Chapter 2

Botnets Overview

> If only it were possible to reproduce yourself a million times over so that you can achieve a million times more than you can today.
>
> *—Dr. Joseph Goebbels, Propaganda Minister for Nazi Germany; from the 15 Feb 1943 entry in his personal diary.*

Solutions in this chapter:

- **What Is a Botnet?**
- **The Botnet Life Cycle**
- **What Does a Botnet Do?**
- **Botnet Economics**

☑ **Summary**

☑ **Solutions Fast Track**

☑ **Frequently Asked Questions**

What Is a Botnet?

What makes a botnet a botnet? In particular, how do you distinguish a botnet client from just another hacker break-in? First, the clients in a botnet must be able to take actions on the client without the hacker having to log into the client's operating system (Windows, UNIX, or Mac OS). Second, many clients must be able to act in a coordinated fashion to accomplish a common goal with little or no intervention from the hacker. If a collection of computers meet this criteria it is a botnet.

A *botnet* is the melding of many threats into one. The typical botnet consists of a bot server (usually an IRC server) and one or more botclients (refer to Figure 1.2). Botnets with hundreds or a few thousands of botclients (called zombies or drones) are considered small botnets. In this typical botnet, the botherder communicates with botclients using an IRC channel on a remote command and control (C&C) server. In step 1, the new botclient joins a predesignated IRC channel on an IRC server and listens for commands. In step 2, the botherder sends a message to the IRC server for each client to retrieve. In step 3, the clients retrieve the commands via the IRC channel and perform the commands. In step 4, the botclients perform the commands—in the case of Figure 1.2, to conduct a DDoS attack against a specified target. In step 5, the botclient reports the results of executing the command.

This arrangement is pleasing to hackers because the computer performing the actions isn't their computer and even the IRC relay isn't on their computer. To stop the botnet the investigator has to backtrack from a client to an IRC server to the hackers. The hacker can add another layer of complexity by sending all commands to the IRC channel through an obfuscating proxy and probably through a series of multiple hops, using a tool like Tor (http://tor.eff.org/download.html.en). Having at least one of these elements in another country also raises the difficulty of the investigation. If the investigator is charged with protecting one or more of the botnet clients, they will usually stop the investigation once they realize the individual damage to their enterprise is low, at least too low to justify a complex investigation involving foreign law enforcement. Add to this the fact that some botnet codebases include commands to erase evidence, commands to encrypt traffic, and even polymorphic stealth techniques, and it's easy to see why hackers like this kind of tool. Modern botnets are being fielded that are organized like real armies, with divisions of zombies controlled by different bot servers. The botherder controls a set of bot servers, which in turn each control a division of zombies. That way, if a communications channel is disrupted, only one division is lost. The other zombie divisions can be used to retaliate or to continue to conduct business.

The Botnet Life Cycle

Botnets follow a similar set of steps throughout their existence. The sets can be characterized as a life cycle. Figure 2.1 illustrates the common life cycle of a botnet client. Our understanding of the botnet life cycle can improve our ability to both detect and respond to botnet threat.

Exploitation

The life of a botnet client, or botclient, begins when it has been exploited. A prospective botclient can be exploited via malicious code that a user is tricked into running; attacks against unpatched vulnerabilities; backdoors left by Trojan worms or remote access Trojans; and password guessing and brute force access attempts. In this section we'll discuss each of these methods of exploiting botnets.

Malicious Code

Examples of this type of exploit include the following:

- Phishing e-mails, which lure or goad the user to a Web site that installs malicious code in the background, sometimes while convincing you to give them your bank userid and password, account information, and such. This approach is very effective if you are looking for a set of botnet clients that meet certain qualifications, such as customers of a common bank.
- Enticing Web sites with Trojan code ("Click here to see the Dancing Monkeys!").
- E-mail attachments that when opened, execute malicious code.

- Spam in instant messaging (SPIM). An instant message is sent to you by someone you know with a message like "You got to see this!" followed by a link to a Web site that downloads and executes malicious code on your computer.

Attacks against Unpatched Vulnerabilities

To support spreading via an attack against unpatched vulnerabilities, most botnet clients include a scanning capability so that each client can expand the botnet. These scanning tools first check for open ports. Then they take the list of systems with open ports and use vulnerability-specific scanning tools to scan those systems with open ports associated with known vulnerabilities. Botnets scan for host systems that have one of a set of vulnerabilities that, when compromised, permit remote control of the vulnerable host. A fairly new development is the use of Google to search for vulnerable systems.

Every "Patch Tuesday" from Microsoft is followed by a flurry of reverse engineering in the hacker community. Within a few days (3 for the last patch Tuesday), someone will release an exploit against the problem that the most recent patch fixed. The hacker community is counting on millions of users that do not update their computers promptly. Modular botnets are able to incorporate new exploits in their scanning tools almost overnight. Diligent patching is the best prevention against this type of attack. If it involves a network protocol that you don't normally use, a host-based firewall can protect you against this attack vector. However, if it is a protocol that you must keep open you will need intrusion detection/protection capabilities. Unfortunately there is usually a lag of some time from when the patch comes out until the intrusion detection/protection updates are released. Your antivirus software may be able to detect the exploit after it happens, if it detects the code before the code hides from the A/V tool or worse, turns it off.

Vulnerabilities Commonly Exploited by Bots:

Agobot spreads via several methods including:

- Remote Procedure Call (RPC) Distributed Component Object Model (DCOM) (TCP ports 135, 139, 445, 593, and others) to XP systems

- RPC Locator vulnerability
- File shares on port 445
- If the target is a Web server, the IIS5 WEBDAV (Port 80) vulnerability

SDBot Spreads through the following exploits:

- NetBios (port 139)
- NTPass (port 445)
- DCom (ports 135, 1025)
- DCom2 (port 135)
- MS RPC service and Windows Messenger port (TCP 1025)
- ASN.1 vulnerability, affects Kerberos (UDP 88), LSASS.exe and Crypt32.dll (TCP ports 135, 139, 445), and IIS Server using SSL
- UPNP (port 5000)
- Server application vulnerabilities
- WebDav (port 80)
- MSSQL (port 1433)
- Third-party application vulnerabilities such as DameWare remote management software (port 6129) or Imail IMAPD Login username vulnerability (port 143)
- A CISCO router vulnerability such as CISCO IOS HTTP authorization (Port 80) vulnerability

IRCBot, Botzori, Zotob, Esbot, a version of Bobax, and a version of Spybot attempt to spread by exploiting the Microsoft Plug and Play vulnerability (MS 05-039).

Backdoors Left by Trojan Worms or Remote Access Trojans

Some botnets look for backdoors left by other bits of malicious code like Remote Access Trojans. Remote Access Trojans include the ability to control another computer without the knowledge of the owner. They are easy to use so many less skilled users deploy them in their default configurations. This means that anyone that knows the default password can take over the Trojan'ed PC.

SDBot exploits the following backdoors:

- Optix backdoor (port 3140)
- Bagle backdoor (port 2745)
- Kuang backdoor (port 17300)
- Mydoom backdoor (port 3127)
- NetDevil backdoor (port 903)
- SubSeven backdoor (port 27347)

Password Guessing and Brute-Force Access Attempts

RBot and other bot families employ several varieties of password guessing. According to the Computer Associates Virus Information Center, RBot spreading is started manually through remote control. It does not have an automatic built-in spreading capability. RBot starts by trying to connect to ports 139 and 445. If successful, RBot attempts to make a connection to the windows share (\\<target>\ipc$), where the target is the IP address or name of the potential victim's computer.

If unsuccessful, the bot gives up and goes on to another computer. It may attempt to gain access using the account it is using on the attacking computer. Otherwise it attempts to enumerate a list of the user accounts on the computer. It will use this list of users to attempt to gain access. If it can't enumerate a list of user accounts it will use a default list that it carries (see the sidebar). This information is valuable to the CISO trying to identify and remove botclients in their environment. The login attempts are recorded in the workstation event logs. These will appear different from normal logins in that the workstation name will not be the local machine's name. In a later chapter we will discuss how this information can be used to trace back to many other members of the same botnet.

Notes from the Underground…

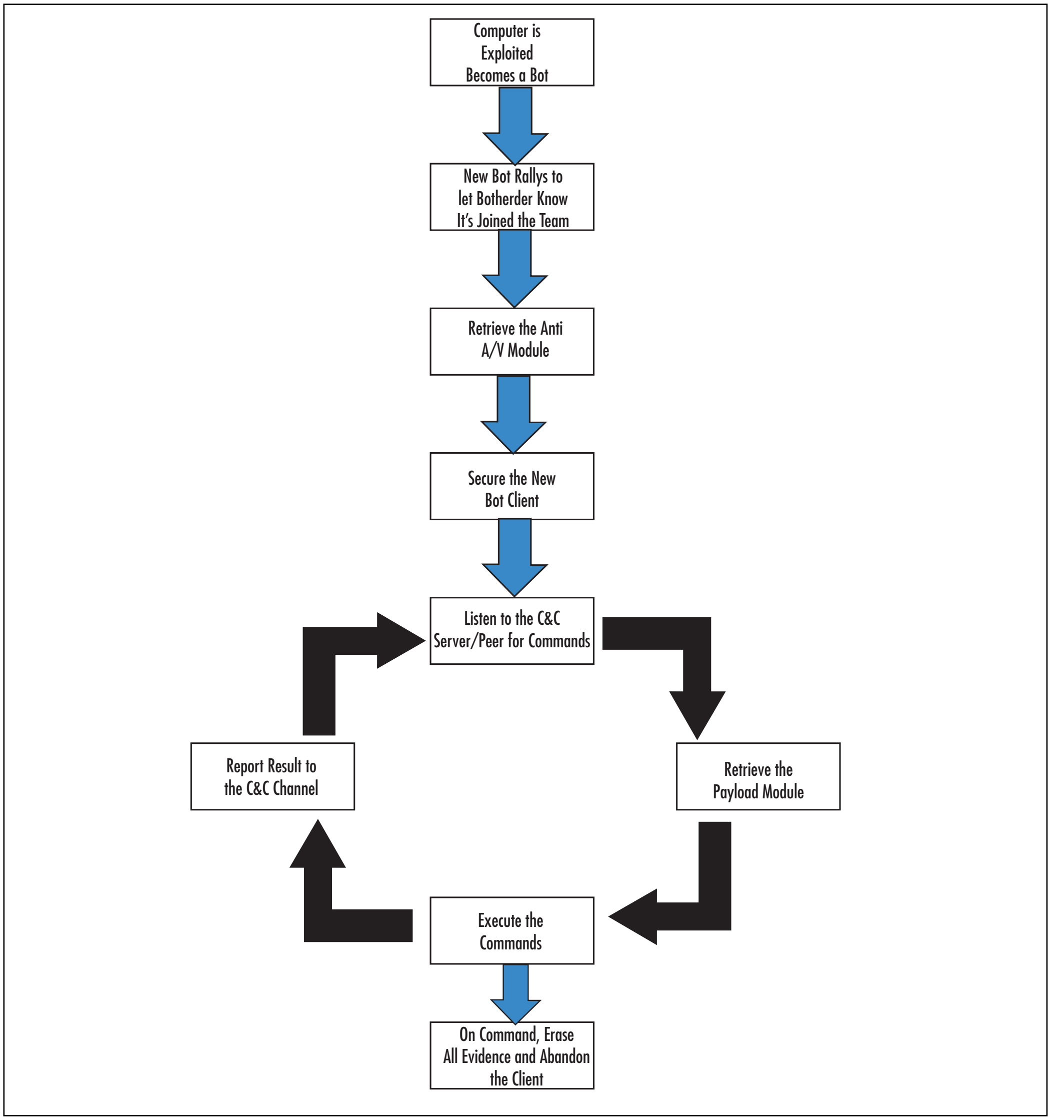
Default UserIDs Tried by RBot

Here is a list of default userids that RBot uses.

