# 1

# Windows Attacks



Sixth century B.C. Chinese war philosopher Sun Tzu is popularly credited with first publishing the "Know Thy Enemy" battle strategy. In order to set up a secure computer defense, you have to define the enemy correctly. This is where many computer security defense courses, books, and articles go wrong. They spend the majority of their time telling you how to defend against the dedicated, manual attacker while either ignoring or giving improbably brief coverage to the much more realistic threat of malicious mobile code and malware networks. And if they can't define the problem correctly, how can they tell you how to successfully defend your computing environment? This chapter summarizes the various types of attacks that malware (and the dedicated hacker) can use to compromise Windows-based computers, and discusses the vulnerable areas of Windows in detail. Table 1-1, "Where Malware Hides," at the end of the chapter, is the most exhaustive list available in any publication.

## Attack Classes

When all the various types of possible attacks against *any* computer system are analyzed, four descriptive classes are noted: automated malware versus the dedicated, manual attacker and remote versus local execution (see Figure 1-1). This section discusses the four categories and then breaks down the methodologies that each employs.

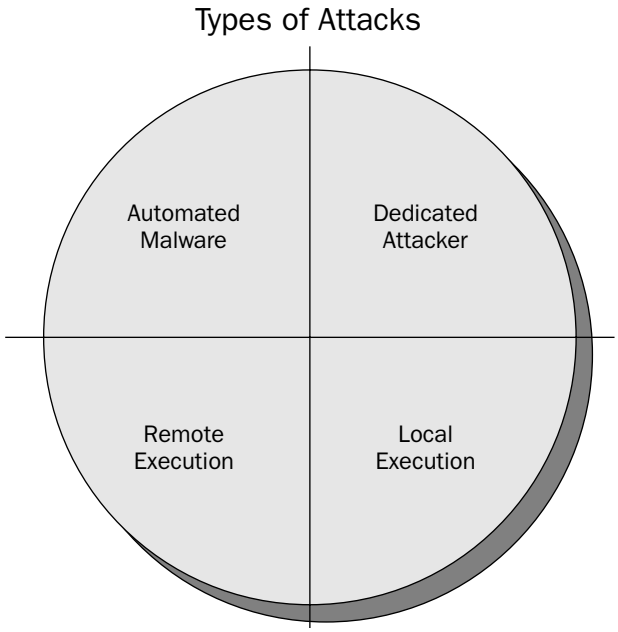


Figure 1-1

## *Automated Versus Dedicated Attacker*

Automated malware is composed of any rogue code designed to exploit and replicate with a minimum of human intervention. The category includes worms, viruses, trojans, spyware, bots, and spam, or phishing attacks launched through any of the former types. Indeed, 99.9% of all computer attacks occur from automated malware! As simplistic as this statement seems, it appears that much of the world still doesn't get it. For reasons still unknown to me, most Windows security courses and defenses concentrate far too much time on the dedicated attacker threat, when automated malware is considerably more prevalent and dangerous. Part of me thinks it is because of the "mysterious intrigue" the evil hacker provokes. It is just human nature to be more concerned with an unpredictable and emotional warm-blooded threat than with the predictable automated attack of a coded program. Whatever the reason, too many books, classes, and people concentrate on the wrong threat.

This is not to say that classes concentrating on the very real threat of hacker's are a waste of time. No, the threat of hackers is a real threat, and the skills learned from those classes can be applied against automated malware. It's just that so many classes ignore the bigger threat of automated malware.

If this text appears overly defensive about the issue, it is because the idea of malware being the largest threat goes against conventional wisdom in many circles and has been the subject of a few heated debates among computer security experts. I've been asked many times to provide proof of my conclusions. My first response is "Go check your own firewall logs!" Any network or security administrator can tell you that their security and antivirus logs are full of daily attacks from automated threats. My web sites and honeypots receive hundreds of attempted exploits a day. Almost all are from automated malware scans. Today, the biggest threats to Windows computers are malicious e-mail attachments, Internet scanning worms, and botnets.

Today, the most popular e-mail worms exploit tens to hundreds of thousands of machines in a single day; and even those barely rate mentioning in the press anymore. It takes a 10-minute infector like the SQL.Slammer worm for the automated attack to become noteworthy. How many sites can a single hacker successfully exploit in a night without having to rely on automated malware? A handful, maybe. The sheer mechanics of automatic attacks alone suggests the veracity of my claim. Once an automated malware program is released, it can exploit millions of computers in a day. They keep on infecting and exploiting until the hole they use is closed or the technology moves on.

The first (known) IBM PC virus, Pakistani Brain, spread around the world with no problem, albeit it took months in the era before the Internet. It infected only 5 1/4-inch floppy diskettes.

The Brain virus remained one of the most popular viruses until the 3.5-inch floppy diskette and hard drives became more prevalent and the original PC floppy drive disappeared.

The Wild List (`www.wildlist.org`) reports each month on the most popular viruses still seen "in the wild." Its November 2005 list (`www.wildlist.org/WildList/200511.htm`) has several viruses from the 1990s, including a few from 1994 and 1995. Once released into the wild, an automated malware program can never be recalled. If a hacker stops hacking, whether through maturity or being incarcerated, the hacking stops immediately.

## Statistics

If you don't believe me and your firewall logs, let's look at some verifiable statistics. According to the FBI's respected 2004 Computer Crime and Security Survey (available at `www.gocsi.com`), 78 percent of reporting businesses detected a computer virus on their network, making it the number one reported computer threat. Although I'm not sure how the survey defines computer virus in the results (for example, does it include worms and Trojans?), I'm fairly confident that many cases went unreported.

The 2004 ICSA Labs Tenth Annual Computer Virus Prevalence Survey (`www.cybertrust.com/intelligence/white_papers.html`) has numbers even more in line with most corporate administrators' experience. The survey reports an average of 392 automated malware encounters per 1,000 computers in the typical large corporation, with an average of 116 successful infections per site per month. Has anyone you've known received 116 manual hacker attacks in a given month? Average recovery cost from a virus outbreak in 2004 was $130,000 (with an average server downtime of 23 hours). Despite the fact that nearly all (99%) the companies surveyed had antivirus measures on at least 90% of their computers (84% claimed 100% coverage), only 12% of the respondents felt that the automated malware problem was the same or better than in the past. This means 88% of the survey takers perceive the problem being worse than ever!

Several computer security reporting agencies have reported that nearly 100% of active e-mail addresses have received spam, viruses, and fraudulent e-mails. MessageLabs, a leading e-mail security service provider (`www.messagelabs.com/emailthreats/intelligence/reports/monthlies/april05/default.asp#t7`) reports that 69% of all e-mail is spam, and 1 in 43 contains a virus. They also report (`www.messagelabs.com/emailthreats/intelligence/reports/monthlies/march05/default.asp`) that over 70% of all spam (including phish e-mails) is created and sent by automated bots that have compromised the computers of innocent users.

The Anti-Phishing Workgroup (`www.antiphishing.org`) reports in its February 2005 report (`www.antiphishing.org/APWG_Phishing_Activity_Report_Feb05.pdf`) that fraudulent "phishing" e-mails have been increasing at a rate of 26% per month since July 2004, with an estimated 75 to 150 million phishing e-mails sent `daily (www.antiphishing.org/APWG-FDICCommentaryLetter.doc)`. The average conversion rate (i.e., the percentage of users revealing personal identification information to the

phisher) ranges from about 2% to 15%. Even the lower end is not miniscule when you consider the overall volume. Spyware is an even more prevalent problem.

A Dell Computers survey (`www1.us.dell.com/content/topics/global.aspx/corp/pressoffice/en/2004/2004_10_15_dc_000?c=us&l=en&s=corp`) revealed that 90% of all computers in the United States have spyware installed, and most users aren't aware of its presence. Another report (`http://aroundcny.com/technofile/texts/tec082904.html`) claims that the average PC has 50 to 70 spyware infections, but PCs with 900 or more infections are not uncommon.

Connect a computer directly to the Internet and it will begin to receive probes within minutes to hours. Before the first 24 hours have passed, it will have received dozens to hundreds of attempted exploits. You can recognize automated attacks versus the manual methods by looking at the type of attacks and the timing. Automated attacks rarely port scan a particular host looking for a vulnerable port, or try to fingerprint any found services. They immediately launch their exploit against a particular host without even trying to figure out whether the host could ever be successfully compromised by its specific exploit. A common web site malware script program tries nearly 100 different attacks in five seconds. About half would only work against Microsoft's Internet Information Server (IIS), the other half would only be successful against the open-source Apache web server. The automated malware program tries all attacks regardless of the found host, and it needs only one to be successful. It then e-mails the originating hacker (who is using a free and hard to trace e-mail account) if it is fruitful. Another common Microsoft SQL Server malware program attempts dozens to hundreds of different passwords against found SQL connections, one right after another. I can tell the SQL malware program is automated because of the speed of the password guesses and the fact that it launch against any PC advertising the standard SQL ports, regardless of whether a SQL login prompt is offered.

Without a doubt, automated malware, not the dedicated lone hacker, is the biggest threat to computers. But in my many years of teaching Windows security, no class I've been hired to teach (outside my own) has focused on automated attack types. Most courses teach the classical hacker methodology (covered below) by teaching students how to be would-be attackers, and then teach how to defeat the hacker menace. Students love becoming ethical hackers. I love teaching them. But it really doesn't do much to make their systems more secure.

