMP echo request (ping), it must reply to the sender with an ICMP echo reply containing the same data as the original request. In other words, if someone says hello you must say hello back. Because of this requirement, if you flood the victim with pings from many different hosts you can overwhelm the machine, and it will spend all its time and network bandwidth replying to pings, creating a very effective DDoS attack. This has become less useful in recent years, as many network administrators have started filtering ping messages at the edge of their network, or simply by shutting off the ICMP echo server.

There are many variants on this theme, including SYN floods (another kind of Internet message that can use up resources), and the “Smurf” attack that combines pings with a special network address called the *broadcast address* to use up network bandwidth.

^MAgobot is an example of a “kit.” It provides an extensible shell for implementing new exploits. Included with Agobot is a set of helpful notes explaining how to build Agobot, and some later versions (Phatbot) include a GUI to assist in customization.

^NThe source for these and other malware can be found on the Packet Storm Web site: <http://packetstormsecurity.org/>

^OThe technical details of the Internet are found in a series of documents called requests for comments, or RFCs, and you can find them at many sites, including www.faqs.org/. The ICMP echo request and reply messages are detailed in RFC 792 and RFC 1122.

DNS Amplification Attacks

Another part of the Internet's infrastructure can be exploited for a DDoS attack. Domain name service (DNS) is responsible for translating a host name like google.com into a network address that can be used to actually connect to the host. A request for a network address can be very small (a few bytes), but yield a very large reply (a few kilobytes). This effect is called *amplification* and is the key to this kind of attack.

Machines in your botnet send out special messages to domain name servers requesting a lookup of an address. The messages sent are *spoofed*. That is, they lie about their return address so that they appear to come from your victim. The replies generated by the servers then flood the victim, choking off its network bandwidth, and making it unavailable. It is quite common to use spoofing to hide the source of an attack.

This attack depends on finding name servers (called *open resolvers*) that will answer a recursive query for any host. Name servers are not required to answer recursive queries, but sometimes are incorrectly configured to do so. In a *nonrecursive* query (also called an *iterative* query), a client asks a name server about a hostname. If the name server does not directly know the hostname, it replies with a referral to another, more authoritative name server. The client should then query that server. This process may repeat several times, resulting ultimately in either failure or success. A name server configured to perform a *recursive* query does all this work on behalf of the client.

Service-Level DDoS

You may also attack your victim by sending a flood of legitimate-looking service requests. For instance, you might have all the machines in your botnet flood a Web server with requests for pages. This is a *service-level* DDoS attack, and it is very hard to detect and defend.

A Web server can typically handle a certain volume of traffic for a page. Suppose that a Web server can handle at most 5,000 requests per second for a particular page. Given this, a botnet of 10,000 machines can easily generate 5,000 requests per second, leaving the Web server maxed out. Additional, legitimate traffic only gets through occasionally, resulting in timeout failures. In this case, the DDoS attack may not be evident to the victim at all, though legitimate users of the Web service are affected.

Further, the Web server can successfully respond to all requests, but still fail from the point of legitimate requests. During a DDoS attack, your server may have to handle 1,000 bogus requests generated as part of the attack for every legitimate request. If each request must be handled in turn, then the 1,000 bogus requests will delay servicing of the legitimate request. Even if it is eventually handled, the user may have observed a significant delay, given up, and moved on. They may try their request again later, or they may simply move on to one of your competitors, remembering that your site is slow.

It is also possible to exploit the computational cost of certain operations to affect a DDoS attack with much less traffic. Requesting a simple HTML page from a server

is quite different from searching a product catalog or trying to log in. If you can find an operation on a site that requires significant computation, you can target that operation in your attack. One emerging tactic is to flood a site with bogus login requests, as checking a user name and password takes longer to handle than a simple page request. This approach relies on computational properties of the requests and can be thought of as a *semantic* DDoS attack.

DANGERS OF DENIAL OF SERVICE

If your DDoS attack exploits a vulnerability of your victim, you may only need to generate a very small amount of traffic. If you can successfully crash a machine, you may cause your victim to suffer downtime, lose business, or even lose customer data. This gives DDoS a high payoff for a low input.