- Administrator
- Administrador
- Administrateur
- administrat
- admins
- admin
- staff
- root
- computer
- owner
- student
- teacher
- wwwadmin
- guest
- default
- database
- dba
- oracle
- db2

The passwords used with these attempts can vary. There is a default list provided, but the botherder can replace it and the userID list with userIDs and passwords that have worked on other computers in the enterprise.

Figure 2.1 The Botnet Life Cycle


Computer is Exploited Becomes a Bot
New Bot Rallys to let Botherder Know It's Joined the Team
Retrieve the Anti A/V Module
Secure the New Bot Client
Listen to the C&C Server/Peer for Commands
Retrieve the Payload Module
Execute the Commands
Report Result to the C&C Channel
On Command, Erase All Evidence and Abandon the Client

Rallying and Securing the Botnet Client

Although the order in the life cycle may vary, at some point early in the life of a new botnet client it must call home, a process called "rallying." When rallying, the botnet client initiates contact with the botnet Command and Control (C&C) Server. Currently, most botnets use IRC for Command and Control. In this chapter we will cover IRC C&C. In the next chapter we will describe advanced C&C methods, such as using Peer-to-Peer protocols. The phrase "Command and Control" is the term given to the act of managing and tasking the botnet clients. Rallying is the term given for the first time a botnet client logins in to a C&C server. The login may use some form of encryption or authentication to limit the ability of others to eavesdrop on the communications. Some botnets are beginning to encrypt the communicated data.

At this point the new botnet client may request updates. The updates could be updated exploit software, an updated list of C&C server names, IP addresses, and/or channel names. This will assure that the botnet client can be managed and can be recovered should the current C&C server be taken offline.

The next order of business is to secure the new client from removal. The client can request location of the latest anti-antivirus (Anti-A/V) tool from the C&C server. The newly controlled botclient would download this software and execute it to remove the A/V tool, hide from it, or render it ineffective. The following list contains a batch file, used by an Rbot client, to shut off antivirus clients. An Rbot gains its access by password guessing or by a brute force attack against a workstation. Once Rbot has guessed or sniffed the password for a local administrator account, it can login to the computer as a legitimate local administrator. An instance of Rbot has been found that runs a bat file that file executes net commands to turn off various A/V applications.

```
net start >>starts
net stop "Symantec antivirus client"
net stop "Symantec AntiVirus"
net stop "Trend NT Realtime Service"
net stop "Symantec AntiVirus"
net stop "Norton antivirus client"
net stop "Norton antivirus"
net stop "etrust antivirus"
net stop "network associate mcshields"
net stop "surveyor"
```

Shutting off the A/V tool may raise suspicions if the user is observant. Some botclients will run a dll that neuters the A/V tool. With an Anti-A/V dll in place the A/V tool may appear to be working normally except that it never detects or reports the files related to the botnet client. It may also change the Hosts file and LMHosts file so that attempts to contact an A/V vendor for updates will not succeed. Using this method, attempts to contact an A/V vendor can be redirected to a site containing malicious code or can yield a "website or server not found" error.

Increasingly, botnet clients have also employed a rootkit or individual tools to try to hide from the OS and other applications that an IT professional might use to detect them. Consequently, some botnet clients scan for rootkits using the Rootkit Revealer from www.sysinternals.com or rkdetector from http://www.rkdetector.com, to check to see if the computer already has a rootkit. One tool, hidden32.exe, is used to hide applications that have a GUI interface from the user. Its use is simple; the botherder creates a batch file that executes hidden32 with the name of the executable to be hidden as its parameter. Another stealthy tool, HideUserv2, adds an invisible user to the administrator group.

Another common task for this phase is that of mundane organization and management. After securing the computer against antivirus tools, previous hackers, and detection by the user, the botherder might check to see what else might be here. In the case of our Rbot infection, the botherder used a batch file called find.bat, which tells the botherder if another hacker had been there before or where he or she put his or her tools on this client. It may also tell the botherder about things on the computer that could be useful. For some payloads it is useful to categorize a client according to hard drive space, processor speed, network speed to certain destinations, and so forth. For this task, our example botnet used a batch file to launch a series of utilities and concatenate the information into a text file (see the sidebar titled "A Batch File Used to Discover the Nature of a New Botnet Client").

Tools & Traps…

A Batch File Used to Discover the Nature of a New Botnet Client

```
@echo off

echo *---------------------------------------------------------------
----*>info.txt

echo *--Computer Specs....
--*>>info.txt

echo *---------------------------------------------------------------
----*>>info.txt

psinfo.exe -d >>info.txt

Diskinfo

echo *---------------------------------------------------------------
----*>>info.txt

echo *--List of Current Processes Running....
--*>>info.txt

echo *---------------------------------------------------------------
----*>>info.txt

fport.exe /ap >>info.txt

echo *---------------------------------------------------------------
----*>>info.txt

echo *--List of Current Running/Stopped Services..
--*>>info.txt

echo *---------------------------------------------------------------
----*>>info.txt

xnet.exe list >>info.txt

echo *---------------------------------------------------------------
----*>>info.txt

echo *--List of Whois Info..
--*>>info.txt

echo *---------------------------------------------------------------
----*>>info.txt

echo *--                 Lista uruchomionych procesów
--*>>info.txt
```

Continued

```
echo *---------------------------------------------------------------
----*>>info.txt
pslist.exe >>info.txt
echo *---------------------------------------------------------------
----*>>info.txt
Password.exe >>info.txt
echo *---------------------------------------------------------------
----*>>uptime.txt
uptime.exe /s>>uptime.txt
echo *---------------------------------------------------------------
----*>>uptime.txt
hidden32.exe find.bat
echo *---------------------------------------------------------------
----*>>info.txt
rkdetector.exe >>rk.txt
hidden32.exe pass.bat
hidden32.exe pwdump2.bat


cls
echo Whoami >> info.txt
echo. >> info.txt
echo Computer Name= %COMPUTERNAME% >> info.txt
echo Login Name=    %USERNAME% >> info.txt
echo Login Domain=  %USERDOMAIN% >> info.txt
echo Logon Server=  %LOGONSERVER% >> info.txt
echo. >> info.txt
echo Home Drive=    %HOMEDRIVE% >> info.txt
echo Home Share=    %HOMESHARE% >> info.txt
echo System Drive=  %SYSTEMDRIVE% >> info.txt
echo System Root=   %SYSTEMROOT% >> info.txt
echo Win Directory= %WINDIR% >> info.txt
echo User Profile Path= %USERPROFILE% >> info.txt
echo. >> info.txt
echo Groups user belongs to: >> info.txt
echo. >> info.txt
.\whoami.exe /user /groups /fo list >> info.txt
```

Continued

```
iplist.exe >> info.txt
FHS.exe >> info.txt
```

The botnet also took the opportunity to start its rootkit detector and hide and launch the password collection programs.

Waiting for Orders and Retrieving the Payload

Once secured, the botnet client will listen to the C&C communications channel. In this overview, we are describing botnets that are controlled using IRC channels. In the following chapter we will describe alternative C&C technologies.

Each botnet family has a set of commands that it supports. For example the SDBot supports the commands in Table 2.1, among others (adapted from the Know Your Enemy series, "Tracking Botnets—Botnet Commands" by the Honeynet Project).

Table 2.1 Botnet Command Examples

Function	Command Code
Recruiting	(scanalllsa)
	(scanstatslstats)
	scandel [portlmethod] —[method] can be one of a list of exploits including lsass, mydoom, DameWare, etc.
	scanstop
	(advscanlasc) [portlmethod] [threads] [delay] [minutes]
Downloading and updating	(updatelup) [url] [botid]
	(downloadldl) [url] [[runfile?]] [[crccheck]] [[length]]
Execute programs locally	(executele) [path]
	(findfilelff) filename
	(renamelmv) [from] [to]

Continued

Table 2.1 continued Botnet Command Examples

Function	Command Code
	findfilestopp
DDoS	syn [ip] [port] [secondslamount] [sip] [sport] [rand]
	udp [host] [num] [size] [delay] [[port]]size)
	ping [host] [num] [size] [delay]num

There are more details about IRC C&C in Chapter 8.

The botnet client will then request the associated payload. The payload is the term I give the software representing the intended function of this botnet client. Note from the diagram in Figure 2.1 that the function can change at any time. This is the beauty of a modular design. Updates can be sent prior to the execution of any assigned task. The primary function of the botnet client can be changed simply by downloading new payload software, designating the target(s), scheduling the execution, and the desired duration of the action. The next few paragraphs will describe some of these potential payloads.

What Does a Botnet Do?

A botnet is a collection of networked computers. They can do anything you can imagine doing with a collection of networked computers. The next few topics describe some of the uses of botnets that have been documented to date.

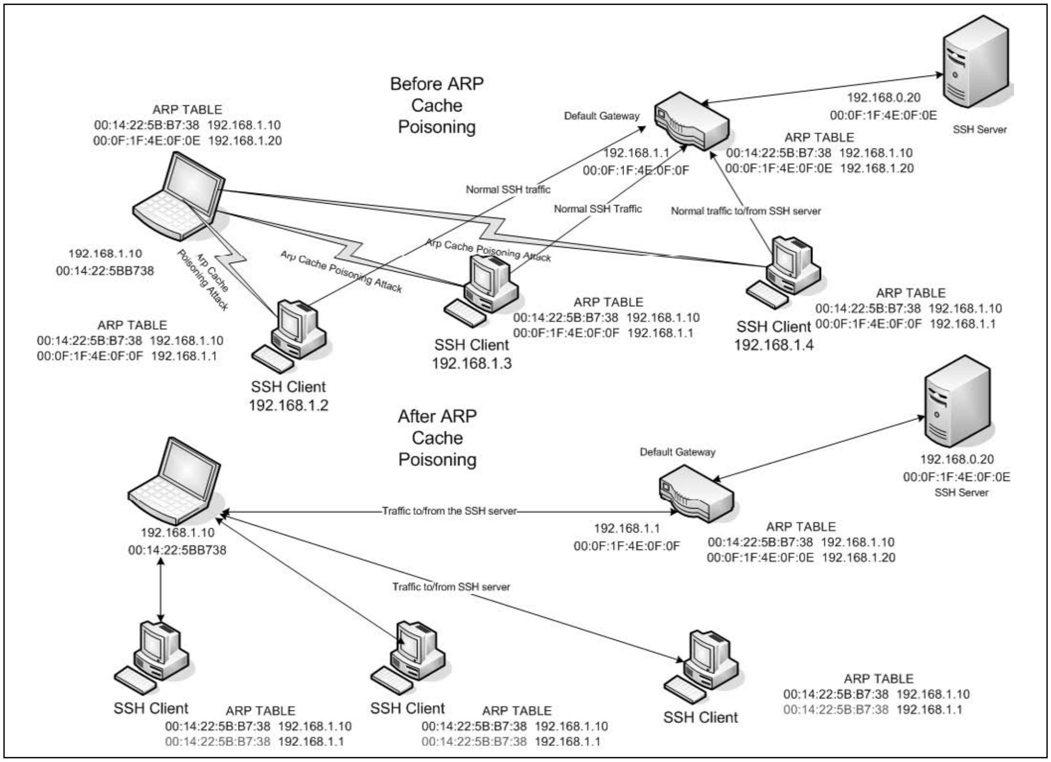
Recruit Others

The most basic thing each botclient does is to recruit other potential botclients. The botclient may scan for candidate systems. Rbot, for example, exploits Windows shares in password guessing or brute force attacks so its botclients scan for other systems that have ports 139 or 445 open, using tools like smbscan.exe, ntscan.exe, or scan500.exe. It also used the net command (net view /DOMAIN and net view /DOMAIN:<*domain name*>) to list NetBIOS names of potential candidate clients.