Some readers may ask if it makes a difference whether we are defending against a dedicated attacker or automated malware. Yes! First, it is very difficult to stop dedicated manual attackers. Usually, they are well trained, methodical, and patient. You would have to ensure perfect security on every computer you manage to keep them out. Hackers only need to find one hole, one unpatched machine, one gullible end user — and your network is theirs. Dedicated hackers can change their attack methodology based upon what they learn during the early course of the attack. If they start out attacking your web server or router and come across your even weaker SMTP server, they can change their attack on the fly. Automated malware can't do that. It does only what it was predefined to exploit. Small defense changes can completely defang an automated attack tool.

Simple things, like renaming the Administrator account or changing the default port number on a service, will defeat automated malware. A dedicated hacker can use *anonymous enumeration* (covered in the forthcoming chapters) to find a renamed Administrator account, and a hacker can easily port scan a particular host and then do connection probes to find out where you moved a particular service's listening port. Automated malware can do the same thing, but they are rarely investigative. For instance, I have one of my honeypot Microsoft SQL Servers listening on the default TCP/IP ports of 1433 and 1434. It receives nearly a hundred SQL brute-force password guessing attempts a day. Another honeypot running the same version of Microsoft SQL on a non-default port has been up for over two years with zero attacks. It has only received three probes at the redirected ports, none of which were SQL-based. So, forget the

conventional wisdom — security by obscurity as a defense works, and works well for automated malware!

Anyone that says otherwise isn't fighting the right problem or practicing critical thinking. For example, if I move a service's default listening port to something other than its expected value, how could I be doing anything but strengthening security? I liken it to a house that is covered with entrance doors on all four externally facing walls. Only one is the right door that will allow entry into the house. If a thief shows up to check out the entry door (i.e., a port scan) to determine whether the door is unlocked (i.e., a vulnerability), by having multiple doors I've increased the thief's workload by a multiplication factor equal to the number of additional doors. Automated malware usually only looks for one door, and only in front where it expects to find it. A dedicated manual attacker can check all the doors until it find the right one. Automated malware can do the same thing, but I've yet to run into the worm or virus that ever looked for a non-default service port or checked for any admin account that wasn't named Administrator or root. Chapter 2 covers more unconventional thinking.

## Remote Versus Local

Another big attack distinction that is significant to attackers and defenders alike is whether the attack can be accomplished remotely or must be executed locally by the end user. Remote attacks without any end user intervention are the most devious kind. The attacker runs a manual or automated program from a remote location and takes over the user's computer. *Buffer overflows* are the most common example of this type of attack. Most security experts worry about remote attacks much more than local attacks because they can, if coded right, exploit every vulnerable machine the rogue code can contact very quickly. SQL Slammer worm anyone?

Fortunately, most malware requires some input from the end user in order for the exploit to occur. As Microsoft's first law of the Ten Immutable Laws of Security (`www.microsoft.com/technet/archive/community/columns/security/essays/10imlaws.mspx`) states, if a bad guy can persuade you or a program on your system to run his program, it isn't your computer anymore.

The vast majority of malware requires that end users (or their computers) be tricked into executing the code locally. E-mail worms, which, according to the ICSA Survey mentioned above, make up 92% of all automated malware attacks, almost own this category. Most e-mail worms are transmitted in malicious e-mail attachments, although a smaller but growing category tricks the user into accessing the malware through a rogue Internet link. E-mail worms can also use malformed attachment formatting, embedded scripts, and other HTML auto-run vulnerabilities to accomplish their malicious deeds.

If we could get all end users to stop clicking on every file attachment or HTML link they get in e-mail, our current malware craze would be almost non-existent. But end users will always ignore the security advice you give them and there will always be at least one person on the network who will execute every file attachment no matter how often warned, so you need to keep reading this book.

Tricking the user into running the malware program can also be done by constructing a malicious web link and embedding it into an HTML e-mail. Most e-mail programs readily execute HTML content, so the malware program or commands can be auto-launched simply by a user opening an e-mail. As we will cover in Chapter 11, *Protecting E-mail*, the two best things you can do for e-mail users is to block malicious file attachments and disable HTML content.

One of my favorite examples of an embedded HTML link trick is when a malicious web link executes a program the user isn't even aware they have installed, like an instant messaging or telnet client. The link then uses the launched program to carry out further malicious instructions on the compromised system. Over the years, I've seen several malicious HTML links embedded in otherwise normal-looking e-mails that launched previously unused software to install worms and other malware programs.

Perhaps the user doesn't even use Instant Messaging (IM). No bother, the web link starts the program, downloads the malicious file, and then executes it. If the user doesn't regularly use IM, it is almost certain that the IM client hasn't been updated with the recent version and is vulnerable to the exploit. Another example, the Blaster worm, used the Trivial File Transfer (TFTP) program located on every Windows computer to download its main body. No regular users that I know have ever heard of the TFTP program, much less one that allows anonymous file transfers to their computer. That's why I always recommend removing unused software. The software you don't use can still be used to hurt you. We will cover how to stop unauthorized software execution in Chapter 8.

In a related category, if hackers can gain physical access to your computer, they can also execute malicious code locally. This is often the case in privilege escalation attacks, password cracking attacks, and data theft cases. Local attacks can be devastating, but remember that if hackers have local access to your computer, then they can do anything. They can steal the computer. They can douse it in lighter fluid and set it on fire. Fear of local attacks is usually concerned with preventing a trusted insider from gaining higher unauthorized access and privileges.

To summarize the previous information, all computer exploits can be divided into four categories: automated or manual, remote or local. Automated malware allows fast exploitation, but cannot be as creative as a dedicated attacker. Remote exploits are the most dangerous because they don't require end user interaction to execute. Most exploits require that the end users (or their computers) be tricked into executing harmful code locally on the system. With few exceptions, automated remote exploits, such as SQL Slammer or the MS-Blaster worms, have been involved in the fastest and most widespread malware outbreaks.

Lastly, it must be recognized that many, if not most, of today's exploits are a combination of multiple categories. Often a hacker will use automated malware to find computers susceptible to the initial exploit and allow the malware to compromise the computer. The malware then is predefined to contact the hacker (e.g., via a secret IM channel), whereby the attacker can pick and choose his targets at his leisure. Today, it is rare to hear of attacks accomplished solely by a hacker manually typing commands on a keyboard one at a time against a remote host, although they do exist.

# Types of Attacks

There are many different types of attacks, including the dedicated manual hacker, automated malware, and blended attacks.

## *Dedicated Manual Attacker Methodology*

The 1983 movie *War Games,* starring Matthew Broderick, kicked off the world's fascination with computer hacking. Several computer historians point to that single film as launching pad for youth hobbyist hacking. In the movie, Broderick's character, David Lightman, spends his free time *war dialing* (i.e., serially dialing multiple phone numbers one at a time until an active phone number is found, looking for unsecured

modem connections. Along with co-star Ally Sheedy, he ends up infiltrating a military computer and unwittingly almost sets off World War III. It received three Oscar nominations and made stars of the two young protagonists.

Early on, computer hacking often involved war dialing for analog phone connections. That was before the Internet. Now, hackers scan the Internet looking for live Internet hosts and probe them for weaknesses. Here is the conventional methodology of the dedicated manual attacker:

1. Hacker uses a TCP/IP scanning program to find an active host on the Internet. In the early days, hackers manually and methodically pinged random IP addresses until they found a responding host. Eventually, the earlier phone war dialing programs were converted to look for active IP addresses. Note, this type of reconnaissance requires that the remote host has an IP address and responds to a *ping* (ICMP Echo Request). As firewalls and routers began to prevent ping responses in order to thwart hackers, hackers started scanning for open TCP or UDP connections instead.
2. After an active IP address is located, a TCP/IP port scan is done against the remote host. This should reveal any open TCP or UDP connections. If a port is not opened, a TCP connection probe is reset by the remote host or the source IP address is sent a *host unreachable* ICMP message (if a UDP port is involved). Firewalls (in *stealth mode*) will block all probes to closed ports from reaching the protected host, forcing the originating prober's computer to wait for a connection timeout.
3. Depending on the TCP/IP ports found, the hacker can learn what services are running on a particular host. For instance, if port 80 is found, the host is probably running a web server. If port 25 is found, it is probably a SMTP e-mail server. If ports 135, 137, 139, and 445 are found, the host is probably a Microsoft Windows computer (this is covered in more detail in Chapter 2).
4. Once the TCP/IP ports are identified, the hacker will begin to fingerprint the host and its services. The hacker can manually connect to each found open port or use an automated fingerprinting tool. Many times, simply connecting to a port will reveal the service and its version number. Popular fingerprinting tools, such as Nmap (`www.insecure.org/nmap`) or Xprobe2 (`http://xprobe.sourceforge.net`), can be used to identify the operating system. Identifying the operating system helps identify the associated applications. For instance, IIS 6 only runs on Windows Server 2003. IIS 5 runs on Windows 2000, IIS 5.1 only runs on Windows XP, and IIS 4 only runs on Windows NT 4.0.
5. Once a service and its version are identified, a hacker can begin to attempt exploits built for just that particular software version. For example, if the hacker found IIS 5.0, he could search the Internet for all the IIS 5.0 exploits available. Several web sites allow this type of research, including Secunia (`www.secunia.com`). Currently, Secunia shows 11 IIS 5 vulnerabilities (`http://www.secunia.com/product/39`). If the software is fully patched, then the hacker must exploit a misconfiguration or host application error. Apparently, these are not hard to find. According to a November 2004 Gartner Research report (`www.g2r.com/DisplayDocument?doc_cd=124806`), "two out of three `successful` attacks exploit misconfigurations rather than vulnerabilities or missing patches."
6. Finally, the attacker manually hacks his way into the remote computer.