Flood attacks require generating a lot more traffic, but require significantly less sophistication to conduct. If you can generate enough legitimate-looking Web traffic, you can flood a Web server and shut down an online business. Again, there is a high payoff for a (relatively) low effort.

The use of botnets helps insulate intruder from the victim, making it harder to track down and prosecute the intruder. Further, if you can grow a large botnet, you can extort money from many large companies, and even conduct more than one DDoS attack at a time. Some people accumulate botnets and then essentially lease them to others to conduct the actual DDoS attack.

A DDoS attack is especially deadly because of key factors:

- It is *simple*. Recruit many machines and have them flood your target with legitimate-looking traffic. There are many automated tools to download, along with tutorials and other help for inexperienced users. Even tools that have been around for years are still very effective and can be used with little or even no modification and tweaking.
- It is *effective*. With enough machines generating traffic you can bring down even the largest targets, either by overloading their servers, or by consuming all the network bandwidth.
- It is *cheap*. DDoS attacks use other people's resources, in the form of zombie machines. The tools and information necessary to conduct a DDoS attack are available for free on the Internet. All that an intruder really requires is access to a computer to launch the attack.
- It can be fairly *safe*. If your victim is in a different country it may be very hard, or even impossible, to arrest you, even if your identity is discovered. Konstantin Goloskokov, the self-identified leader of the Estonia attack described at the start of this chapter, remains at large and it is unlikely he will be prosecuted. To be sure, this last point depends on where you live, and whom you choose as your victim. As mentioned previously, intruders often intentionally avoid exploiting machines in their home country.

As more economic activity moves onto the Internet, the use of DDoS for extortion, terrorism, and other purposes will increase. In particular, DDoS extortion leverages a high payoff from a low cost of entry. While people have been caught and prosecuted for such cyber crime, it is widely assumed that, at least at present, many companies prefer to simply pay a “reasonable” amount to avoid disruption, and do not report the incident to law enforcement. Extortion has become a cost of doing business on the Internet.

DEFENSE AGAINST DENIAL OF SERVICE

Recall the two essential ingredients of a DoS attack.

- A resource of finite capacity, and
- The means to acquire or “use up” the resource faster than it can be replenished.

This immediately suggests one possible defense strategy: increase the capacity of the resource, or the rate at which the resource is replenished. Returning to the earlier example of the Picard/Kirk letter campaign, increasing the capacity of the resource is like getting a bigger mailbox; increasing the rate at which the resource is replenished is like getting more friends to help you sort your mail. As you can probably tell, neither approach guarantees success. It is still easier for your enemies to generate work for you than it is for you to do that work.

You can make a list of the people who send you hate mail, and discard letters from those people. That is, you *blacklist* certain senders. Unfortunately, the senders don’t include return addresses on the envelopes, or sometimes *spoof* their return address, giving someone else’s instead. This tricks you into opening the letter from Aunt Martha to discover another rant about how Kirk was the only real captain of the Enterprise. You can also base your blacklist on the postmark, and begin dropping all mail from certain zip codes. This means you are going to discard some checks, too (*false positives*). This strategy corresponds to blocking Internet Protocol (IP) address ranges.

The only completely effective way to secure against a DDoS attack is to secure a very large portion of the machines attached to the Internet against misuse. Unfortunately, this is impractical because the Internet spans political and even continental boundaries. Keeping machines up-to-date on patches can help prevent them from becoming part of a botnet. Sadly, when a security hole is detected, an exploit can often be generated long before a patch is available. When a patch is released it is typically hours, and sometimes days, before machines are updated. The *automated* generation of exploit code from the patch has been demonstrated, and typically takes only minutes.^P

The lesson here is that completely securing machines against a DDoS attack is probably not possible (though we should certainly try!). Much of the focus must be on *survivability* instead of security. That is, it may be impossible to prevent a DDoS attack, but you can take steps to survive the attack.