The botclient may be equipped to sniff network traffic for passwords. The clients use small, specialized password grabbers that collect only enough of the traffic to grab the username and password data. They may harvest encrypted forms of passwords in the SAM cache using a program like pwdump2, 3, or 4 and use SAM password crackers like Lopht Crack to break them. For some encrypted password data, they reformat the password data into a UNIX-like password file and send it to another, presumably faster, computer to brute force.

When the botherder discovers a botclient that uses encrypted traffic to a server, he or she may include a tool, such as Cain and Abel, to perform man-in-the-middle (MITM) attacks as part of the payload. In the MITM attack (see Figure 2.2), the botclient convinces other computers on its subnet that it is actually the default gateway through Arp cache poisoning, and then relays any data it receives to the actual gateway.

Figure 2.2 Arp Cache Poisoning for MITM Attacks



Before ARP Cache Poisoning
ARP TABLE
00:14:22:5B:B7:38 192.168.1.10
00:0F:1F:4E:0F:0E 192.168.1.20
Default Gateway
192.168.1.1
00:0F:1F:4E:0F:0F
192.168.0.20
00:0F:1F:4E:0F:0E
SSH Server
ARP TABLE
00:14:22:5B:B7:38 192.168.1.10
00:0F:1F:4E:0F:0E 192.168.1.20
Normal SSH traffic
Normal SSH Traffic
Normal traffic to/from SSH server
192.168.1.10
00:14:22:5BB738
Arp Cache Poisoning Attack
Arp Cache Poisoning Attack
Arp Cache Poisoning Attack
ARP TABLE
00:14:22:5B:B7:38 192.168.1.10
00:0F:1F:4E:0F:0F 192.168.1.1
SSH Client
192.168.1.2
ARP TABLE
00:14:22:5B:B7:38 192.168.1.10
00:0F:1F:4E:0F:0F 192.168.1.1
SSH Client
192.168.1.3
SSH Client
192.168.1.4
ARP TABLE
00:14:22:5B:B7:38 192.168.1.10
00:0F:1F:4E:0F:0F 192.168.1.1
After ARP Cache Poisoning
Default Gateway
192.168.0.20
00:0F:1F:4E:0F:0E
SSH Server
Traffic to/from the SSH server
192.168.1.10
00:14:22:5BB738
192.168.1.1
00:0F:1F:4E:0F:0F
ARP TABLE
00:14:22:5B:B7:38 192.168.1.10
00:0F:1F:4E:0F:0E 192.168.1.20
Traffic to/from SSH server
SSH Client
ARP TABLE
00:14:22:5B:B7:38 192.168.1.10
00:14:22:5B:B7:38 192.168.1.1
SSH Client
ARP TABLE
00:14:22:5B:B7:38 192.168.1.10
00:14:22:5B:B7:38 192.168.1.1
SSH Client
ARP TABLE
00:14:22:5B:B7:38 192.168.1.10
00:14:22:5B:B7:38 192.168.1.1

At the time of this writing, Cain included the capabilities to sniff all traffic from the subnet outbound, intercept and decrypt (through the MITM attack) SSH-1, HTTPS, RDP, and others, as well as searching for and cracking passwords in caches and files on the host computer. See the following sidebar for a list of the output files collected by the hacker tool Cain and ABEL. What's that? You don't run SSH-1? That's okay; Cain will negotiate with your clients to get them to switch to SSH-1. The CERT.lst file contains copies of fake Certs Cain creates on the fly when a workstation tries to go to a Web site that uses Certificates. The VOIP file is interesting in that it contains the names of .wav files containing actual conversations it recorded. For a detailed description of cracking password files with Cain, see www.rainbowtables.net/tutorials/cryptanalisys.php. Rainbowtables.net is a Web site that sells additional rainbow tables for use with Cain. Rainbow tables are tables of already cracked hashes. According to the Rainbowtables.net Web site, using their tables and others on the Internet "it is possible to crack almost any password under 15 characters using a mixed alphanumeric combination with symbols for LM, NTLM, PIX Firewall, MD4, and MD5." Their market spiel says, "hackers have them and so should you."

Are You Owned?

Cain Collection Files

Cain uses the following collection files:

- 80211.LST
- APOP-MD5.LST
- APR.LST
- CACHE.LST
- CCDU.LST
- CERT.LST
- CRAM-MD5.LST
- DICT.LST

Continued

- DRR.LST
- FTP.LST
- HOSTS.LST
- HTTP.LST
- HTTPS.LST
- HTTP_PASS_FIELDS.LST
- HTTP_USER_FIELDS.LST
- ICQ.LST
- IKE-PSK.LST
- IKEPSKHashes.LST
- IMAP.LST
- IOS-MD5.LST
- K5.LST
- KRB5.LST
- LMNT.LST
- MD2.LST
- MD4.LST
- MD5.LST
- MSSQLHashes.LST
- MySQL.LST
- MySQLHashes.LST
- NNTP.LST
- NTLMv2.LST
- ORACLE.LST
- OSPF-MD5.LST
- PIX-MD5.LST
- POP3.LST
- PWLS.LST
- QLIST.LST
- RADIUS.LST
- RADIUS_SHARED_HASHES.LST
- RADIUS_USERS.LST

Continued

- RDP.LST
- RIP-MD5.LST
- RIPEMD-160.LST
- SHA-1.LST
- SHA-2.LST
- SIP.LST
- SIPHASHES.LST
- SMB.LST
- SMTP.LST
- SNMP.LST
- SSH-1.LST
- TDS.LST
- TELNET.LST
- VNC-3DES.LST
- VNC.LST
- VoIP.LST
- VRRP-HMAC.LST

DDoS

The earliest malicious use of a botnet was to launch Distributed Denial of Service attacks against competitors, rivals, or people who annoyed the both-erder. You can see a typical botnet DDoS attack in Figure 2.3. The sidebar, "A Simple Botnet" in Chapter 1 describes the play-by-play for the DDoS. The actual DDoS attack could involve any one of a number of attack technologies, for example TCP Syn floods or UDP floods.

In order to understand how a TCP Syn Flood works you first have to understand the TCP connection handshake. TCP is a connection-oriented protocol. In order to establish a connection, TCP sends a starting synchronization (SYN) message that establishes an initial sequence number. The receiving party acknowledges the request by returning the SYN message and also includes an acknowledgement message for the initial SYN. The sending party increments the acknowledgment number and sends it back to the receiver. Figure 2.4 illustrates the TCP three-way handshake.

Figure 2.3 A DDoS Attack

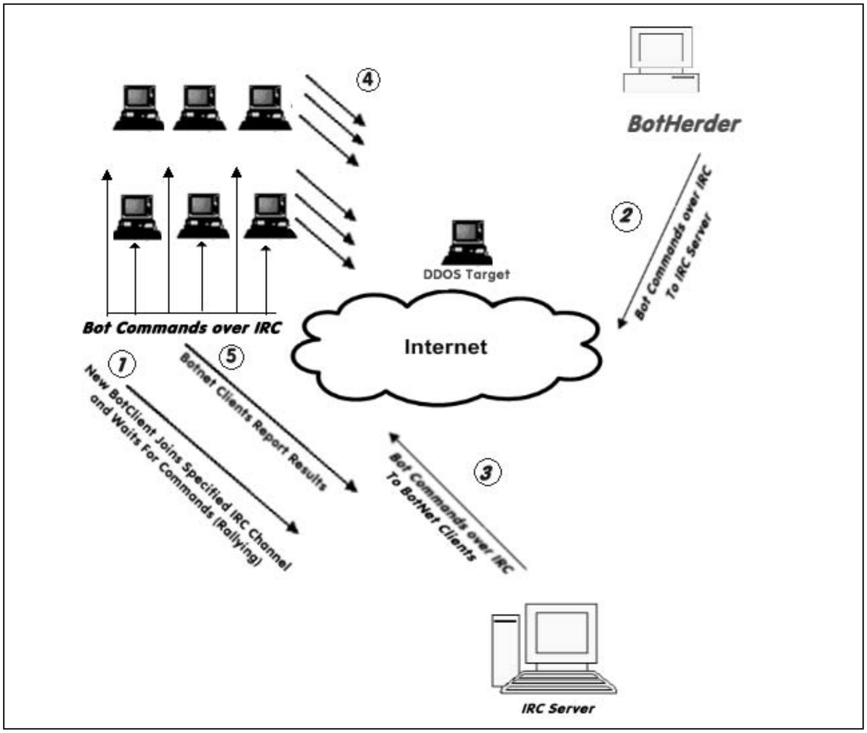

4
BotHerder
2
Bot Commands over IRC To IRC Server
DDOS Target
Internet
Bot Commands over IRC
1
5
New BotClient Joins Specified IRC Channel and Waits For Commands (Rallying)
Botnet Clients Report Results
3
Bot Commands over IRC To BotNet Clients
IRC Server