What happens at this point depends on the hacker. If the hacker isn't logged in as an administrator, a privilege-escalation attack may be attempted. Most of the time, hackers spends their next few minutes making sure they can easily break back in when needed. They may set up a new backdoor program, create a new user account, or set up a new malicious listening program. Many times, hackers then close

the hole that allowed them to break in. Surprising as it may be, sometimes the first sign that network administrators receive that they've been hacked is that their systems are fully patched. This was especially notable when many network administrators held off on applying Windows XP Professional's Service Pack 2 (SP2) because of incompatibility issues. They were surprised to find SP2 installed even after they had specifically configured Windows not to install it. It was their complaint to Microsoft about the service pack installing even when requested that it not be that led to the malicious hacker being discovered. I always think it is sad when the hacker does the patching before the administrator.

A fast-growing percentage of professional hackers will inspect the hacked PC to determine whether there is any valuable information to compromise. Often this means they look for credit card or financial information. The hacker may install other malware programs to do keystroke logging or remote control. Before professional hacking became popular, the hobbyist hacker would often use the newly exploited computer as a repository for illegally copied games and videos. Other hackers would set up remote control trojans and start to attack other computers on the Internet. Then, if the secondary hack gets discovered, the evidence trail stops at the intermediary hacked computer instead of at the hacker's true location.

When doing forensic analysis on a hacking crime, two things need to be looked for:

1. How did the hacker or malware initially break in?
2. What did the hacker or malware do after the initial compromise?

The first question will help you prevent future occurrences of the same hack; the latter will help you understand why a particular computer was hacked. Unless the hacker has an interest in a particular company, the found victim computer was probably randomly selected. I don't mean to imply that a dedicated hacker isn't ever trying to attack a particular company. Interesting entities such as NASA, popular web sites, and Fortune 1000 company web sites receive dedicated hacker traffic each day. But most companies aren't being especially targeted by the dedicated hacker, or if they are, it is only by raw luck of the draw. More often, hackers use automated, roving malware to find and exploit vulnerable hosts.

## *Types of Hackers*

Hackers don't come in just one flavor. The vast majority of hackers are called *script kiddies*. These are individuals whose enthusiasm for hacking far outweighs their own hacking abilities. Rarely can they do dedicated manual hacking, keystroke by keystroke. More often they download hacking utilities, tools, and scripts that automate the entire process for them. They just plug in an IP address, and the tool does the rest of the work.

### Don't Run with Knives

Here's a humorous story. In May 2005, I hosted an IIS hacking contest at `www.hackiis6.com`. It was hugely successfully, withstanding up to 250,000 hack attempts and probes per second during the contest. Few of the attacks were interesting. Most attacks came from automated attack tools used by script kiddies. What surprised me was how many of the attacks were meant for Apache web servers — the attacks would never be successful against Microsoft IIS servers. I never knew whether the hackers thought maybe I was kidding when I said it was an IIS server or if they just fired off the only web site hacking tool they could find. Even funnier was another strange story of sweet revenge. One of the popular security mailing lists where budding hackers like to lurk was discussing how to hack my contest site. A participant

> uploaded a script that he said was sure to hack my web site. Even though the script was written in Shell script (a Unix/Linux scripting language), which is not my specialty, I was able to determine that the 13-line script did not attack my web site. Instead it sent the root password of the user who executed it to the script's creator. With a bare minimum of programming knowledge, they could have easily seen that it had nothing to do with my web site and was only grabbing their password and sending it to a remote person. Instead dozens of hacker wanna-bees executed the script. There were even days of e-mails from the script kiddies asking why the script didn't hack my IIS web site even though they were executing it over and over. A few claimed it worked. Too funny!

Although script kiddies aren't very knowledgeable, they probably accomplish most of the hacking that you hear about. They don't need to be experts; the knowledge is built into the tool they are using. They usually aren't very good at covering their tracks and can usually be caught with a minimum of forensic investigation.

Unfortunately, a smaller subset of hackers *are* knowledgeable attackers. They can use and modify a plethora of different hacking tools to accomplish their task at hand, following steps that somewhat mimic the manual dedicated hacker progression listed above. They can switch and modify tools from their defaults to be successful in their compromise pursuit. This type of hacker is persistent and patient. They are smart enough to cover their tracks and can avoid immediate detection. Hackers at this level may write their own malware, but the techniques they use are known. An even smaller subset of dedicated hackers are the master technicians — the überhackers. They develop their own tools and methods for breaking in. Their original malware uses new malware techniques or pushes existing envelopes. They are confident and don't often participate in security mailing lists. If they discover a new hacking hole, they keep it for themselves or sell it to commercial interests. This is the professional hacker and the one defense experts fear most. Although it is difficult to tell how many master hackers there are at any given time, the recent, significant increase in professional computer crime suggests they are on the rise.

> *Microsoft employee Robert Hensing has written a short guide (*`http://weblogs.asp.net/robert_hensing/archive/2004/08/09/211383.aspx`*) discussing the various hacker personas, as he calls them. Robert has spent years on the front lines at Microsoft fighting hackers and discovering new malware and hacking techniques. Although he also summarizes the various hacking skills over three levels, his discussions are more in depth.*

Keeping the dedicated manual hacker out of your computing environment is very difficult. You have to be perfect. Every computer attached to the Internet has to be perfectly configured, fully patched, and secured. The patient, dedicated hacker only has to find one hole. But even the dedicated hacker often resorts to using automated malware to speed up the process.

## Automated Malware

The more popular attack method is using automated malware. Most automated attacks can be categorized by the three main parent types: virus, worm, and trojan.

## Virus

*Computer viruses* are malicious programs that must rely on other host programs or code to spread. In most cases, the virus inserts itself into a legitimate host program (or boot sector) so that when the host is executed, so is the virus (see Figure 1-2). The smallest working PC-based virus is called Define (`http://vil.nai.com/vil/content/v_346.htm`). At only 30 bytes, it is the minimalist blueprint of every virus. It finds a host file, adds its own code to the host file (actually, it overwrites the original host file), and closes the file. When the infected host is run, the process repeats. The other 60,000 plus viruses are just more sophisticated versions of the same routine. Some infect boot sectors, others infect program files, and others infect data files (e.g., macro or script viruses). Some infect at the beginning of a file, others at the end, others in the middle, and still others don't infect the original host at all. A small subset of viruses, called *companion viruses* (aka *twins* or *spawners)*, rely on operating system tricks to fool users into executing the malicious code first when trying to execute the legitimate host.

Some viruses print messages to the screen, make funny noises, draw graphics, and mess with print jobs. Others cause damage. They format hard drives, erase files, and corrupt data. A computer virus can do anything that is programmatically possible.

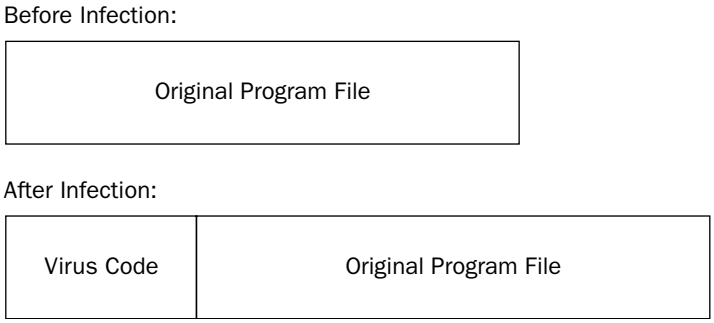


**Figure 1-2**

The first personal computer virus, the Apple-based Elk Cloner, was written by teenager Richard Skrenta (`www.skrenta.com/cloner`) in 1982. Early on, only a few viruses came out each year, mostly for the Apple and Amiga platforms. When the IBM PC started becoming the PC of choice, viruses migrated to the new platform. Although Trojan Horse programs (i.e., trojans) had been around longer than viruses, viruses quickly became the malware writer's program of choice. Before the end of the century, there would be over 60,000 different viruses.

Today, computer viruses aren't as prevalent on the Windows platform because Microsoft created a system file protection mechanism known as *Windows File Protection (WFP).* Ostensibly only created to fight the programming problem known as "DLL hell," WFP will replace any modified, renamed, or deleted Windows system file with a legitimate copy. WFP had the net effect of decreasing the easy spread of computer viruses because when the virus infected a protected file, it was immediately removed.

Along the way, antivirus mechanisms got significantly better at preventing attacks. From 1995 to 2000, macro viruses enjoyed malicious success, but today, Microsoft-enabled programs have, for the most part, defeated the threat. A few years ago, viruses migrated to malicious script files (e.g., VBScript, JavaScript, and HTML applications), but even those threats have been mostly minimized.

## Trojans

*Trojan* programs masquerade to the end user as one type of program (e.g., game, program, etc.), but usually have a hidden malicious agenda. A common trojan ploy is for trojan code to be added to a real, legitimate commercial program. This is often accomplished using a program known as a *linker*. The compromised legitimate program is offered free on the Internet. Users, looking for pirated software, download the program and install it. The program installs as it normally would and this is all the user sees, but the trojan installs "invisibly" behind the scenes.

The defining characteristic of a trojan is that it does not automatically spread or replicate (as does a worm or virus). In order to spread from computer to computer, it relies on different users downloading and executing it prior to being identified as malware.