^PSee David Brumley, *et al.*, “Automatic Patch-Based Exploit Generation is Possible: Techniques and Implications,” in the *Proceedings of the IEEE Security and Privacy Symposium*, May 2008.

General Advice

Before we get into the details of defending yourself against a DDoS attack, here are a few pieces of general advice:

- If you are the victim of a DDoS attack, save your traffic logs, record everything you observe, and keep a record of everything you do. This is essential if you intend to work with law enforcement to try to track down the intruder, but also provides a record you can use after the attack to better understand what happened and how you can better defend your network in the future.
- Keep yourself up to date on DDoS attack methods and defenses. New articles are routinely published and contain trends and other information to help you better plan and update your network defenses.
- Monitor your network for vulnerable systems. Running tools like Nessus and Nmap (see the “Recruitment” section) can help you quickly identify vulnerable systems on your network.
- Routinely scan your machines to make sure that they are not part of a botnet. It will not look good if an attack against one of your competitors is traced back to your machines.
- Enable logging and monitor log files for suspicious activity. There are tools to help you do this, called *intrusion detection systems* (IDS). These tools can also help you detect an attack and will be discussed in more detail in the section “IDS/IPS Systems.”
- Establish a routine for updating, scanning, and monitoring so that these activities are carried out routinely and regularly. Organizations such as CERT^Q and SANS^R publish guidelines on best practices. Make yourself familiar with these.

TIP

The “InfoSec” reading room on the SANS Institute site contains white papers on a variety of topics, including incident handling. See www.sans.org/. The CERT Web site has a section for system administrators that includes vulnerability information and freely available tools including AirCERT, which provides for automated incident reporting. See www.cert.org/.

WARNING

While it is a good idea to run a tool like Nessus on your own network, you should be careful when configuring a scan to avoid accidentally scanning outside your network. In some cases, your ISP may detect the scan and simply shut down your access. In other cases, like scanning your employer’s network, you may receive an unwanted visit from the IT department. Running scanners is typically against network policy; use them on your own network, only.

^Qwww.cert.org/

^Rwww.sans.org/

Strategy

The general life cycle of an incident response is to *protect* against attacks, *detect* attacks, and finally *react* to attacks.^S Each of these stages is essential to an overall strategy for dealing with DDoS attacks. The remaining sections of this chapter provide details about how to defend against DDoS attacks. Keep in mind that this is a rapidly changing environment, and that what works today may not work tomorrow. It is essential that you keep yourself up to date on DDoS defense.

Network Configuration

Understand your network and make sure that it is configured correctly. This is fundamental both to preventing DDoS attacks and to preventing your machines from being used to launch a DDoS attack. First, and perhaps simplest, block any unused ports at the network edge. In particular, ports 6665 through 6669 (tcp) have traditionally been used for IRC, though of course any port may be used.

One way to prevent your machines from being used in a DDoS attack (or at least to make it less likely to succeed) is to guard against address spoofing. Recall that every message has an embedded return address. It is typical to alter, or *spoof* this address when conducting a DDoS attack to make it harder to properly block incoming traffic, or to even determine the origin of the traffic at all.

Configure routers and firewalls at the edge of your network to filter *outgoing* packets from your network whose address field does not match your network's address. That is, if an outgoing packet has an address that is not from your network, discard the packet.

Some DDoS attacks use the special *broadcast* address. This is an address on a network that forwards messages to *all* hosts on the network. There is no legitimate reason someone outside your network will need to broadcast to all your hosts. Configure your firewall and routers to block any packets directed to the broadcast address. Likewise, there is no reason people on your network should be sending packets to the broadcast address of *another* network. Block outgoing packets destined for the broadcast address.

Finally, not all addresses are valid Internet addresses. If you have set up a home network, you may have used an address starting with 192.168. The class C address block 192.168.0.0/16 is a private address block; these addresses are never assigned to a host on the Internet. Likewise, 10.0.0.0/8 and 172.16.0.0/12 are class A and class B, respectively, private address blocks. Do not allow incoming or outgoing packets with either a source or destination address from one of these address blocks.^T There are special-use address ranges to avoid, too.^U For example, 127.0.0.0/8 is reserved for the “loopback” address, used by a network interface to refer to itself. A list of addresses to block is given in Table 1.1.