Figure 2.4 A TCP Connection Handshake

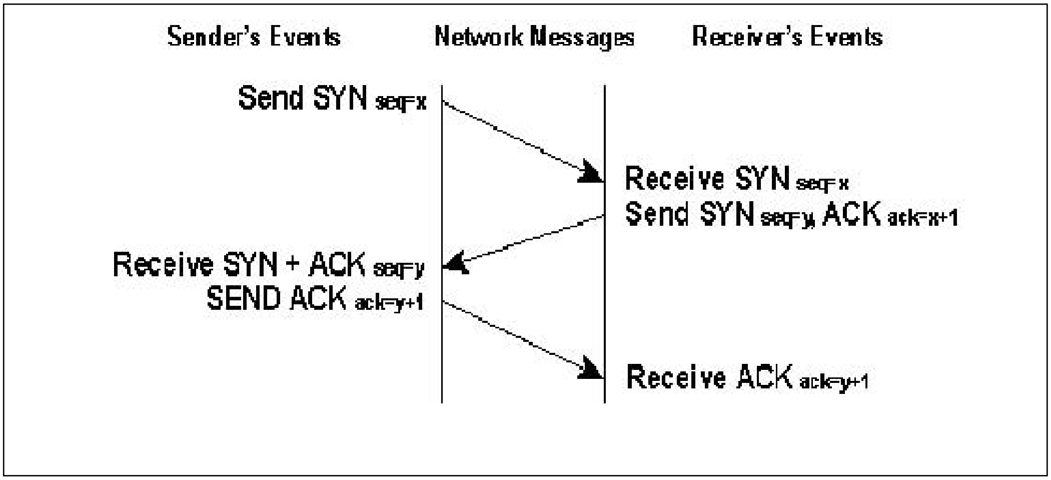
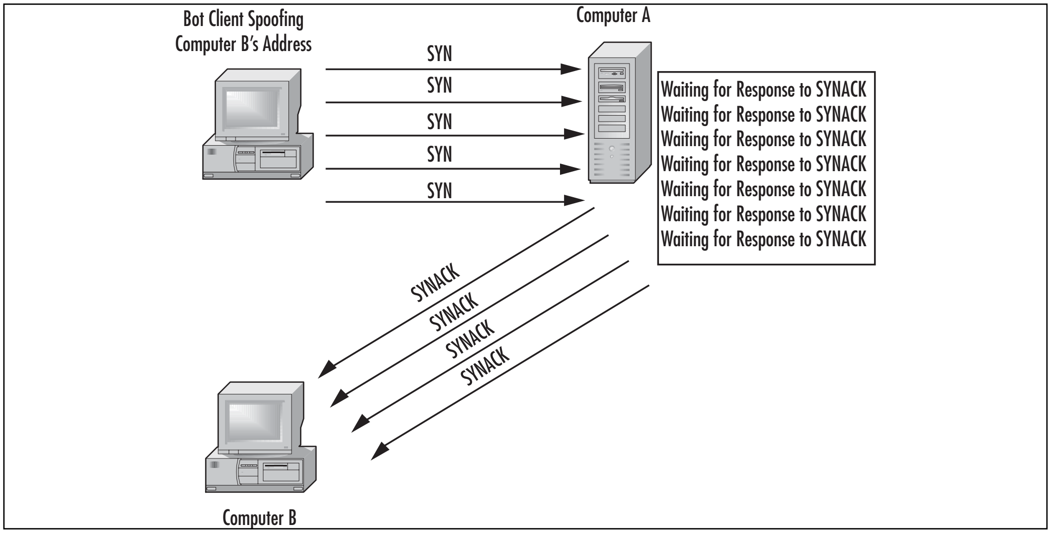


Figure 2.5 illustrates a SYN Flood attack. A SYN flood attacker sends just the SYN messages without replying to the receiver's response. The TCP specification requires the receiver to allocate a chunk of memory called a control block and wait a certain length of time before giving up on the connection. If the attacker sends thousands of SYN messages the receiver has to queue up the messages in a connection table and wait the required time before clearing them and releasing any associated memory. Once the buffer for storing these SYN messages is full, the receiver may not be able to receive any more TCP messages until the required waiting period allows the receiver to clear out some of the SYNs. A SYN flood attack can cause the receiver to be unable to accept any TCP type messages, which includes Web traffic, FTP, Telnet, SMTP, and most network applications.

Figure 2.5 SYN Flood Example


Bot Client Spoofing Computer B's Address
Computer A
SYN
SYN
SYN
SYN
SYN
Waiting for Response to SYNACK
Waiting for Response to SYNACK
Waiting for Response to SYNACK
Waiting for Response to SYNACK
Waiting for Response to SYNACK
Waiting for Response to SYNACK
Waiting for Response to SYNACK
SYNACK
SYNACK
SYNACK
SYNACK
Computer B

Other DDoS attacks include:

- **UDP Flood**. In a UDP Flood attack, the attacker sends a large number of small UDP packets, sometimes to random diagnostic ports (chargen, echo, daytime, etc.), or possibly to other ports. Each packet requires processing time, memory, and bandwidth. If the attacker sends enough packets, then the victim's computer is unable to receive legitimate traffic.

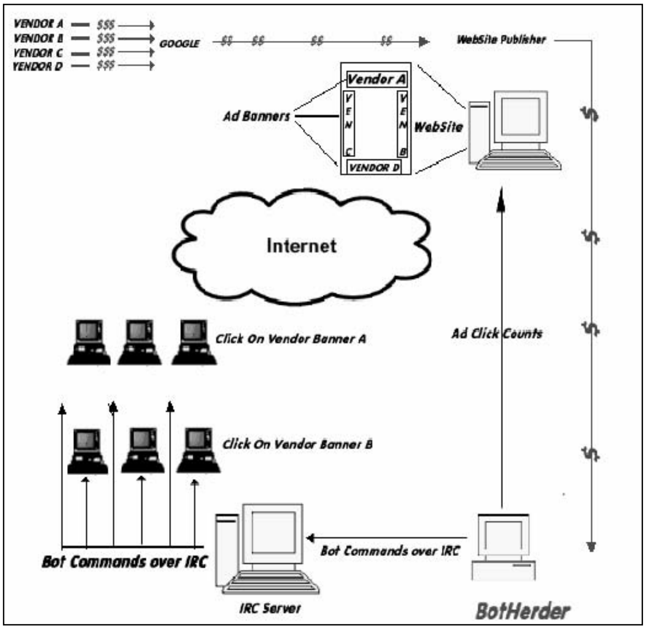
- **Smurf attack**. In a Smurf attack, the attacker floods an ICMP ping to a directed broadcast address, but spoofs the return IP address, which traditionally might be the IP address of a local Web server. When each targeted computer responds to the ping they send their replies to the Web server, causing it to be overwhelmed by local messages. Smurf attacks are easy to block these days by using ingress filters at routers that check to make sure external IP source addresses do not belong to the inside network. If a spoofed packet is detected, it is dropped at the border router. However given that hackers may have subverted 50000 remote hosts and not care about spoofing IP addresses, they can easily be replicated with TCP SYN or UDP flooding attacks aimed at a local Web server.

Installation of Adware and Clicks4Hire

The first criminal case involving a botnet went to trial in November 2005. Jeanson James Ancheta (a.k.a. Resili3nt), age 21, of Downey, California, was convicted and sentenced to five years in jail for conspiring to violate the Computer Fraud Abuse Act, conspiring to violate the CAN-SPAM Act, causing damage to computers used by the federal government in national defense, and accessing protected computers without authorization to commit fraud.

Ancheta's botnet consisted of thousands of zombies. He would sell the use of his zombies to other users, who would launch DDoS or send spam. He also used a botnet of more than 400,000 zombies to generate income in a "Clicks4Hire scam" (see Figure 2.6) by surreptitiously installing adware for which he was paid more than $100,000 by advertising affiliate companies. A DOJ press release stated that Ancheta was able to avoid detection by varying the download times and rates of the adware installations, as well as by redirecting the compromised computers between various servers equipped to install different types of modified adware. For information on how Clicks4Hire schemes work, read the following sidebar and refer to Figure 2.6. Companies like Dollarrevenue.com and Gimmycash.com pay varying rates for installation of their adware software in different countries. Companies like these are paying for criminal activity—that is, the intentional installation of their software on computers without the explicit permission of the owner of the computer. Pressure from the FTC caused one of these vendors (180 Solutions) to terminate 500 of its affiliate agreements for failing to gain user acceptance prior to installing their software. This resulted in the DDoS attack described in Chapter 1, the involvement of the FBI, and a lawsuit against the former affiliates. It also resulted in 180 Solutions changing its name to Zango.

Figure 2.6 A Clicks4Hire Botnet Scam



Are You Owned?

A Botnet Clicks4Hire Scheme

On May 15, 2006, the Internet Storm Center reported another case where a botnet was being used to scam Google's Adsense program into paying for clicks that were artificially generated (for more information see http://isc.sans.org/diary.php?storyid=1334). Here's how it worked (refer to Figure 2.6 to follow along with this explanation).

Under normal circumstances, companies will pay Google for the number of clicks that are generated from banners on Google Web sites.

Continued

Google has relationships with a number of Web site publishers and pays them a significant portion of the revenue they receive in return for hosting these Google banners. Some of the Web site publishers are less than ethical and attempt to find ways to generate their own clicks in a way that Google will not detect. Google does some fraud detection to prevent this kind of activity. Now, however, unscrupulous Web site publishers are hiring hackers that control botnets to command their botclients to click on these Adsense banners. The Web site publishers then share a portion of the revenue with the botnet controllers.

In the hands of a less competent hacker, botnets can cause unintended damage. This was the case with Christopher Maxwell, 20, of Vacaville, California. According to the DOJ press release announcing his conviction, as his botnet searched for additional computers to compromise, it infected the computer network at Northwest Hospital in Seattle. The increase in computer traffic as the botnet scanned the system interrupted normal hospital computer communications. These disruptions affected the hospital's systems in numerous ways: Doors to the operating rooms did not open, pagers did not work, and computers in the intensive care unit shut down.

Last year a set of three Trojans were detected, which worked in sequence to create a botnet. The sequence began with a variant of the Bagle mass-mailing virus, which dropped one of many variations of the W32.Glieder.AK Trojan (see www3.ca.com/securityadvisor/virusinfo/virus.aspx?id=43216 for more information). This Trojan attempted to execute prior to virus signatures being in place. It had shut off antivirus software, firewall software, and XP's Security Center service. Then Glieder went through a hard-coded list of URLs to download the W32.Fantibag.A Trojan. Fantibag prevented the infected machine from getting updates from Windows and from communicating with antivirus vendor sites and downloaded the W32.Mitglieder.CT remote access Trojan. Mitglieder established the botclient and joined the botnet. It also may have downloaded a password-stealing Trojan.