Trojans are just starting to get the respect they deserve. Trojans often install keylogging programs, which record everything the user types in, including passwords, bank account numbers, and other confidential information. Once a trojan has successfully compromised a computer, the exploited computer can be redirected to do anything. If the trojan infects the PC and awaits further commands from the originating hacker, it is called a *zombie* or *command and control* trojan. For decades, trojans were designed to attack and exploit just the one computer they infected. They would infect the computer and then "dial home" to the hacker and announce their presence. The hacker then would manipulate the single exploited machine. In some cases, the trojan enables the hacker to do anything to the computer that he wants — sort of like a malicious PC Anywhere remote control program. These remote access trojans (or RATs), can record user keystrokes, manipulate files, steal passwords, capture screen shots, and even capture video and sound if the computer is equipped with a working video camera and microphone. The dangerous aspect of a RAT is that even after the malware is gone, the user usually has no idea what the hacker did or learned while the RAT was active. A RAT was the first type of malware program to be able to cause problems well past its removal. Now, multiply that unknown risk by 100,000 or so.

Today, hackers use trojans to compromise as many innocent machines as they can, creating huge networks of compromised machines under the control of a single hacker or group. These *botnets* can then be used do the bidding of the malicious attacker. The hacker may use the compromised machine to attack other targets. If the secondary target attempts to trace back the attack, it ends at the trojan-controlled machine, and not at the originating hacker's machine. Botnets can be comprised of tens of thousands to over 100,000 computers.

In a few months, Honeynet Project researchers (`www.honeynet.org/papers/bots`) tracked more than 100 different botnets, some containing over 50,000 compromised machines. Symantec tracked over 30,000 botnets over a six-month period (`www.securityfocus.com/infocus/1838`). According to Baylor University professor Randal Vaughn, the most popular trojans used to make botnets in 2005 are (in order of popularity) Korgobot, Spybot, Optix Pro, rBot, and other Spybot variants (e.g., AgoBot, PhatBot, SDbots, etc.)

Professional hackers will accumulate as many computers as they can in their botnet and then sell or rent the botnet to the highest bidder. One report (`www.securityfocus.com/infocus/1838`), says a botnet can be rented for *as little as* $100 per hour. Spammers and other professional hackers then use the botnets for their own reasons, including distributed denial of service (DDoS) attacks, spreading spam, hosting malicious web sites, spreading other malware, manipulating online games and polls, and participating in identity theft. Several other companies and researchers have estimated that hundreds of thousands of PCs unwittingly become involved in botnets each month. The company CipherTrust, which tracks zombie nets, estimates (`www.ciphertrust.com/resources/statistics/zombie.php`) that in May

2005, 172,000 new exploited PCs appeared each day. This figure is probably on the high side, but obviously a huge number of PCs are under the control of someone other than their owners. Most of the compromised machines are in the United States or China. Go USA??

Another emerging trojan type is the *rootkit*. A rootkit malware program spends extra effort to hide itself from casual observers and the dedicated forensic inspector. Rootkits use stealth-like mechanisms to hide any sign or symptom of their infection when inquiring eyes or processes query for their existence. Rootkits used to only be a concern for Linux/Unix administrators, but are now moving into the Windows world. There are several web sites dedicated to Windows rootkits, including `www.rootkit.com`. Many security experts are concerned because a properly coded rootkit can be very difficult to discover. In theory, since every PC can possibly be compromised with a rootkit, every forensic investigator needs to take a heavy-handed approach when dealing with an exploited PC, even if it's just suspected of being compromised by a normal virus, worm, or trojan. Although rootkits can be more problematic, the author feels that the antivirus and other security defense companies will rise to the challenge, and rootkits will pose no more harm than yesterday's DOS stealth viruses. For example, you can download Sysinternal's RootKitRevealer (`www.sysinternals.com/utilities/rootkitrevealer.html`) to search for hidden Windows rootkits.

### Worms

As common as trojans are, computer worms are even more so. According to several resources, e-mail-based computer worms make up over three-fourths of all attempted attacks against a PC. A worm is a malicious program that does not need to infect a legitimate host program to replicate—it is self-replicating. E-mail worms usually arrive as e-mail attachments pretending to be something legitimate (in that way, they are like trojans). When clicked, they install themselves in such a way as to ensure that they execute when the computer is rebooted (see "Where Malware Hides," below). The worm uses its own coding to spread. Often the worm will look for the infected user's e-mail address book, retrieve potential e-mail addresses, and then send themselves to the new recipients. Early on, e-mail worms used the user's SMTP server to send themselves. Today, most contain their own SMTP sending engines.

Worms don't need e-mail servers to spread. They can spread using any service or program shared among PCs. Internet worms often scan large swathes of public IP addresses looking for computers to exploit. The SQL Slammer worm exploited servers and workstations running Microsoft's SQL software. The MS-Blaster worm exploited the Windows RPC service. Many worms target web servers. The truth is that a worm can target any listening port and service on a computer. These days, when vulnerability exploit details are released, someone is making a worm to exploit more PCs faster. Whereas we used to have a few weeks or months to patch our machines, automated worms have decreased the low-risk period to a few days or less.

When reviewing a malicious mobile code program, it helps to categorize its overall behavior as a worm, virus, or trojan. A worm can replicate on its own and probably hasn't directly infected other legitimate files. A virus has infected other host files, and those will need to be cleaned up or replaced. A trojan isn't as hard to remove, but what its controlling hacker did while the trojan was in control of the PC is usually unknown. Today, like most e-mail worms, many malware attacks borrow techniques from multiple parent classes. These hybrids, although not new, are becoming more and more common.

## Remote

Remote attacks, which successfully compromise a computer without the end user being involved or making a mistake, concern security analysts the most. The idea that a remote attacker can connect to an innocent remote PC and compromise it is a compelling issue. Common remote attacks include denial-of-service, buffer overflows, misconfigurations, sniffing, and man-in-the middle attacks.

## Denial of Service

Sometimes a computer doesn't have to be exploited or modified to be maligned. A *denial-of-service (DoS)* attack causes a listening port or service on a remote computer to become hung and unusable by legitimate users until it is restarted. Hackers have used botnets for years to bring some of the world's most popular web sites down, including eTrade, eBay, Microsoft, and popular gambling web sites. Denial-of-service attacks work by sending a listening service too much fake traffic or something it is not coded to expect, causing the program to crash or go into an endless loop. For example, an attacker using a large botnet can send millions of packets per second to a single web site, overwhelming it, and making it unavailable to legitimate users. Steve Gibson of Gibson Research Corporation has one of the best accounts of an extended DoS attack (`http://grc.com/dos/grcdos.htm`). His company was held hostage by a 13-year-old kid offended by a quote that was misattributed to Steve. It's a great story full of technical detail, if you have time to read it.

Other types of DoS attacks malform network packets to accomplish their task. For instance, a *LAND attack*, which recently reappeared to successfully compromise Windows XP and Server 2003 (`www.eweek.com/article2/0,1759,1773958,00.asp`) for a few months, sends packets with identical origination and destination IP addresses and port numbers to hosts. The destination host receives the packet and attempts to respond to the bogus origination host, but ends up in an endless loop responding to itself. The scary thing about DoS attacks is that the Internet's primary protocol (IP version 4) is pretty much useless to prevent them. Many web sites have been down for days while fighting DoS attacks, finding and shutting down remote connections from DoS bots one-by-one to filter out the illegitimate traffic from the legitimate.

## Buffer Overflow

The most common type of remote exploit is accomplished using a *buffer overflow*. A buffer overflow attack sends more information to a receiving program than the receiving program is prepared to accept. The receiving program incorrectly accepts the overly large data sent its way and inadvertently places the rogue data (which is composed of malicious machine-language commands) into executable memory where it is subsequently executed as program code by the CPU.

A buffer overflow can simply result in a DoS attack or result in a complete system compromise. If the buffer overflow attack results in a complete system compromise, the attacker gets logged into the remote system in the security context in which the remote service was executing. Unfortunately, most Windows services run in the LocalSystem context, which gives the hacker absolute control of the PC. Buffer overflows always occur because the program and/or the programming language did not control the inputted size of the data being submitted (as they should).

Related to the buffer overflow attack are invalid data *injection* attacks, where applications are sent malformed data that then tricks the application into executing commands instead of the intended normal database actions. SQL injection attacks are the most popular of these attacks, but they occur with PHP, CGI, and dozens of other programs and languages. Injection attacks can only be fixed with program patch updates and better program coding that checks for these malware data types. Drastic changes are being made to Windows and to the supporting computer chips to prevent buffer overflows. Windows XP and Windows Server 2003 have undergone extreme code review cycles (several rounds of manual and automated review) to prevent buffer overflows. Still, patches fixing Windows buffer overflows occur at least every few months.

## Misconfiguration Weaknesses

Contrary to popular belief, it isn't only unpatched software that gets exploited. Dedicated attackers (and penetration testers) often exploit machines through configuration errors induced by the computer administrator, programmer, or application program.

The author of this book regularly sees the following misconfigurations on remote computers:

- The Everyone group has Full Control permissions to all files and folders on a file server.
- The null session Anonymous user credential is given Full Control permissions to all files and folders on a domain controller.
- The IIS Anonymous user account is given Full Control permissions to all files and folders on an Internet-facing PC.

Computers and computer programs are complex, and complexity is the enemy of security. Most computer administrators are overworked and undertrained. Security is just one part of their job. Getting the computer up and getting the applications working for the end user is job number one. In their rush to satisfy multiple user requests, it is easy to see how administrators could misconfigure a computer. All it takes is answering yes to one unknown message box and Windows will do the rest. Misconfiguration errors can only be resolved by better training, configuration management, more secure software defaults, and configuration reviews.