^SSee the CERT “Handbook for computer security incident response teams (CSIRTS),” available on the Web at the time of writing at www.cert.org/archive/pdf/csirt-handbook.pdf.

^TThese address blocks are defined in RFC-1918.

^USee RFC-3330.

Table 1.1 Addresses not on the Internet

Address	Use
0.0.0.0/32	See RFC-1700.
10.0.0.0/8	Private network addresses, as per RFC-1918.
127.0.0.0/8	Loopback addresses, as per RFC-1700.
169.254.0.0/16	“Link local” block. These addresses are typically assigned when another address cannot be obtained, such as when DHCP fails.
172.16.0.0/12	Private network addresses, as per RFC-1918.
192.0.2.0/24	Test network addresses. These addresses are for use in example code or documentation.
192.88.99.0/24	These addresses are reserved for relay anycast. See RFC-3068.
192.168.0.0/16	Private network addresses, as per RFC-1918.
192.18.0.0/15	Reserved for benchmarking tests of network devices, as per RFC-2544.
224.0.0.0/4	Reserved for IPv4 multicast. See RFC-3171.
240.0.0.0/4	The original class E address space is still reserved. This includes the special address 255.255.255.255 used for limited broadcast. See RFC-1700.

DDoS Appliances

DDoS appliances are network hardware with specialized firmware to detect anomalous network traffic and divert or drop that traffic to mitigate a DDoS attack. Many models also offer screening of *outbound* traffic, so that you aren’t the source of a DDoS attack. Cisco Systems, Inc.,^V RioRey, Inc.,^W Top Layer Security, Inc.,^X and others sell DDoS appliances.

These appliances rely on detecting anomalous network traffic. While it is easy for legitimate users to generate legitimate-looking traffic, automatically generated traffic tends to have statistical characteristics that make it easier to spot. As a simple example, if your server is receiving a series of requests for a login from a particular host, all faster than any human could type, this traffic is probably part of a DDoS attack and can be dropped.

This approach has limits. If you know the method used to detect the anomalous traffic, you can work around it and still achieve the desired effect. For example, with more machines you can send requests to log in more slowly from each machine, but maintain the same overall rate. One way to do this is to simply purchase your victim’s DDoS appliance, or network monitoring software, install it, and then craft your attack so that it is not detected. Despite these limits, deploying software or hardware to detect and block DDoS traffic can prevent many DDoS attacks.

^Vwww.cisco.com/

^Wwww.riorey.com/

^Xwww.toplayer.com/

IDS/IPS Systems

IDS observes traffic on a port and attempts to match it against known patterns corresponding to malware, port scanning, or DDoS attacks. A well-known open-source IDS is Snort.^Y Figure 1.4 shows the ACID Web interface to the Snort database. When Snort matches a pattern, it triggers an alert and can notify system administrators immediately. Additionally, it captures and logs traffic, which can help determine precisely what is happening.

EasyIDS is built around Snort and a few other tools, and packaged as a Linux distribution based on CentOS. It is installed on a single dedicated machine with two network interface cards (one facing the Internet, one facing your local network) to protect a network. This simplifies the installation, configuration, and maintenance of a network IDS (NIDS).

Bro is another open-source IDS.^Z Bro relies heavily on protocol analysis to detect abnormal traffic, while Snort relies on a simple signature-based matching. Both Bro and Snort can be used together, as they are complementary and they have the capability to execute actions when they detect an event of interest. For example, if your IDS detects that a host is scanning ports on your machine, it might write a rule to your firewall or router to block the scanning host's address. The use of an IDS to actively