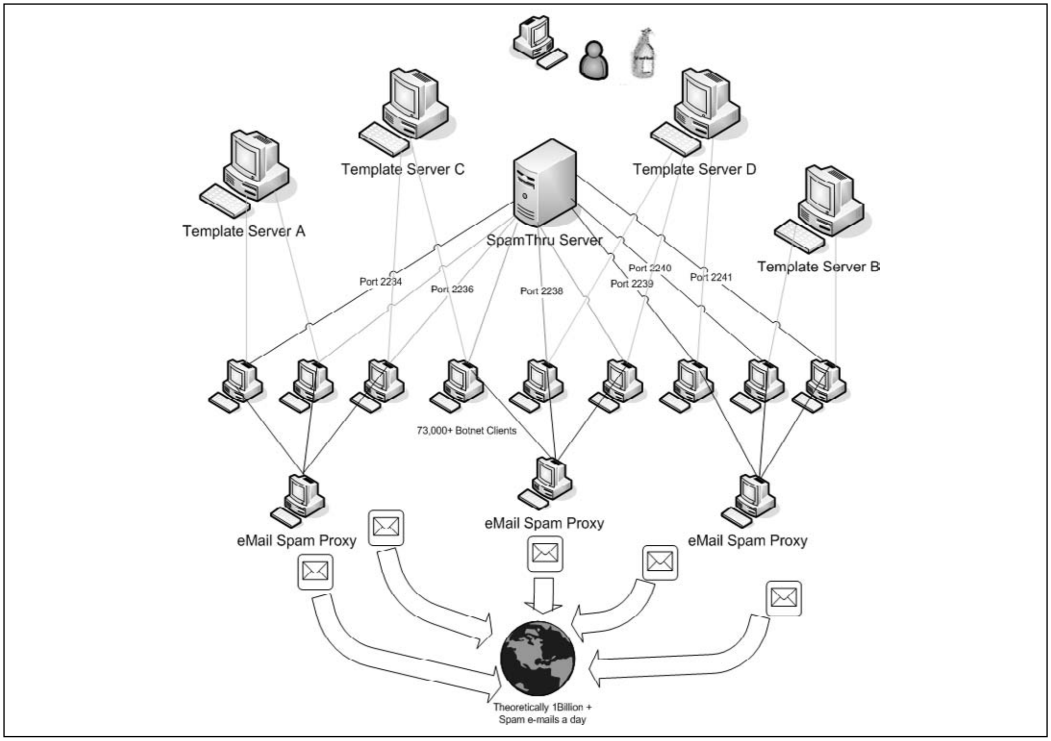
The Botnet-Spam and Phishing Connection

How do spammers and phishers stay in business? As soon as you identify a spam source or phishing Web site you blacklist the IP address or contact the ISP and he's gone, right? Wrong. Today's spammers and phishers operate or rent botnets. Instead of sending spam from one source, today's spammers send spam from multiple zombies in a botnet. Losing one zombie doesn't affect the flow of spam to any great effect. For a botnet-supported phishing Web site, shutting down a phishing Web site only triggers a Dynamic DNS change to the IP address associated with the DNS name. Some bot codebases, such as Agobot, include specific commands to facilitate use in support of spamming operations. There are commands to harvest e-mails, download a list of e-mails prior to spamming, start spamming, and stop spamming. Analyzing the headers of similar spam payloads and phishing attacks may permit investigators to begin to discover members of common botnets. Monitoring activity between these members and the bot server may yield enough information to take the botnet down. Cross-correlation of different kinds of attacks from the same zombie may permit investigators to begin to "follow the money."

Using a botnet, the botherder can set up an automated spam network. Joe Stewart, a senior security researcher from SecureWorks in Atlanta, Georgia, recently gained access to files from a botnet that was using the SpamThru Trojan. The botherders were a well-organized hacker gang in Russia, controlling a 73,000 node botnet. An article in the 20 November 2006 issue of e-Week, titled, "Spam Surge Linked to Hackers," describes Mr. Stewart's analysis for the masses. The details of this analysis can be found at www.secureworks.com/analysis/spamthru/.

Figure 2.7 illustrates the SpamThru Trojan. The botnet clients are organized into groups of similar processing and network speeds. For example, all the Windows 95 and Windows 98 systems that are connected to dial-up connections might be assigned to port 2234, and the higher speed XP Pro systems connected to High Speed Internet connections might be assigned to port 2236. The Russian botherder sends commands through the IRC C&C server to each of the botclients instructing them to obtain the appropriate templates for the next spam campaign. The botnet client then downloads the templates and modifies the data from the template every time it transmits an e-mail. The template includes text and graphics. To foil the graphics spam detectors, the spam clients modify the size and padding in the graphic images for each message.

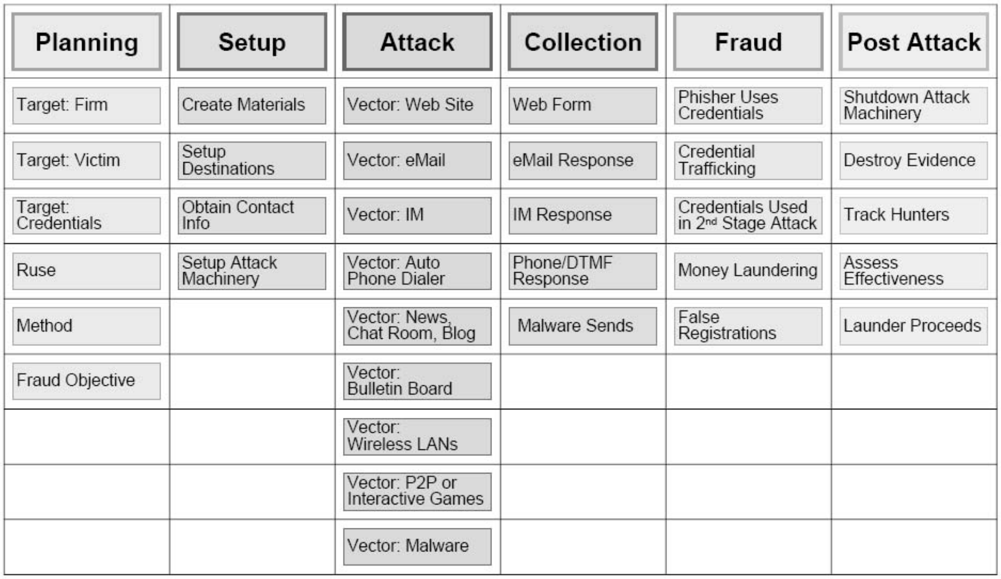
Figure 2.7 The SpamThru Trojan



The botnet clients transmit their spam to an e-mail spam proxy for relay. By using a spam proxy instead of sending the spam directly from each botclient, the spammer protects himself from Relay Black Lists (RBL). Once a proxy is listed as being in an RBL it becomes ineffective to whoever uses the RBL service, since the point of the RBL is to permit organizations to ignore traffic from known spam sites. Using proxies permits the spammer to replace any proxy that is RBL listed with one of the existing clients. They promote the client to a proxy and demote the old proxy back to being a spam engine. By periodically rotating proxy duty sometimes you can avoid being listed by an RBL at all. Stewart calculated that the Russian botnet he analyzed was theoretically capable of sending 1billion spam e-mails a day, given that they had enough e-mail addresses and enough varieties of spam to need that many. These calculations assumed five seconds for each SMTP transaction and that each e-mail would go to only one recipient. You can group your e-mail distribution and send one e-mail to an e-mail server that goes to 100 names on a distribution list. You can see that even the estimate of 1 billion spam e-mails a day is conservative.

Phishing attacks have been analyzed by the Financial Services Technology Consortium (FSTC). Figure 2.8 illustrates a Phishing Operation Taxonomy. It is used with the permission of the Financial Services Technology Consortium (FSTC) and taken from *Understanding and Countering the Phishing Threat*, published by the FSTC on 01/31/2005.

Figure 2.8 FSTC Phishing Attack Taxonomy



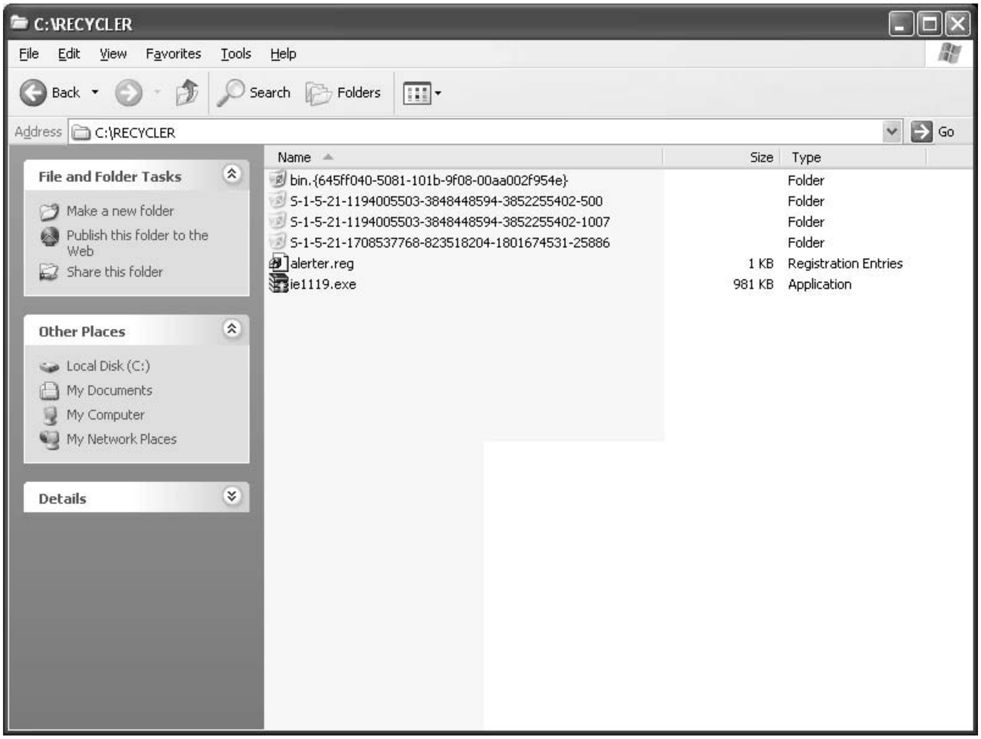
Planning	Setup	Attack	Collection	Fraud	Post Attack
Target: Firm	Create Materials	Vector: Web Site	Web Form	Phisher Uses Credentials	Shutdown Attack Machinery
Target: Victim	Setup Destinations	Vector: eMail	eMail Response	Credential Trafficking	Destroy Evidence
Target: Credentials	Obtain Contact Info	Vector: IM	IM Response	Credentials Used in 2nd Stage Attack	Track Hunters
Ruse	Setup Attack Machinery	Vector: Auto Phone Dialer	Phone/DTMF Response	Money Laundering	Assess Effectiveness
Method		Vector: News, Chat Room, Blog	Malware Sends	False Registrations	Launder Proceeds
Fraud Objective		Vector: Bulletin Board			
		Vector: Wireless LANs			
		Vector: P2P or Interactive Games			
		Vector: Malware			

Each heading in Figure 2.8 represents a phase in the life cycle of a phishing attack. The entries under each life cycle phase represent actions that may take place during that phase. This phase-based approach allows us to examine activities taken by the botherder/phisher for opportunities to intervene. Starting from the left, a botherder participating in phishing attacks would plan the attack by selecting the targets (the financial institution, the victim, and which credentials to go after), selecting the ruse or scam to try, deciding how to carry out the scam by choosing a method from the list in the attack phase, and determining what the goal of this fraud will be. In the setup phase, the phisher creates materials (phishing e-mails and Web sites), and obtains e-mail addresses of potential victims and sets up the attack machinery (botnets, Web pages, template servers, socks proxies). Note that a socks proxy is a system that is configured to relay traffic from a specified protocol. It is a more generalized version of a spam proxy. The name socks comes from the term socket, which is the "identification of a port for machine to machine communications" (RFC 147). Next he launches the attack. The Collection phase uses the method chosen to collect the victim's credentials. The credentials could be gathered using a Web page, a response to an e-mail, a response to an IM, a telephone call, or data collected and transmitted by malware that was downloaded onto the victim's computer. The fraud phase usually is performed by a different group of individuals known as *cashers*. The cashers are responsible for converting the credential information into cash or bartered goods and services. This may involve the casher using the credentials directly, selling the credentials to others, or using the credentials to gain access to the victim's financial accounts. Following the attack, the phisher needs to shut down the phishing attack mechanism, erase the evidence, assess the effectiveness of the attack, and finally, launder the process.