## Sniffing

Network protocol analyzer programs (i.e., *sniffers*) capture network packets crossing through their network interface card. By default, most network interface cards will drop network traffic not intended for them. A sniffer modifies the network interface card's network protocol stack so that the sniffer program can capture every packet that the card sees (called *promiscuous mode*). An attacker using a sniffer can capture anything that passes along the network in unencrypted form, including passwords and confidential information. Sniffing attacks were much more common in the early days of local area networks, where Ethernet hubs passed all information headed to and from any host on the hub to all other hubs. Ethernet switches replaced dumb hubs starting in around 1995. An Ethernet switch, by default, allows each connected PC to see only the traffic headed to and from it (plus broadcasts). This effectively killed the eavesdropping sniffer for about 15 years as a popular means of attack.

Today's proliferation of unprotected wireless local area networks (WLANs) has lead to a huge resurgence in wireless sniffing attacks. A remote attacker, listening in with a portable and small computing device, can listen in on any unprotected wireless network. Unfortunately, even the first popular wireless encryption protocol (Wired Equivalent Privacy) wasn't all that secure. Today, anyone can walk around and find dozens to hundreds of hackable wireless networks, which they can then exploit to capture confidential information, steal free bandwidth access, or inject malicious code.

A more sophisticated form of the sniffing attack is a man-in-the middle (MitM) attack. After establishing an eavesdropping session between two hosts, a MitM attacker can forge his identity to appear as the other host to each participating host. The MitM hacker can then capture information, or even more dangerous, manipulate the communication session to change the data sent between sender and receiver (without either innocent party being aware of the change). There have been some scary MitM attack demonstrations. The hacker program Cain & Able (`www.oxid.it/cain.html`) allows MitM attacks to be carried out with a few clicks of the mouse. Using the tool, an attacker puts the program into sniffing mode and captures the IP and MAC addresses of the machines it can listen to. Then, by simply clicking the mouse a

few times, the Cain & Able user can initiate a MitM session. I've seen Cain & Able used to successfully intervene in SSL and RDP encrypted sessions. To prevent MitM attacks, the two hosts (or their programs) must use some sort of session authentication protection (e.g., IPSec, asymmetric cryptography, etc.).

## Other Types of Attacks

Other types of attacks include physical, insider, obscurity, directory transversal, password cracking, social engineering, adware, spyware, spam, phishing, and pharming.

### Physical

It's important in any computer security defense plan to consider physical security. If attackers have physical access to a computer, they can do anything to it. They can install hardware keyloggers, steal files, crack passwords, and just about anything else they want to do. They can steal the PC or set it on fire. Physical security is covered in Chapter 2.

### Insider

Many surveys claim that trusted insiders account for 70-80% of all corporate computer crime. It makes sense that a trusted authorized employee, with intimate knowledge of a particular computer, has a strong chance of compromising that system, compared to the unknowledgeable outsider. But according to a May 2005 report prepared for the Secret Service (`www.secretservice.gov/ntac/its_report_050516.pdf`), insiders account for only 29% of all successful intrusions. Current, former, and contract employees were the most common insiders to commit a computer crime. In 62% of cases, a negative work-related event trigged the crime. In 82% of cases, the individual "acted in a concerning manner in the workplace prior to carrying out the crime." More than a third of them had already been arrested for other crimes. When hired, most had authorized, highly privileged access, but less than half did at the time of the crime. The majority of them used remote access to gain access to an unauthorized account, created a backdoor, and then launched a malicious attack. What does this say? A disgruntled employee or contractor is responsible for most insider attacks. Your security policy should take these statistics into account.

### Obscurity

Many attacks work by fooling users or their programs into believing they are headed to a legitimate web site or network location, but instead redirect them to a malicious location. Obscurity attacks also used to fool computer security defenses. The simplest obscurity attack might be a user receiving an e-mail directing her to `www.microsoft.com.site.com/downloads` to download a proposed Microsoft patch. Those of us who understand URLs understand that the example URL would direct the user to a computer located on `site.com`, not `www.microsoft.com`. But most end users don't know the details of how URLs work and would probably be fooled. Typical obscurity attacks are even cleverer than the simple example. Usually the e-mail would be full of legitimate official links, and the `site.com` portion of the URL would be an IP address or a hexadecimal encoded (or decimal dot notation) location instead.

Obscurity attacks fool even the most secure of devices. Frequently, content and links that would normally not be allowed are passed through computer security defenses only to be converted into their legitimate locations on the computer endpoint. This is because many Internet standards require that inspection devices, computers, and browsers be able to take instructions and commands in many forms besides the normal ASCII form. This leads to inherent conversion problems of which the hackers take advantage. These types of attacks can be difficult in the short term to defeat, as hackers have been ingenious in creating ways to bypass the created defenses. For example, in 2005, there have been several obscurity attacks

against many of the common Internet browsers. Browser developers responded by developing special tools that would alert the user when the presented URL did not match the web site to which they were being redirected. Enterprising hackers coded their malicious web pages to post "messages" that looked like the user's normal screen over the resulting warning messages. Very tricky indeed.

### Directory Transversal

Directory transversal attacks are similar to obscurity attacks in that they attempt to use an obscure or encoded representation of a directory path in order to access a directory location or file to which they would normally not have access. For example, an older (now patched) attack, `http://hostdomain/../../../../../../../../windows/system32/cmd.exe?/c+dir+c` will be converted to `c:\windows\system32\cmd.exe`. In unpatched versions of IIS 5, it would allow a remote attacker to have command-line shell access to the web server even though IIS is specifically coded to prevented anonymous access to that location. All of the popular web servers and web browsers, and many computer security devices, are susceptible to these types of attacks. The vendors close up one hole only to have another discovered a few months later. The defenses against these types of attacks will be covered in chapters 10 and 12.

### Password cracking

If an attacker can gain physical access to a Windows computer in its default configuration, its password database is very susceptible to attack. Remotely, an attacker is usually forced to guess at user logon names and passwords to be successful. Unfortunately, most networks are full of many weak passwords that are easy to guess. Chapter 4 will cover password cracking thoroughly.

### Social Engineering

Sometimes the easiest hack isn't a hack at all. The world's most controversial and popular culture hackers have all been masters of social engineering. Instead of using a malware program or trying to manually break into a system, they just ask for the necessary information. Most people's overly helpful nature leads to an unnatural trust that social engineering hackers exploit. Social engineering doesn't always require elaborate ruses. Frequently, company receptionists are called by external hackers claiming to work for the telephone company. They ask the helpful receptionist for an outside line to help troubleshoot a problem. The hacker then makes free long-distance calls at his leisure. Often, when I've been hired to do penetration testing, I'll walk up to the CEO's assistant and explain that I've been hired to do penetration testing, including password strength testing. I then ask for the CEO's password to test it. I've never not gotten the password. The only defense against social engineering is good computer security policies and employee training.

### Adware, Spyware, Spam, Phishing, and Pharming

Spammers and mischievous advertisers are making the virus, worm, and trojan problem of the past two decades seem like child's play. Spam e-mail is more common on the Internet than legitimate e-mail. An unprotected, unpatched PC will often have dozens to hundreds of installed adware, spyware programs and tracking cookies after a few days of surfing the web. Some "free" software programs install hundreds of malicious programs that the user is unaware of as a cost of using the free program. Adware and pharming attempt to direct the user to a predefined location instead of where the user intended to go in order to sell a product. Adware usually does this by modifying the user's browser configuration or DNS client information (see "Where Malware Hides," below). Pharming (`http://en.wikipedia.org/wiki/Pharming`) tricks DNS servers or clients into having fraudulent information in order to misdirect end users. Pharming attackers are interested in either Adware-like activities or stealing the user's identity. Phishing attacks send malicious, but legitimate-looking, e-mails to users asking them to re-verify their credit card information. Phishing e-mails look quite realistic, even often reminding users not to be fooled by phony phishing e-mails at the same time as they scam the user.

Spyware attempts to capture a user's personal information in order to sell the information to dubious vendors or criminals.

All five types of attacks previously mentioned are on the rise, despite increased legislation and better computer defense tools. They install themselves in elaborate, multi-angled attacks, which make it difficult to remove. They have software installers that install other software installers that install dozens of programs, each from multiple, moving malicious web sites. Trying to track down who is spreading the spam, spyware, or phishing attack is one of the most difficult computer forensic challenges possible. It takes massive coordination of several networks and security administrators to capture a single bad guy.

> *Read the multi-part SANS Handler's Diary article, "Follow the Bouncing Malware" (*`http://isc.sans.org/diary.php?date=2004-07-23&isc=00ee9070d060393ec1a20ebfef2b48b7`*), to get a better understanding of how difficult the task can be.*

## Malware Trends

There are three major trends in malware today. Attacks are becoming more:

- ❑ Professional, with criminal intent
- ❑ Blended
- ❑ Self-updating

This means the attacks are getting more sophisticated, more secretive, and more likely to cause financial harm. The single biggest change in malware during 2005 was the evolution of malware from a teenage hacker's hobby to a professional criminal enterprise. Almost all the malware released in 2005 attempts to steal personal or financial information. Symantec's Internet Security Threat Report VIII (`http://enterprisesecurity.symantec.com/content.cfm?articleid=1539`) reported that 74% of the most popular malware programs monitored in 2005 had the ability to steal information. This is a tremendous change from the past, when most malware programs were created for bragging rights or to upload pirated software. The use of malware programs to commit computer crime is so prevalent that a new malware classification, *crimeware*, has been coined to accurately state the intent.

Second, attacks are often are blended to include automated portions directed by a dedicated attacker. What starts out as a trojan ends up writing a worm that "drops" a virus. The virus downloads a spam bot, and the cycle continues. Lastly, many viruses and worms are pulled from one or more centralized (often illegally compromised) web sites. Once these malicious web sites are recognized, the owning ISP can be identified and they can shut down the web site.