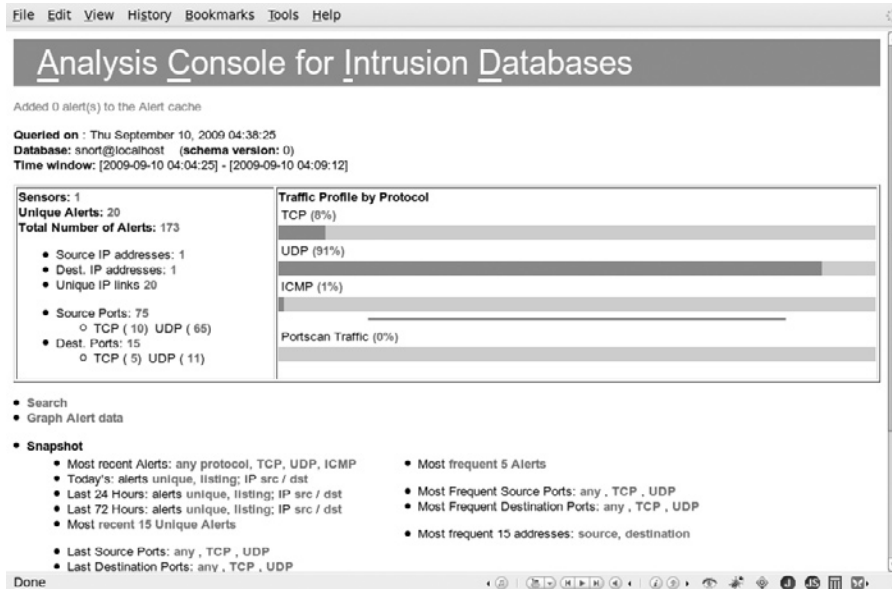


FIGURE 1.4
Displaying Intrusion Detection Data with ACID

^Ywww.snort.org/
^Zwww.bro-ids.org/

respond to intrusion attempts and block them transforms this system into one known as an intrusion prevention system (IPS).

Another strategy for detecting intrusion attempts is to create a *honeypot*. A honeypot is a carefully monitored machine or address that is used only to detect attacks. Normal, legitimate traffic is never directed to the honeypot machine, so any traffic that is detected at the honeypot is likely malicious traffic. Thus, by monitoring the honeypot, an IDS can detect when a network intrusion is being attempted. A simple example of a honeypot is an e-mail address created for the sole purpose of collecting spam. Since the e-mail address does not go to any legitimate recipient, any e-mail that arrives at the address is spam, and should be filtered from the rest of the e-mail. Project Honey Pot uses exactly this technique to monitor spammers.^{AA}

Reacting to DDoS Attacks

Once a DDoS attack is underway, you have several options. You can attempt to block the hosts generating the traffic. Because of spoofing and the large number of hosts used in a DDoS attack, this may be difficult.

To block hosts, you must first identify them. If you are running an IDS, you will already have a list of addresses provided by the tool. Otherwise, you have to capture network traffic and analyze it. To capture network traffic, you can use `tcpdump`.^{BB} This is a packet *sniffer* that can observe and record network traffic on an interface.

In the following example, we capture 1,000 packets using `tcpdump`.

```
$ tcpdump -c 1000 -w record.tcp eth0
tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture
size 96 bytes
1000 packets captured
1000 packets received by filter
0 packets dropped by kernel
```

An easier way to analyze network traffic is to use an actual network traffic analyzer, such as *Wireshark*.^{CC} *Wireshark* is available for most platforms and is relatively easy to use. Figure 1.5 shows packets captured using *Wireshark*.

It is also possible to configure some routers to provide this information. Cisco routers, for instance, keep a total of the number of times each rule is matched. In the following, we create a rule to match echo and one to match echo reply (pings).

```
access-list 169 permit icmp any any echo
access-list 169 permit icmp any any echo-reply
```

Including *log-input* at the end of a rule will also create a log of matching network traffic. Use `show log` to see the log. We can now get the number of times each rule is matched via the `show access-list` command, which gives the number of times each rule is matched by network traffic.

^{AA}<http://projecthoneypot.org/>

^{BB}While `tcpdump` is common on UNIX and Linux machines, on Windows you can use *WinDump*, an open-source clone of `tcpdump`.