Storage and Distribution of Stolen or Illegal Intellectual Property

A recent report from the Institute for Policy Innovation, *The True Cost of Motion Picture Piracy to the US Economy*, by Stephen E. Siwek, claims that in 2005 the Motion Picture industry sustained losses of approximately $2.3 billion from Internet Piracy. An army of controlled PCs can also represent a virtually limitless amount of storage for hackers to hide warez, stolen movies, games, and such. In one case, hackers had established a network of storage locations. For each botclient they had documented the location, amount of storage, and had calculated file transfer speeds to several countries. The files were stored in hidden directories, some in the recycle bin (see Figure 2.9) where the only visible portion was a folder called "bin.{a long SID-like number here}." Note the period after the word bin. Other systems had files hidden deep below the Windows/java/trustlib directory.

Figure 2.9 Files Hidden in the RECYCLER bin Folder



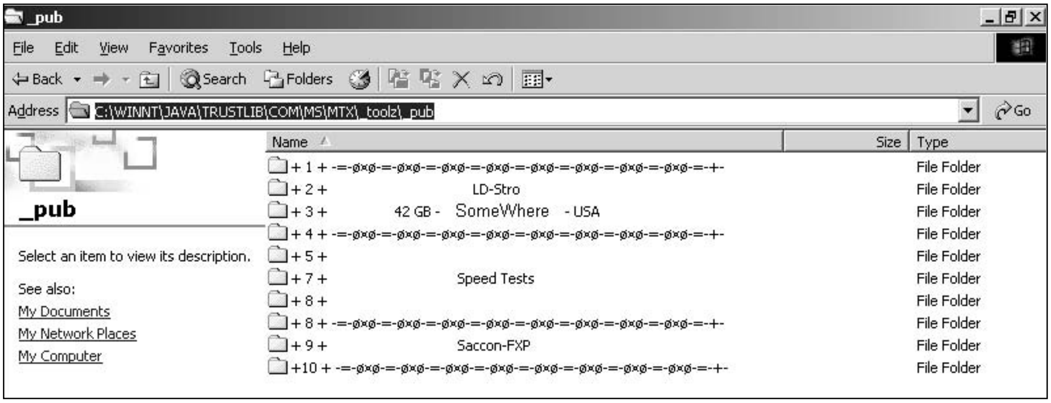
Name	Size	Type
bin.{645ff040-5081-101b-9f08-00aa002f954e}		Folder
S-1-5-21-1194005503-3848448594-3852255402-500		Folder
S-1-5-21-1194005503-3848448594-3852255402-1007		Folder
S-1-5-21-1708537768-823518204-1801674531-25886		Folder
alerter.reg	1 KB	Registration Entries
ie1119.exe	981 KB	Application

Included in the hidden directories were directories called _toolz, _pub and another called sp33d. The botherder also stored stolen intellectual property in the windows uninstall directories for windows patches (see Figure 2.10), such as the following example:

c:\WINDOWS\$NtUninstallKB867282$\spuninst_tmp__\«««SA©©Ø N»»»_Pub

We were able to track these using our workstation management tool, Altiris from Altiris, Inc., by querying managed workstations to see if these directories were on them.

Figure 2.10 Hidden Directories for Stolen Intellectual Property



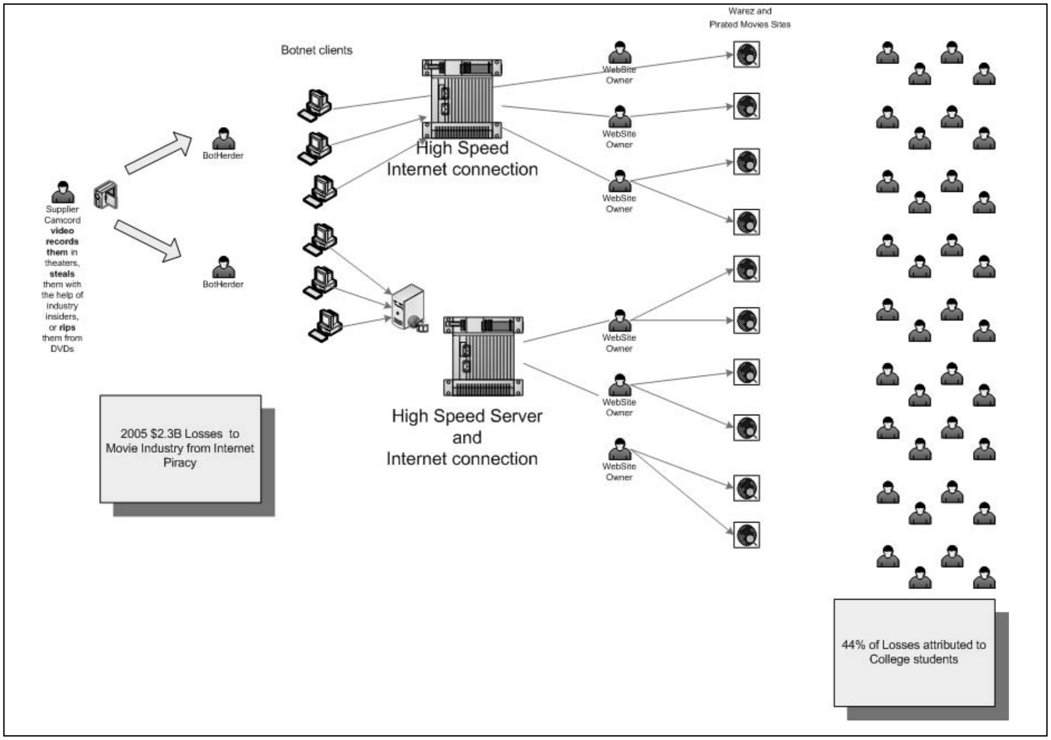
C:\WINNT\JAVA\TRUSTLIB\COM\MS\MTX_toolz_pub

Name	Size	Type
+ 1 + -=-ø×ø-=-ø×ø-=-ø×ø-=-ø×ø-=-ø×ø-=-ø×ø-=-ø×ø-=-ø×ø-=-+-		File Folder
+ 2 + LD-Stro		File Folder
+ 3 + 42 GB - SomeWhere - USA		File Folder
+ 4 + -=-ø×ø-=-ø×ø-=-ø×ø-=-ø×ø-=-ø×ø-=-ø×ø-=-ø×ø-=-ø×ø-=-+-		File Folder
+ 5 +		File Folder
+ 7 + Speed Tests		File Folder
+ 8 +		File Folder
+ 8 + -=-ø×ø-=-ø×ø-=-ø×ø-=-ø×ø-=-ø×ø-=-ø×ø-=-ø×ø-=-ø×ø-=-+-		File Folder
+ 9 + Saccon-FXP		File Folder
+10 + -=-ø×ø-=-ø×ø-=-ø×ø-=-ø×ø-=-ø×ø-=-ø×ø-=-ø×ø-=-ø×ø-=-+-		File Folder

Some of the files were managed using the distributed ftp daemon (Drftpd). The botnet clients run a slave application and take direction from a master ftp server. Others had only a simple ftp server such as a hacked copy of ServU Secure from RhinoSoft.com. ServU is able to set up and use virtual directories, including directories for media on different computers. In addition it includes SSL for secure authentication and encryption of transmitted files, a big plus if you are stealing someone else's intellectual property.

Figure 2.11 illustrates the use of botnets for selling stolen intellectual property, in this case Movies, TV shows, or video. The diagram is based on information from the Pyramid of Internet Piracy created by Motion Picture Arts Association (MPAA) and an actual case. To start the process, a supplier rips a movie or software from an existing DVD or uses a camcorder to record a first run movie in the theaters. These are either burnt to DVDs to be sold on the black market or they are sold or provided to a Release Group. The Release Group is likely to be an organized crime group, excuse me, business associates who wish to invest in the entertainment industry. I am speculating that the Release Group engages (hires) a botnet operator that can meet their delivery and performance specifications. The botherder then commands the botnet clients to retrieve the media from the supplier and store it in a participating botnet client. These botnet clients may be qualified according to the system processor speed and the nature of the Internet connection. The huge Internet pipe, fast connection, and lax security at most universities make them a prime target for this form of botnet application. MPAA calls these clusters of high speed locations "Topsites."

Figure 2.11 Botnet Used to Store and Sell Stolen Movies, Games, and Software


Warez and
Pirated Movies Sites
Botnet clients
Supplier Camcord **video records them** in theaters, **steals** them with the help of industry insiders, or **rips** them from DVDs
BotHerder
BotHerder
High Speed
Internet connection
High Speed Server
and
Internet connection
WebSite Owner
WebSite Owner
WebSite Owner
WebSite Owner
WebSite Owner
WebSite Owner
2005 $2.3B Losses to Movie Industry from Internet Piracy
44% of Losses attributed to College students

According to the MPAA, 44 percent of all movie piracy is attributed to college students. Therefore it makes sense that the Release Groups would try to use university botnet clients as Topsites. The next groups in the chain are called Facilitators. They operate Web sites and search engines and act as Internet directories. These may be Web sites for which you pay a monthly fee or a fee per download. Finally individuals download the films for their own use or they list them via Peer-to-Peer sharing applications like Gnutella, BitTorrent for download.

In part the motivation for Release Groups to begin to use botnets and universities may be successful law enforcement efforts over the last few years. Operation Buccaneer (2001), Operation Fastlink (2004-ongoing), Operation D-Elite (2005-2006), and Operation SiteDown (2005-ongoing) all targeted Topsite operators. Operation Buccaneer included raids on computers related to MIT, University of Oregon, UCLA, Purdue, and Duke University. The universities were not considered targets of the criminal investigations. However, in each case the courts have ordered the seizure and forfeiture of hundreds of computers owned and operated by the Topsite operators. In order to limit their losses, I believe that some Topsites have turned to botnets to store their stolen IP instead of investing in their own equipment that may be lost if they are caught.

WARNING

Piracy can lead to felony convictions and seizure of property. Table 2.2 lists defendants who have been convicted of various piracy-related offenses.