Today, the malware searches for victim web sites to compromise. Then a worm from the first compromised web site starts to infect client machines. The subsequently infected client machines reconnect to the original *mothership* malware web site and download new malware. The new malware instructs the client to connect to another malicious web site, where it downloads a new malware program, and so on. Using this method, the malware mothership web site is always changing. By the time authorities shut down one malicious web site, a hundred more are compromised. And the code downloaded from each malware web site changes with each version, so the malware is self-updating. The originating hacker can drop off new malware programs with different infection and damage routines anytime they want to.

Even with all the attack types mentioned previously, we are just discussing the tip of the iceberg. There are literally hundreds of attack types and more than 100,000 different malware programs. Malware is as varied as real programs. Rogue programs are becoming more complex, and will follow technology wherever it goes. Today, malicious e-mail attachments and embedded links are the most popular malicious code types. In the future, wireless worms or rogue XML code will probably take center stage. It is doubtful that we will ever see a complete defeat of malware and a completely safe computing environment. There may be more than 100,000 programs, but they use the same few dozen compromise methods.

# Where Malware Hides

With few exceptions (e.g., notably the SQL Slammer and Code Red worms), almost every malware program programmatically exploits Windows to ensure its continued survival. Table 1-1 contains the most extensive listing of where malware can hide in a Windows system of any publication. In summary, malware tries to use applications, files, folders, registry keys, and other mechanisms that are automatically executed when Windows (or another common program) starts. The forthcoming lessons in the remaining chapters are based upon defending Windows computers against these common malware exploitation techniques.

**Table 1-1**

| Area | Name | Function | Notes |
|---|---|---|---|
| Application | Archive files | Malware can be hidden or launched from within archive file formats. | Archive file formats, such as PKZIP, Cab, Stuff-it, and Tar, manipulate/obscure the original file and can allow malicious files to bypass detection mechanisms.<br>Malware files can be hidden in nested archive files, and won't be detected unless detection mechanisms use recursive scanning; even then the key is how "deep" the recursive scanning will try.<br>Denial-of-service attacks and detection bypass have been successfully caused by overly large uncompressed file names, overly "deep" directory structures, etc.<br>Exploded archive files have also be used to overwrite other legitimate files in directories the user did not intend. |

| Area | Name | Function | Notes |
|---|---|---|---|
| Application | Auto-run application files | Malware can launch from any auto-running file associated with a particular application. | Examples include MS-Office auto-run macros.<br>Archive files can also have auto-run files executed after the archive is opened. |
| Application | Embedded or linked files | Many applications and their file formats allow other document types to be embedded/executed. | For instance, MS-Word files can have MS-Excel files embedded that are automatically executed when the Word file is opened. |
| Application | Microsoft Word | Embedded scripting can be used to manipulate remote file systems—to write over, copy, and delete files on the system which opened the MS-Word file. | It has been demonstrated that a maliciously crafted MS-Word file can secretly send a named document to a remote intruder. |
| Application | Cross-site scripting (XSS) | HMTL-based forms and e-mail allow malicious scripts to be embedded by a rogue hacker and executed on other computers that innocently view the HTML code. | Very common malware vector. Most HTML-based e-mail services have been the victim of one or more cross-site scripting attacks. Has also affected many web sites, blogs, and databases. Can only be defeated when the HTML-based service prevents the insertion of malicious scripting into input fields that are later displayed to other viewers. |
| Application | Outlook | Malware can manipulate Outlook to send other recipients malicious e-mails. | Can be done by malware becoming an add-in (e.g., Hotbar adware).<br>Can be done by manipulating SMTP server settings or the HOST file and intercepting sent e-mail.<br>Can be done by adding malicious script as an unauthorized e-mail signature (ex. JS.Fortnight worm). This exploit occurs more in Outlook Express than Outlook. |
| File | Alternate Data Streams | Malware can hide itself in the Alternate Data Streams (ADS) of a Windows file. | ADS example: regularfile.exe: malware.exe<br>If executed, ADS process (i.e., malware .exe) will appear as regularfile.exe in Task Manager.<br>No built-in Windows utility to show ADS files, but many companies, including `www.sysinternals.com` and Microsoft (Resource Kits), have tools to do so. |

*Table continued on following page*

| Area | Name | Function | Notes |
|---|---|---|---|
| File | Any executable | Viruses can modify any executable, script file, or macro file to run. | Works in DOS and any version of Windows.<br>Microsoft system executables cannot be modified in Win ME and W2K and above because of Windows File Protection (System File Protection) in Win ME. |
| File | Autoexec.bat | Loads real-mode programs prior to Windows loading | Works with DOS, Win 3.x, and Win 9x. Replaced by Autoexec.nt in NT and later, and even then only gets called when a DOS session is started. Stored in root directory.<br>Not commonly used by malware today.<br>If used by malware, malware often inserted dozens of blank lines to the end of the file and pushed malicious commands below the normal viewing area of the file to fool inspectors.<br>Win 9x looks for Autoexec when it starts, not necessarily Autoexec.bat; so an Autoexec.com or C:\Autoexec.exe could be run instead.<br>Other variations relevant to Win 9x are C:*.DOS, C:*.WOS, C:*.W40, and C:*.APP files. |
| File | Autoexec.nt | File allows real-mode programs to be associated with specific 16-bit or 32-bit command shell sessions. | Works with NT family.<br>Stored in %windir%\system32.<br>Not common with malware. |
| File | AUTORUN.INF | Autorun file; runs commands or programs referenced by open= or shellexecute= after inserting (or choosing to Autoplay) media storage (i.e., CD-ROM discs). | Works with Win 9x and later, and can work with any type of media. By default, it doesn't work with most USB memory keys.<br>Media that works with the Autorun.inf file can be modified using registry edits and third-party apps (such as TweakUI).<br>Not widely exploited by malware, but the potential exists. By default, hard-drive volumes are enabled for Aurorun.inf processing. |

| Area | Name | Function | Notes |
|---|---|---|---|
| File | Batch or Command files | Will run listed programs or commands | Batch file viruses will search for these types of files to infect.<br>Although not rare, most malware programs do not use this vector anymore. Windows fails to verify file content when opening a .BAT, .CMD, or .PIF file, so if a raw code .EXE is renamed as any of these, then it will still run as raw code. This is not the case with .LNK file, which is a threat only in that it is a shortcut that can be used to execute other files or load web sites. |
| File | Boot.ini | File used by NT OS family to determine which OS to load and where OS is located on disk | So far, not successfully manipulated by malware, but is sometimes the target of payload damage attacks. |
| File | Bootsect.dos | DOS boot sector on NT systems that dual boot with earlier versions of Windows or DOS | Could be infected by viruses in early versions of Windows and DOS.<br>Pointed to by Boot.ini file in dual-boot scenarios in NT and later.<br>In reality, any type of code can be referenced to run in the Boot.ini file (e.g., Recovery Console).<br>Not widely exploited by malware. |
| File | Command.com | Default DOS shell in Windows 9x and earlier | Could be infected by viruses in early versions of Windows and DOS. Not possible in Win ME and W2K and later because of Windows File Protection. |
| File | Config.nt | File allows real-mode programs or drivers to be associated with specific 16-bit or 32-bit command shell sessions | Works with NT family.<br>Stored in %windir%\system32.<br>Not common with malware. |
| File | Config.sys | Loads real-mode programs or drivers prior to Windows load | Works with DOS, Win 3.x, and Win 9x. Replaced by Config.nt file in newer OSs.<br>Stored in root directory.<br>Not commonly used by malware today. If used by malware, malware often inserted dozens of blank lines to the end of the file and pushed malicious commands below the normal viewing area of the file to fool inspectors. |

*Table continued on following page*

| Area | Name | Function | Notes |
|---|---|---|---|
| File | Desktop.ini | Used to customize folder behavior. It is meant to allow users to customize folder appearance and behavior, but can be used to hide files and auto-launch programs when referred to folders are viewed. | Several worms (ex. WuKill, Rusty, Opposum, and Expobot) use Desktop.ini to launch their malicious executables when a related folder is viewed.<br>Can be used to hide files and auto-launch programs.<br>Desktop.ini is usually marked hidden.<br>Folder.htt is used instead of desktop.ini when desktop is in "Web view".<br>MSDN link:<br>http://msdn.microsoft.com/library/default.asp?url=/library/en-us/shellcc/platform/Shell/programmersguide/shell_basics/shell_basics_extending/custom.asp |
| File | DOSSTART.BAT | Would load listed real-mode programs when starting a command prompt session or when booting to a command prompt session during Safe mode | Works with Windows 3.x and Win 9x family.<br>Located in %Windir%.<br>Superseded by registry key. |
| File | HOSTS | Used to place static DNS resolution entries | Works with Win 9x and later.<br>Located in %windir%\System32\drivers\etc in NT and later.<br>Malware or adware will often modify this file to redirect a user or program to a bogus location when the associated DNS entry is queried. |
| File | IERESET.INF | Used as the "initial" values when Internet Explorer is *reset*. Can be manipulated to place malicious entries. | Not used in the wild, yet.<br>Proposed by Andrew Aronoff of SilentRunners.org.<br>Default security is Read & Execute by normal users; requires Admin rights to modify. |
| File | LMHOSTS | Used to place static NetBIOS resolution entries | Works with Win 9x and later.<br>Not commonly used by malware, but could be modified to do bogus NetBIOS name resolution redirection.<br>Located in %windir%\System32\drivers\etc in NT and later. |