^{CC}www.wireshark.org/

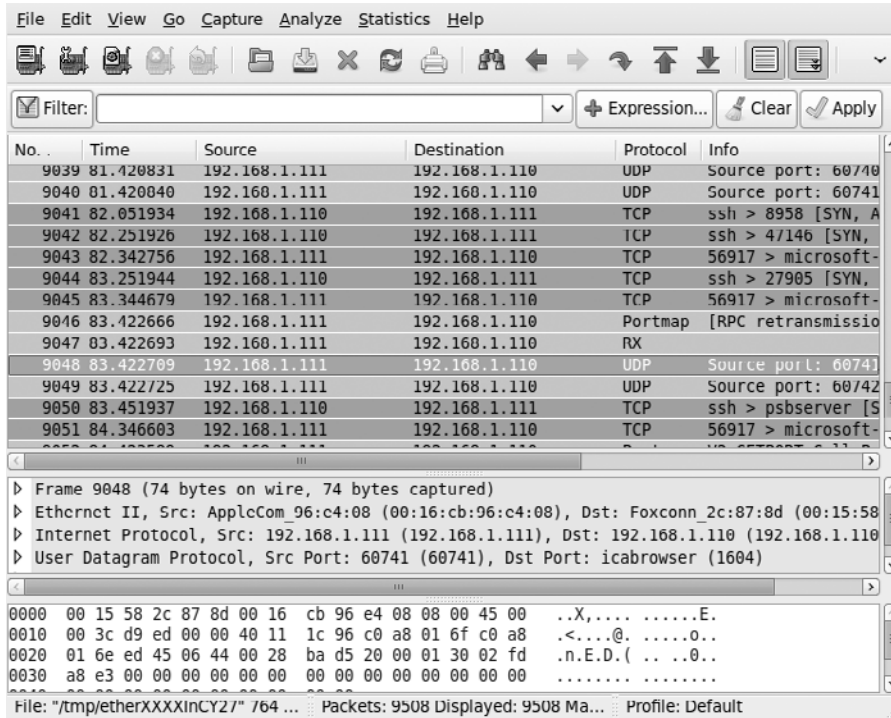


FIGURE 1.5

Wireshark

If you are being subjected to a ping flood or some other type of DDoS attack using a particular kind of traffic, one approach is to *rate limit* that kind of traffic. Rate limiting sets a limit as to the amount of bandwidth a particular kind of traffic can consume. When the threshold is exceeded, packets are simply dropped. This solves the problem of a ping flood, but if the DDoS traffic looks like legitimate traffic this may not be an option.^{DD}

Over-Provisioning and Adaptive Provisioning

The DDoS attack will only succeed if it can overwhelm your network bandwidth or your servers. One way to prevent a DDoS attack is simply to increase your network and server capacity beyond what is necessary for legitimate traffic. This extra capability can be brought online if a DDoS attack is detected. This *over-provisioning* can be expensive, however.

^{DD}Network Security Technologies and Solutions (CCIE Professional Development Series) by Yusuf Bhajji, Cisco Press, 2008, discusses how to properly configure routers to filter or throttle ICMP traffic. The United States National Security Agency (NSA) also publishes guides to router security configuration. See www.nsa.gov/ia/guidance/security_configuration_guides/.

An alternative if you use an ISP is to purchase additional capacity for the duration of the DDoS attack. This adaptive provisioning can be provided via *burstable circuits*, which can carry additional capacity if necessary. You typically pay a rate based on your *average* use of the connection. Since attackers often “tune” the attack while it is in progress, you will still need to carefully monitor the attack to determine how best to respond. For example, an attack may change from trying to overload a server to exploiting a network protocol.

THE FUTURE OF DENIAL OF SERVICE

Malware is now big business, and operating botnets has become a source of significant income. Developing a large botnet is a financial investment. Likewise, considerable economic activity is moving onto the Internet, with some companies completely dependent on the Internet for their business, including Google and Amazon.com. All this creates a marketplace to supply more advanced attack and defense capabilities.