Table 2.2 Piracy Felons

Defendant	Nickname	Warez Group Affiliations	Conviction Date	Offense
SANKUS, John, Jr. Philadelphia, PA.	eriFlleH	DrinkOr Die, Harm	Felony Feb. 27, 2002	Conspiracy
ERICKSON, Barry Eugene, OR	Radsl	RiscISO, DrinkOrDie, POPZ	Felony May 2, 2002	Conspiracy
GRIMES, David A. Arlington, TX	Chevelle	DrinkOrDie, RISC, RTS	Felony March 4, 2002	Conspiracy
NAWARA, Stacey Rosenberg, TX	Avec	RTS, Razor1911, DrinkOrDie	Felony March 19, 2002	Conspiracy
HUNT, Nathan Waterford, PA	Azide	CORPS, DrinkOrDie	Felony April 3, 2002	Conspiracy
PATTANAYEK, Sabuj Durham, NC	Buj	DrinkOrDie, CORPS, RTS	Felony April 11, 2002	Conspiracy
KELLY, Michael Miami, FL	Erupt	RiSC, AMNESiA, CORE, DrinkOrDie	Felony April 10, 2002	Conspiracy
CLARDY, Andrew Galesburg, IL	Doodad	POPZ, DrinkOrDie	Felony April 4, 2002	Criminal copyright infringement and aiding and abetting

Continued

Table 2.2 continued Piracy Felons

Defendant	Nickname	Warez Group Affiliations	Conviction Date	Offense
TRESCO, Christopher Boston, MA	BigRar	RiSC, DrinkorDie	Felony May 28, 2002	Conspiracy
EISER, Derek Philadelphia, PA	Psychod	DrinkOrDie	Felony June 21, 2002	Criminal Copyright Infringement
NGUYEN, Mike Los Angeles, CA	Hackrat	Razor1911, RISC	Felony Jan. 31, 2002	Conspiracy
KARTADINATA, Kent Los Angeles, CA	Tenkuken	DrinkOrDie	Felony Jan. 31, 2002	Conspiracy
BERRY, Richard Rockville, MD	Flood	POPZ, DrinkOrDie	Felony Apr. 29, 2002	Conspiracy
RIFFE, John Port St. John, FL	blue	SMR, EXODUS	Felony May 9, 2002	Criminal Copyright Infringement
GROSS, Robert Horsham, PA	target-practice	DrinkOrDie	Felony May 22, 2002	Criminal Copyright Infringement
COLE, Myron Warminster, PA	t3rminal	DrinkOrDie	Felony July 10, 2002	Criminal Copyright Infringement
BUCHANAN, Anthony Eugene, OR	spaceace	POPZ, DrinkOrDie	Felony August 19, 2002	Criminal Copyright Infringement

Ransomware

As a category this includes any of the ways that hackers may hold a person's computer or information hostage. Ransomware, for this book, includes using a botnet to DDoS a computer or a company until a ransom is paid to make the DOS stop. The hacker may use Paypal or Western Union to arrange for difficult-to-trace money transactions. When a botnet handler realizes they have a computer that might be worth ransoming, they can encrypt important files and demand a ransom for the key and/or software to decrypt them. Last year a DDoS ransom attack was launched to target 180Solutions(now known as Zango), a spyware company that tried to go legit. 180Solutions terminated over 500 of the company's affiliates due to their practice of installing the company's adware without the knowledge of the user. One group of affiliates used the same botnet that had been installing the adware to launch their DDoS attack.The company responded by contacting the FBI. With the FBI's help they tracked down the operators of the botnet in several countries around the world. Once the attackers were known, 180Solutions filed a civil suit against the seven hackers involved in the DDoS attacks.

Data Mining

The final payload type we will cover is data mining.This can be added to any of the other types of functionality pertaining to botnet clients. For this, the botherder employs tools to gather information from each of the botnet clients or their users.They will at a minimum enumerate the users of the computer and note which accounts have local administrator accounts.They may collect the Security Accounts Manager (SAM) database or any password cache storage to be broken. Breaking these passwords may take place on the client or the information may be reformatted and sent to another computer to have a password cracking program run against it.

The botnet client can be searched for numbers that look like credit card numbers or Social Security Account Numbers (SSANs). Credit card and SSAN information can be sold on special Web sites established for that purpose. Some botnets establish keylogger programs that record every keystroke taken on the computer. Later, userIDs and passwords can be harvested from the logs. Recent malicious code has been very precisely targeted. Code has been found that piggybacks a legitimate user as they login to an e-Gold account. Once in, they initiate an electronic funds transfer and siphon off the user's money.

Reporting Results

Using the Command and Control mechanism, the botclient would report results (when appropriate) back to the C&C server or to a location directed by the commands from the botherder. For some of these payloads (spamming, Clicks4Hire, etc.), reporting back to the botherder may provide needed data to help the botherder know how much to expect to be paid. Reporting also lets the botherder know that the bot is ready for another assignment. This brings the botnet client to the beginning of the iterative portion of the life cycle. Botnet clients repeat this cycle ad naseum until the botnet client is discovered or until the botherder decides to abandon it.

Erase the Evidence, Abandon the Client

If the botherder believes that the botclient has been discovered or if a portion of the botnet in the same domain has been found or the botclient is no longer suitable (too slow, too old), the botherder may execute a prestaged command that erases the payload and hacker tools. We've observed cases where the security event logs and antivirus risk histories have been cleared or erased. A tool like clearlogs.exe automates the process. Sometimes when the botherder abandons a client, our antivirus tool will pick up several components when the hide capability is turned off. When this happens, the detection date reflects their exit date instead of the actual date of infection.

Botnet Economics

> I have ways of making money that you know nothing of.
> *—John D. Rockefeller*

Spam and Phishing Attacks

Most people can't understand how anyone could make money sending out spam. It is the global scope of the Internet that makes it possible. When Jeremy Jaynes was arrested as one of the top ten spammers in the world authorities say he earned $750,000 a month selling fake goods, services, and pornography via spam. Evidence presented during the trial showed that he had made $24 million through various e-mail schemes. For every 30,000 e-mails he sent one person bought what he was selling, earning him $40. It is estimated that he sent over 10 million e-mails. He was arrested in December 2003 and convicted in November 2004.

Christopher Abad provides insight into the phishing economy in an article published online by FirstMonday.org (http://www.firstmonday.org/issues/issue10_9/abad/). The article, "The economy of phishing: A survey of the operations of the phishing market," reveals the final phase of the phishing life cycle, called *cashing*. These are usually not the botherders or the phishers. The phishers are simply providers of credential goods to the cashers. Cashers buy the credential goods from the phishers, either taking a commission on the funds extracted or earned based on the quality, completeness, which financial institution it is from, and the victim's balance in the account. A high-balance, verified, full-credential account can be purchased for up to $100. Full credentials means that you have the credit card number, bank and routing numbers, the expiration date, the security verification code (cvv2) on the back of the card, the ATM pin number, and the current balance. Credit card numbers for a financial institution selected by the supplier can be bought for 50 cents per account. The casher's commission of this transaction may run as much as 70 percent. When the deal calls for commissions to be paid in cash, the vehicle of choice is Western Union.

The continuation of phishing attacks depends largely on the ability of the casher's to convert the information into cash. The preferred method is to use the credential information to create duplicate ATM cards and use the cards to withdraw cash from ATM terminals. Not surprisingly the demand for these cards leans heavily in favor of banks that provide inadequate protections of the ATM cards. Institutions like Bank of America are almost nonexistent in the phisher marketplace due to the strong encryption (triple DES) used to protect information on its ATM cards.

Adware Installation and Clicks4Hire Schemes

Dollar-Revenue and GimmyCash are two companies that have paid for installation of their Adware programs. Each has a pay rate formula based on the country of installation. Dollar-Revenue pays 30 cents for installing their adware in a U.S. Web site, 20 cents for a Canadian Web site, 10 cents for a U.K. Web site, 1 cent for a Chinese Web site, and 2 cents for all other Web sites. GimmyCash.com pays 40 cents for U.S. and Canadian Web site installs, 20 cents for 16 European countries, and 2 cents for everywhere else. In addition, GimmyCash pays 5 percent of the webmaster's earnings that you refer to GimmyCash.

Before the New York and California class action lawsuits against DirectRevenue, the *Washington Post* profiled the life of a botherder that called himself 0x80. In the article, "Invasion of the Computer Snatchers," written by Brian Krebs (www.washingtonpost.com/wp-dyn/content/article/2006/02/14/AR2006021401342.html), Krebs says that 0x80 earned between $6,000 and $10,000 a month installing adware. Not bad for a high school dropout from Roland, Oklahoma. That works out to about $300 a day, if he works only on weekdays. If he installed GimmeCash adware on U.S. and Canadian computers it would take 750 computers to make that amount. If you have 10,000 clients in your botnet you can see the opportunity. In addition, you would add a variable amount of profit related to the 5 percent you earn on any sales that come from the ads. When that runs dry, you can start over with the next adware vendor. All the while you could be adding more botclients to the net.

Proposed Settlement of the DirectRevenue California Class Action Lawsuit

Here is a summary of the proposed settlement of California's class action lawsuit against DirectRevenue. Under the settlement, DirectRevenue will be required to conform to the following business practices, among others, concerning its Software (as that term is defined in the Agreement). The following excerpt from this settlement was taken from Case No.: 05-CV-02547-LKK-PAN (JFM) filed in United States District Court, Eastern District of California (http://classactiondefense.jmbm.com/battagliaclassactiondefense_fao.pdf).

> a. Direct Revenue will not intentionally collect any personally identifiable information
>
> (name, address, phone number, social security number, e-mail address, bank account information, etc.) about computer users.

b. Direct Revenue will assure that, prior to the installation of the Software, computer users are (a) provided with Direct Revenue's End User License Agreement ("EULA"), and (b) given two choices, of equal prominence within the modal box or landing page, to the effect of:

"I have read and accept the agreement" or

"I do not accept the terms of the agreement"

The "accept" option will not be a default option. If the user selects the "I do not accept" choice, the Software will not be installed.

An example of an acceptable disclosure is attached hereto as Exhibit A.

c. In addition to providing computer operators with its EULA, Direct Revenue will also disclose, separate and apart from the EULA, that: (1) users will receive advertisements while online, along with a brief description of the types of ads that will be displayed; (2) Direct Revenue will collect information about web sites visited by users; and (3) the Software will be included in their installation of the adsupported software. This disclosure will be independently displayed within the modal box containing the "I have read and accept" and "I do no accept" choices described above. The additional disclosures shall appear above the choices described in subparagraph b, above, but will end no more than one inch away from those choices.

d. Direct Revenue, will not install Software via ActiveX installations, or by any other method that does not require users' affirmative consent.

e. Direct Revenue will not install Software via computer security exploits.

f. In Direct Revenue's EULA, Direct Revenue will disclose the fact that the Software serves pop-up ads based on web sites visited by the user, and that Direct Revenue collects non-personally identifiable information, in order to serve those ads. The EULA will explain Direct Revenue's use of the non-personally identifiable information. The EULA will also notify users as to how the Software can be uninstalled, and will provide information on how to access Direct Revenue's website and customer support.

g. In distribution contracts executed following the parties execution of this settlement agreement, DirectRevenue will require distributors to abide by the policies represented in this settlement. DirectRevenue will closely police its distributors. If DirectRevenue learns that a distributor is violating the terms of its distribution contract, Direct Revenue will take appropriate action based on the circumstances of the violation, potentially including termination of the distributor.

h. Distributors will not be permitted to use sub-distributors unless those entities are bound by contract to adhere to the policies represented herein.

i. DirectRevenue will not distribute the Software via web sites that in DirectRevenue's good faith belief are targeted primarily at children. The EULA will include a disclosure that the Software should only be installed by users 18 years of age and older, and instructions (or a reference link to such instructions) on how to manage the user's operating system to minimize the possibility that children will be served with ads by the Software. Direct Revenue will disclose to Net Nanny (and similar services) the IP address of any server sending adult content ads through the Software.

j. DirectRevenue will not use the word "free" in banner ads describing the underlying program (i.e., the screen saver or video game) unless the ad also discloses that the program is ad-supported.

k. When the Software displays a pop-up ad, the "X" button on the title bar of the ad window (used to close the ad window) will not appear off-screen, unless this effect is caused by a technical issue without DirectRevenue's knowledge or beyond DirectRevenue's, control.

l. All DirectRevenue ads will include a "?" button on the title bar, or a text link indicating that further information is available, which displays information about the Software when clicked. This information will include (1) an explanation of why the user is receiving the ad; (2) the identity of the consumer application the user downloaded with the Software (when and to the extent this is technically feasible); and (3) an instruction that, if the user so desires, the user can uninstall the Software using the Windows "Add/Remove Programs" function.

m. The Software will not display adult content ads unless the user is viewing adult websites. DirectRevenue will disclose to Net Nanny (and similar services) the IP address of any server sending adult content ads through the Software.

n. The Software will be listed in the Windows "Add/Remove Programs" list under the exact same name used in branding the ads.

o. DirectRevenue will not modify security settings on users' computers.

p. DirectRevenue will not reinstall its Software once a user has uninstalled it through the Windows "Add/Remove Programs" function or other removal method, unless the user later opts to download and install another bundled application and the installation proceeds in accordance with the terms herein.