| Area | Name | Function | Notes |
|---|---|---|---|
| File | Msdos.sys, Io.sys | Default boot files in earlier versions of Windows and DOS | Could be infected by viruses in Windows 3.x and DOS.<br><br>In Win 9x, Msdos.sys is used as an editable configuration file, not as a system file.<br>In Win 9x, the original Msdos.sys and Io.sys files are renamed Io.dos and Msdos.dos. If you boot to DOS with Win 9x, the files are renamed Winboot.sys and Msdos.w40. Could end in other extensions, including .Wos and .App.<br>In Win 9x, if Winboot.ini exists (it is normally deleted by the OS after a completed setup), then it can override the use of Msdos.sys.<br>Not used in NT, 2000, and later, but may be present because of upgrades or dual booting situations.<br>Not very dangerous these days. For Win 9x, also C:*.DOS and C:*.W40, which toggle these into active status via the F8 boot menu's "Previous MSDOS" option. If a C:\WINBOOT.SYS exists, it is automatically copied over C:\IO.SYS when a Win 9x boots; As far as I know, this hasn't been used by malware, much to everyone's relief. If a C:\WINBOOT.INI exists when a Win 9x boots, then it is processed instead of C:\MSDOS.SYS. |
| File | Normal.dot or any .dot file | Microsoft Word document template | Used by macro viruses.<br>Not commonly manipulated anymore because default MS-Office security minimized success of macro viruses. |
| File | Ntldr | NT family OS boot code loader | So far, not successfully manipulated by malware, but is sometimes the target of payload damage attacks.<br>Protected by Windows File Protection. |

*Table continued on following page*

| Area | Name | Function | Notes |
|---|---|---|---|
| File | OLE2 document trick | OLE2-formatted documents can be executed no matter what their file extension | Many applications, especially Microsoft applications, use the OLE2 file format, including Microsoft Office applications, MSHTA, SHS, and SHB files.<br>Files with an OLE2 format will be run by the related application (as indicated by the OLE2 file's embedded OLE2 Root Entry CLSID value) regardless of the file name or extension. Thus, harmless.txt could really be a macro virus or hta malware script.<br>The OLE2 file format is also known as Compound Document file format.<br>OLE2 documents are essentially their own little file systems ("file system within a file"), resembling something like a FAT disk subsystem with its own root entries and subsections and files.<br>The OLE2 trick is used in the wild by spammers, etc.<br>The Root Entry CLSID can be found in OLE2 files following the string label R.o.o.t. .E.n.t.r.y. |
| File | Rasphone.pbk | Can be used to modify dial-up network settings, including which DNS servers (IpDnsAdress and IpDns2Address) the dial-up connection uses and to place unauthorized long-distance calls. | Located in %UserProfile%\Application Data\Microsoft\Network\Connections\Pbk folder.<br>Don't forget to look in AllUsers profile.<br>Trojans and malicious "Dialer" programs frequently manipulate this phonebook file, including Flush.D trojan and HotPleasure Dialer.<br>Can be present with Windows 9x and above PCs.<br>Key is not present (or a threat) unless you use Dial-up networking. |
| File | SYSTEM.INI [boot] scrnsaver= | If referenced by 16-bit Windows applications, will load the screensaver listed | Works with Windows 3.x and Win 9x family.<br>Located in %Windir%.<br>Screensaver files usually end with .SCR, .EXE, or .DLL extensions.<br>Common malware vector in the Win 9x days.<br>Replaced by registry entry in the NT family. |

| Area | Name | Function | Notes |
|---|---|---|---|
| File | SYSTEM.INI [boot] shell= | If referenced by 16-bit Windows applications, will load command shell listed (e.g. explorer.exe). | Works with Windows 3.x and later.<br>Located in %Windir% .<br>Only referenced by 16-bit Windows programs.<br>Superseded by registry entries in NT and later. |
| File | WIN.INI [windows] load=, run= | If referenced by 16-bit Windows applications, will execute programs listed. Run= loads programs in maximized state, load= runs programs in minimized state | Works with Windows 3.x and later.<br>Located in %Windir%.<br>Only referenced by 16-bit Windows programs.<br>Superseded by registry entries in NT and later. |
| File | Wininit.ini | Contains pending file operations (e.g., rename, copy, etc.) to be executed on the next reboot of Windows | Works with Win 9x and NT, but not in W2K or later.<br>Located in %windir%.<br>Replaced by registry key in later version of Windows.<br>For more information, see `http://support.microsoft.com/kb/140570`. |
| File | Winsock.dll or Winsock2 service provider dlls | Used by Windows for network communications | Often used by trojans for their dirty work.<br>Usually located in C:\%Windir%\System32 and protected by Windows File Protection in Win ME and W2K and later. Trojan versions may be located elsewhere (e.g., %Windir%\System or %Windir% folder).<br>Trojan Winsock service providers can be added to Windows and can manipulate any network communications.<br>Can be removed by Winsock service provider cleaners, such as Lsp-fix. |
| File | WINSTART.BAT | Would load listed real-mode programs prior to Windows loading or when user exited command prompt session. | Works with Windows 3.x and Win 9x family.<br>Located in %Windir%.<br>Superseded by registry key. |

*Table continued on following page*

| Area | Name | Function | Notes |
|---|---|---|---|
| Folder | %Windir%\Favorites*.url<br><br>%UserProfile%\Favorites*.url<br><br>%Windir%\Favorites\Links*.url<br><br>%UserProfile%\Favorites\Links*.url | Lists Favorites in Internet Explorer | Often manipulated by adware, but has also been manipulated by malware |
| Folder | %Windir%\Start Menu\Programs\Startup<br><br>%Windir%\All Users\Start Menu\Programs\Startup<br><br>%USERPROFILE%\Start Menu\Programs\Startup<br><br>%ALLUSERSPROFILE%\Start Menu\Programs\Startup | Default Startup folders; any program or command listed in one of these folders will be automatically executed when the user logs on | Works with Win 3.x and later, depending on default location for the particular version of Windows.<br>Default is C:\Documents and Settings\%userprofile%\Start Menu\Programs\Startup in Windows 2000 and later.<br>Default is C:\%windir%\%user profile%\Start Menu\Programs\Startup in NT.<br>Default is %windir%\Start Menu\Programs\Startup in Win 9x family.<br>Startup folder location determined by registry key.<br>HKLM\Software\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders. |
| Folder | Recycler | Recycle Bin's temporary storage location for deleted files and folders | Often used by malware to store malicious code.<br>Earlier versions of antivirus scanners would often skip the Recycle Bin storage area, and hence, escape detection. |
| Folder | System<br>System32<br>%Windir% | Malware often writes itself to Windows system directories | Non-admins usually do not have permissions to write to System folders.<br>In Win ME and W2K and later, because of Windows File Protection, legitimate system files cannot be overwritten, deleted, renamed, or modified, but new files can be written if the program has Write access.<br>By default, most users have Read & Execute permissions to System folders. |

| Area | Name | Function | Notes |
|---|---|---|---|
| Folder | Tasks | Lists Task Scheduler Tasks | Works with Win 3.x and later.<br>Located in %Windir%. |
| Folder | Temporary Internet Files | Malicious files are often stored/hidden in Internet Explorer's Temporary Internet Files (TIF) folder. | In 2000 and later, TIF is C:\Documents and Settings\<logonname>\Local Settings\Temporary Internet Files.<br>Can be modified in Internet Explorer.<br>If malware exploits System account (i.e., using a buffer overflow) and uses IE or Wininet APIs, the TIF location will be located under the Default User or Network Service profile directories (which are hidden by default).<br>Some web browsers will have their own web caches that may hold the "as-arrived" form of malware dropped by web sites, as well as potentially exploitable application startup axes and/or settings locations. |
| Other | ActiveX Control | Installed ActiveX Control | If already installed, may be able to re-install other malware/spyware automatically even after removal.<br>May need to set Kill Bit to defeat. |
| Other | Executable pathway | The PATH statement determines which paths OS should try if the file is not found in the default directory it was called from (i.e., Frog.exe vs. C:\Program Files\ Frog.exe) | Was a bigger problem in the latter days of DOS (.bat, .com, .exe).<br>Some malware programs (ex. Spawner or twin viruses) rely on defects in the way Windows executes files when only the relative file name or path is given (ex. Frog.exe vs. C:\Program Files\Frog.exe).<br>The PATH statement can be set by the DOS PATH command (located under Environment variables in NT family) or by the registry key.<br>In Win 9x and earlier, autoexec.bat file could be modified to change the PATH statement.<br>Can still be a problem today. |