Attack

Based in St. Petersburg, the Russian Business Network (RBN) is a hosting organization that offers “bulletproof” hosting and a variety of other services, including botnet rental, so you don’t even need to create your own botnet.^{EE} If you can lease time on an existing botnet, and use it effectively for a DDoS attack, you have successfully reduced a technical challenge to a simple matter of paying some money.

This is probably the shape of the future in DDoS. Creating and controlling a large botnet is still somewhat challenging, and also presents legal risks. It is also the case that botnets can be “hijacked” if someone else, perhaps another criminal enterprise, gets access to the command and control channels.

Current trends show that the malware world is becoming less vertically integrated and more specialized. Many components of successful malware are not, by themselves, illegal. It is thus perfectly reasonable to set up shop and develop and sell software to enable DDoS attacks, so long as the software has a potential legal application, as well. For example, software used to protect intellectual property and thwart reverse engineering can also be used to harden malware against reverse engineering, making it hard to discover just what a particular kind of malware is doing, and how to counteract it. Digital rights management (DRM) techniques can cloak a malware payload until the time arrives to launch a DDoS attack. While the tools to conduct a DDoS attack are improving, fortunately so are the tools to defend against an attack.

^{EE}RBN is the originator of the MPack malware kit, which you can purchase for between \$500 and \$1000 (at the time of writing). Included with this kit is technical support and regular updates to the vulnerabilities it can exploit. Additional functionality can be purchased to target specific vulnerabilities or to cloak the software against antimalware and antivirus programs. MPack includes a Web-based management console to allow you to track its progress as it infects machines.

Defense

One possibility may be to structure the defense in the same way as the offense. By connecting several nodes using peer-to-peer networks and exchanging data about network traffic, it may be possible to automatically detect and mitigate DDoS attacks in real time.^{FF}

Even if one is correctly filtering the malicious traffic from a DDoS attack a network segment may become overloaded, since the filtering is being done locally. By adding additional layers of protocol to the Internet it may be possible to distribute the defense against a DDoS attack. In this case when a DDoS attack is detected, information about the attack is pushed back so that routers close to the attack sources perform the filtering. This effectively distributes the defense against the attack by recruiting routers near the sources.^{GG}

Right now a key aspect of DDoS is that it is computationally easy to generate the traffic. One potential way to make DDoS harder is to impose additional overhead on the originator of the traffic. This approach uses “puzzles” that must be solved before a connection is allowed. This reduces the rate at which attack sources can generate traffic, and reduces the overall effectiveness of the DDoS attack.

One approach that has been proposed for dealing with DDoS extortion is to legally require companies to report both threats and attacks to law enforcement, and to impose severe penalties for not reporting. The idea is that forcing these attacks into the open and involving law enforcement early raises the stakes for would-be extortionists.

SUMMARY

This chapter showed you that DDoS attacks can do significant harm. By recruiting many machines into a botnet and directing a flood of traffic against a victim, one can overwhelm the victim and degrade, deny, or in some cases even destroy the service. We explained what DoS and DDoS attacks are and how a DDoS attack is launched, as well as providing an understanding of why DDoS attacks are so hard to defend against. However, you should also have the capability of building a defensive strategy based on the framework of tools and methods that we discussed.

Endnotes

1. Richards J. Denial of service: the Estonian Cyberwar and its implications for U.S. national security. *Int Aff Rev* 2009; 18(1).
2. Clayton R. The rising tide: DDoS by defective designs and defaults. Proceedings of the 2nd Conference on Steps to Reducing Unwanted Traffic on the Internet (SRUTI'06). Berkeley, California; 2006.

^{FF}For one such approach see Guangsen Zhang and Manish Parashar, “Cooperative Defence Against DDoS Attacks,” *Journal of Research and Practice in Information Technology*, vol. 38, no. 1, 2006.

^{GG}See Katerina Argyraki and David Cheriton, “Active Internet Traffic Filtering: Real-Time Response to Denial of Service Attacks,” presented at the USENIX Annual Technical Conference, 2005.