> q. DirectRevenue will not delete other software on the user's computer other than any underlying program (e.g. screensaver) that was bundled with the Software upon the user's removal of the Software.
>
> r. DirectRevenue will not materially modify the Software's functionality without providing the user with notice and an opportunity to uninstall the Software.
>
> s. DirectRevenue will agree to limit its advertisements to a network average of 10 or less per computer per 24-hour period.
>
> t. DirectRevenue agrees that its removal instructions shall continue to be posted in a form in substantial conformity with that currently found at: http://www.bestoffersnetworks.com/uninstall/.
>
> u. DirectRevenue will limit its number of name changes used on its advertisements (*i.e.*, "Best Offers") to once per two years.
>
> v. DirectRevenue will agree to purchase sponsored links, if Google is willing to sell such sponsored links, that provide links to help consumers remove DirectRevenue's software. At a minimum, DirectRevenue will agree to purchase links, if Google is willing to sell such sponsored links, for "BestOffers" and "BestOffers removal". By clicking on the sponsored link, the user will be taken to an Internet page with instructions on how to remove the Software. Should DirectRevenue change the name of its software, it will purchase sponsored links with the new name of the Software referenced.
>
> w. DirectRevenue will not "flush" or otherwise remove domain names from browser's list of "trusted sites".

The current trend of State's Attorney Generals suing adware companies that support this industry should have an impact on this threat in the long run. With the attention received from the lawsuits and public scrutiny raised by Security activist Ben Edelman, major adware/spyware companies are in retreat. DirectRevenue is down to a couple of dozen employees and has lost many of their largest accounts.

The botherder is well positioned to conduct click fraud attacks against advertisers and adware companies that pay commissions for affiliates to drive customers to advertising clients' Web sites. Business offerings like the Google Adsense program do not advertise their algorithm for paying click commissions but they do pay, or actually, Google advertising customers have the option of paying, for this service. Google employs an algorithm to try to detect click fraud. Google tells its customers that they are not charged for fraudulent clicks but there is no way to gauge the effectiveness of their fraud detection efforts.

Ransomware

In an online article titled "Script Kiddies Killing The Margins In Online Extortion," published in the online magazine *TechDirt Corporate Intelligence* (www.techdirt.com), the author (who goes by Mike) claims that the going rate to decrypt online ransoms of files has been between $50 and$100. The Zippo ransomware Trojan demanded $300 be paid to an e-gold account for the password to decrypt ransomed files. The codebreakers at Sophos determined the password was:

```
C:\Program Files\Microsoft Visual Studio\VC98
```

The Arhiveus ransomware Trojan encrypts all of the files in the My Documents folder with a 30-character password. Sophos has determined this password to be:

mf2lro8sw03ufvnsq034jfowr18f3cszc20vmw

Without the password, victims were forced to make a purchase from one of three online drug stores.

The Ransom A Trojan is a budget ransomware package. It encrypts the user's data, and then instructs the user to wire $10.99 to a Western Union CIDN. Once the CIDN number is entered in the ransomware, the software promises to remove itself and restore access to the data.

Summary

With botnets, hackers called botherders are able to wield thousands of computers to do their will. By using a command interpreter to execute a common set of commands, a botherder is able to coordinate and manage these thousands. The botclients are not viruses, per se. They are, instead, a collection of software that is being put to malicious use. The software can include viruses, Trojan backdoors and remote controls, hacker tools such as tools to hide from the operating system, as well as nonmalicious tools that are useful. The fact that the botherder does not actually touch the computer that performs the illegal acts is a model that has been used by organized crime for years.

Botclients operate in a regular cycle that can be characterized as a life cycle. Understanding the life cycle in Figure 2.1 will help both investigators and researchers in finding ways to discover, defend against, and reduce the threat of botnet technology.

Similarly, studying the economics behind each of the botnet payload types can reveal strategy and tactics that can be used against the problem. Particularly, finding ways to reduce the demand element could result in less use of botnets in whole classes of behavior.

Solutions Fast Track

What Is a Botnet?

☑ A botnet consists of at least one bot server or controller and one or more botclients, usually in the many thousands.

☑ The heart of each botclient is a command interpreter that is able to independently retrieve commands and carry them out.

☑ The ability of the botnet to act in a coordinated fashion with all or some parts of the botnet is fundamental to the botnet concept.

☑ Botnets are not a virus in the traditional sense of the word. Rather they are a collection of software (some viruses, some malicious code, some not) put together for malicious purposes.

☑ Botnets are managed by a botherder.

☑ Hackers are attracted to botnets because botnet clients carry out their orders on computers that are at least two computers removed from any computer directly connected to them. This makes investigation and prosecution more difficult.

The Botnet Life Cycle

☑ The life of a botclient can be described as a life cycle. Steps 5 through 8 are iterative and are repeated until the command to abandon the client is given.

1 Computer exploited and becomes a botclient.

2 New botclient rallies to let botherder know he's joined the botnet.

3 Retrieve the latest Anti-A/V module.

4 Secure the new botclient from A/V, user detection, and other hacker intervention.

5 Listen or subscribe to the C&C Server/Peer for commands.

6 Retrieve the payloads modules.

7 Execute the commands.

8 Report results back to the C&C server.

9 On command, erase all evidence and abandon the client.

What Does a Botnet Do?

☑ Botnets can do anything a single computer or network of computers is capable of doing. Botnets advertise their availability on IRC channels and other places and sell all or portions for others to use.

☑ Here are the most commonly reported uses of botnets:

- Recruit other botclients (sniffing for passwords, scanning for vulnerable systems).
- Conduct DDoS attacks.
- Harvest identity information and financial credentials.

- Conduct spamming campaigns.
- Conduct phishing campaigns.
- Scam adware companies.
- Install adware for pay without the permission of the user.
- Conduct Clicks4Hire campaigns.
- Store and distribute stolen or illegal intellectual property (movies, games, etc.).
- Analysis of the various attack taxonomies, such as that performed by Financial Services Technology Consortium (FSTC), can reveal valuable strategic and tactical information about how to respond to these threats.

Botnet Economics

- ☑ The big news in 2006 was the announcement of the discovery of evidence for the long-suspected ties between botnet/spam/phishing activity and organized crime.
- ☑ With spammers making as much as $750,000 a month it is no wonder that there is such a demand for botnets that spam. It is the global reach and economy of scale of the botnet that makes this market possible.
- ☑ Adware/spyware companies created a marketplace for unscrupulous botherders to install adware/spyware on thousands of computers for pay.
- ☑ Companies that seek to drive qualified customers to their Web sites have created another market. This market takes the form of advertising programs that pay for ads on Web sites that pay affiliates each time a potential customer clicks on ads on the affiliate's Web site. Botherders saw an opportunity in the form of thousands of botclients sitting idle that could be orchestrated to simulate random customers across the Internet.

☑ The demand for free or cheap movies, software, games, and other intellectual property and law enforcement's confiscation of computer equipment engaged in the commission of major thefts of these commodities has created another opportunity for the botherders. Botnets are being used to store an amazing amount of stolen property on their botclients. With hard drive capacities growing, the botherders are finding that they can snag 20G or 30G of hard drive space from most of their clients without the user noticing. This type of venture yields either cash, services, or other stolen intellectual property.

☑ Botherders recognized that some of their client's owners might pay if certain data were held for ransom. A group of ransomware Trojans have been used to encrypt all of the user's files. The botherder then has the victim pay by e-Gold, Western Union, or the old fashion way by making purchases from designated online stores. Ransoms ranged from the budget-minded $10.99 to $300 for the Zippo ransomware Trojan.

Frequently Asked Questions

The following Frequently Asked Questions, answered by the authors of this book, are designed to both measure your understanding of the concepts presented in this chapter and to assist you with real-life implementation of these concepts. To have your questions about this chapter answered by the author, browse to www.syngress.com/solutions and click on the "Ask the Author" form.

Q: How do I know if my computer is part of a botnet?

A: If you are part of a company or organization, you will likely learn that your computer is part of a botnet from either network administrators, system administrators, or your information security organization. It is difficult for an individual to know for sure. Here are some signs to look for. Not all signs will be present in all cases and the presence of these signs could also be explained by other phenomena.

- At times your computer may run significantly slower than normal. Unfortunately this is commonly due to AV software searching for various forms of malware, including botnet clients.
- The network activity light on your DSL modem or NIC card may flash rapidly during a time when you aren't doing anything that you believe would cause network traffic.
- Your antivirus program may shut off by itself.
- If it's still running, your antivirus program may detect several types of malicious code at one time. The names given to the viruses may indicate parts of a botclient's functionality like hide windows, backdoor, and so on.
- Your Windows XP firewall log, which may be called pfirewall.log if a domain policy hasn't picked another standard, is located in the Windows or WINNT directory. Examine any Inbound Open source IP addresses and destination ports for a rational explanation. If you have access to lists of Command and Control servers, any traffic to a known C&C server should be considered a big clue.
- Run TCPView from www.systeminternals.com. Examine all of the network connections and the processes that are associated with them. Any unknown processes or unfamiliar connection IP addresses should be investigated.
- Run Process Explorer from www.systeminternals.com. Examine the processes to see if any processes are running that don't normally run on your computer. Right-click to be able to select Verify. If the vendor is unable to verify the process, you can click on Google on the same menu. Using Google you can see if anyone else has reported bad things about the process. One problem with this approach is that hackers may replace known good executables with malware and reuse the good software's name.
- Check the security event log for login failure for network type 3 where the workstation name does not match the local computer name. This would be a sign of a password guessing attack, particularly

if there is no reason for other workstations to log in to your computer.

Q: How do botnets use IRC for Command and Control?

A: When recruited, botclients are instructed to subscribe to an IRC server, on a specific channel. Each channel has several different topics. The IRC channel topics contain bot commands. Some versions of botnets use multiple channels for different functions. The main channel topic may direct the botclient to go to a string of additional channels. Each channel's topic contains the commands that the botclient will carry out. Each botclient has a command interpreter that understands the command strings found in the channel topic names. It is this command interpreter that makes a bot a bot. It's also easy to see how other technologies could be used for the Command and Control function. There is much more on this topic in Chapter 8.

Q: Why do botherders do these terrible things?

A: The easy answer is for money and power. I believe that a large part of the problem is that we, as a society, do not teach ethics and responsibility when kids learn about computers and the power of the Internet. On the other side of the equation, academia, business, and industry continue to underfund security and produce products and services with inadequate security. The Organization of Economically Cooperating Democracies (OECD) says that the world needs to create a culture of security. Unfortunately academia, business, and industry want to continue to believe that it is okay to deliver functionality first and add security later, if the market demands it. Only later never comes or when the market does demand it, the retrofit is very expensive or is only a band-aid. Our current culture makes it very easy for an unethical hacker to turn our security failings to their financial advantage.