*Table continued on following page*

| Area | Name | Function | Notes |
|---|---|---|---|
| | | | For example, some malware places itself in default application directories, which the application executes instead of the legitimate program executable. One trojan placed its malicious code in the user's My Documents folder. Because the malware was named after a legitimate MS-Word executable, MS-Word would always load it first instead of the legitimate version located under Program Files. More detail on path-spoofing: Set statements in Config.sys can define the PATH too, as can DOSStart.bat and DOS mode .pif for Win95 and Win98. Additional extensions may be set up as "executable" via file associations, and precedence override set by PathExt environment variable and registry setting in NT. Registry AppPaths, and possibly other locations where code overrides can be effected, may offer opportunities to spoof "companion" code into place. FaberToys (`www.faberbox.com`) is a free tool that includes program aliases as one of the integrations it lists. Any executable can be run as an associated "batch file" via a .PIF. |
| Other | Hidden files | Hidden (or system) files/folders will not appear to casual searches. | Dir *.* /ah /s will search and reveal all hidden files. Many legitimate files are marked as hidden or system. Mostly concerned with hidden executables, script, or batch files in root, %Windir, or System32. You can use Windows Explorer or Attrib.exe to unhide files. |
| Other | System Restore | XP/ME Restore feature may inadvertently restore malware located in older restore copies. | Most AV and malware remove software programs suggest turning off this feature prior to any active cleanup. Enabled by default, and usually a good thing to have running unless you need the storage or CPU resources. Can be enabled or disabled manually, by regedit, or by GPOs. Note that WinME's Wininit.exe has inherent SR functionality that will populate the SR subtree even when SR has been disabled. |

| Area | Name | Function | Notes |
|---|---|---|---|
| Other | Task Scheduler | Will run listed programs and commands | Sometimes used by malware to reload malware at a predetermined time interval or to gain initial access.<br>Some scheduled tasks are run in the System context, allowing privilege escalation attacks. |
| Other | Trusted Publisher | Vendors listed here can execute programs without prompting for end user approval. | Be very cautious about which vendors are listed here, as it allows them to execute any program without approval from the end user. |
| Other | Unusual folder/ file names | Hackers and malware often use unusual names to hide malicious files and folders. | Some tricks fool Windows-GUI, some command prompt, some both.<br>Be wary of soundalikes (svchosts.exe, win.exe, win32.exe, service.exe, users32.dll, etc.).<br>Be wary of legitimate file names located in the wrong directory (e.g., svchost.exe located in %windir% instead of System32).<br>Overly long file names that make the file name appear to be blank or push the file name or extension offscreen.<br>Files with multiple extensions (e.g., malware.txt.ext).<br>Files with incorrect extensions can still be executed at the command prompt.<br>Files with nonstandard character sets (`http://weblogs.asp.net/robert_hensing/archive/2005/01/10/350359.aspx`).<br>Isoglyph "puns," e.g., reversed-case EXPiORER.EXE, Unicode tricks.<br>Files with incorrect extensions (i.e., a readme.txt that is really a .dll file or vice-versa).<br>In Windows 2000 and NT, ADS code is shown in Task Manager with the parent file's name instead, and may spoof past firewall per-application monitoring.<br>Various registry settings will cause code in an incorrectly extensioned file to be run as raw code, even when the Windows generic "open" would not have failed to exclude it. |

*Table continued on following page*

| Area | Name | Function | Notes |
|---|---|---|---|
| | | | Registry content that is "too long" will not be shown via Regedit.exe but will run anyway; LVNSearch (`http://isc.sans.org/LVNSearch.exe`) is a free tool that seeks such exploits.<br>Files with invalid dates (i.e., before 1/1/1980 or well into the future). Windows Search GUI's date filter will not find files with dates before 1/1/1980. |
| Other | URL Monikers | URL Monikers can be added to Internet Explorer to load associated programs when a particular keyword is typed. | Internet Explore can be modified to allow keywords typed in the URL to launch associated programs.<br>Also known as URL handlers.<br>For more information, see `http://msdn.microsoft.com/library/default.asp?url=/workshop/networking/moniker/monikers.asp`<br>Malicious coded web sites or HTML e-mails can launch and maliciously manipulate local programs using URL monikers.<br>For example, AOL's Instant Messenger program, AIM, installs a URL handler called AIM://. It has been used to load buffer overflows known to be successful with particular programs.<br>The associated program need only be installed, not even used, to be launched.<br>HKCR\<urlhandler>\shell\open\command is the registry location for URL handlers. |
| Registry | HKCR\<fileext> NeverShowExt | Real file extensions can be hidden. | Although most users know that Windows allows registered file extensions to be hidden (the default), most users don't know about the "super hidden" extension attribute, which allows selected files (dozens of file types, including SHS, SHB, SHC, LNK, PIF, XNK, and several shortcut and CLSID files) to hide their extensions even if you told Windows not to hide file extensions. |

| Area | Name | Function | Notes |
|---|---|---|---|
| | | | The super hidden file attribute can be enabled by creating a NeverShowExt registry entry under HKCR\<fileext>. To disable, search for and delete any occurrence of the NeverShowExt key under HKCR. Note that Never-ShowExt also overrides Explorer's option to "Show file name extensions for registered file types." |
| Registry | HKCU\ Control Panel\ Desktop Scrnsave.exe= | Will load listed programs or commands when the screensaver is configured. | Not commonly used by malware. Used by Petch trojan (`http://securityresponse.symantec.com/avcenter/venc/data/w32.petch.b.html`). Screensaver is significant in that it is applied in Safe mode, even Safe Mode Command Prompt Only. This could allow malware to activate during long unattended scanning procedures, although this particular trick appears yet to be exploited by malware. |
| Registry | HKCU\ Software\ Microsoft\ Internet Explorer\Main\ Start Page<br><br>HKCU\ Software\ Microsoft\ Internet Explorer\Main\ Search Page<br><br>HKCU\ Software\ Microsoft\ Internet Explorer\Main\ Search Bar | Configures Internet Explorer's Startup page or search bars. | Commonly manipulated by adware and spyware |
| Registry | HKCU\ Software\ Microsoft\ Internet Explorer\ SearchURL | Redirects any URLs typed in Internet Explorer to the defined URL. | Commonly manipulated by adware and spyware |

*Table continued on following page*

| Area | Name | Function | Notes |
|---|---|---|---|
| Registry | HKCU or HKLM \ Software\ Internet Explorer\ Explorer Bars | Malicious adware\ spyware could create new menu bars in Internet Explorer. | Allows new entries to be made to standard menu bars.<br>Available in IE 4.x and later.<br>Commonly manipulated by adware and spyware.<br>Menu bar will be a CLSID subkey listed under Explorer Bars.<br>Used by Hotbar adware (`http://securityresponse.symantec.com/avcenter/venc/data/adware.hotbar.html`) |
| Registry | HKLM\ Software\ Classes\CLSID\ {CLSID}\ Implemented Categories\ {00021493-0000-0000-C000-000000000046} | ...93 defines a vertical Explorer bar | Commonly manipulated by adware and spyware |
| | HKLM\ Software\ Classes\CLSID\ {CLSID}\ Implemented Categories\ {00021494-0000-0000-C000-000000000046} | ...94 defines a horizontal Explorer bar | |
| Registry | HKCU\ or HKLM\ Software\ Internet Explorer\ Extensions | Adware/spyware can add buttons to IE that connect directly to malicious programs and scripts. | Available in IE 5.x and later.<br>http://msdn.microsoft.com/library/default.asp?url=/workshop/browser/ext/overview/overview.asp<br>Commonly manipulated by adware and spyware, including Adblock. |
| Registry | HKCU\ Software\ Microsoft\OLE | Used to register Windows OLE programs | Available with Win 3.x and later.<br>Not a common malware vector.<br>Used by Bropia trojan (`www.sarc.com/avcenter/venc/data/w32.bropia.j.html`). |

| Area | Name | Function | Notes |
|---|---|---|---|
| Registry | HKCU\ Software\ Microsoft\ Windows NT\ CurrentVersion\ Windows\load | Runs commands or programs after the user logs on | Works with all versions of Windows NT and later.<br>Replaces Win 9x's Win.ini Load= functionality.<br>Executes programs in minimized state. |
| Registry | HKCU\ Software\ Microsoft\ Windows NT\ CurrentVersion\ Windows\run | Runs commands or programs after the user logs on | Works with all versions of Windows NT and later.<br>Replaces Win 9x's Win.ini Run= functionality.<br>Executes programs in maximized state. |
| Registry | HKCU or HKLM\ Software\ Microsoft\ Windows NT\ CurrentVersion\ Winlogon\Shell<br><br>HKCU or HKLM\ Software\ Microsoft\ Windows\ CurrentVersion\ Policies\System\ Shell | Runs commands or programs after the user logs on | Works with all versions of Windows NT and later.<br>Replaces Win 9x's System.ini Shell= functionality.<br>Should only have 'Explorer.exe" as a data value, if any value is displayed.<br>Should not include a directory path.<br>Some malware points to a bogus Explorer.exe (not located in %Windir%.<br>Should not have additional programs before or after Explorer.exe unless a program is known to be legitimate. |
| Registry | HKLM\ Software\ Microsoft\ Windows NT\ CurrentVersion\ Winlogon\ System | Runs programs after the user logs on | Key is present by default, but assigned no value. |
| Registry | HKLM\ Software\ Microsoft\ Windows NT\ CurrentVersion\ Winlogon\ Taskman | Runs programs in Task Manager after the user logs on | Key not present by default |

*Table continued on following page*

| Area | Name | Function | Notes |
|---|---|---|---|
| Registry | HKCU or HKLM\ Software\ Microsoft\ Windows\ CurrentVersion\ Policies\ Explorer\Run | Runs programs after the user logs on, when the Windows default shell (explorer.exe) runs for the first time during every logon | Works with W2K and later.<br>Not unusual to find legitimate programs, such as Microsoft's ctfmon.exe, listed here.<br>Does not require reboot.<br>Does not execute commands if explorer.exe is executed manually.<br>W2K will run any subkey with any program listed under this key.<br>Discovered by Andrew Aronoff of SilentRunners.org. |
| Registry | HKCU\ Software\ Microsoft\ Windows\ CurrentVersion\ Policies\System\ Shell | Runs programs or commands after the user logs on, but before the desktop is displayed | Works with W2K and later.<br>Shell subkey may not exist by default.<br>Does not require reboot after modification.<br>If malware creates the Shell key, and does not launch the Windows shell too, the desktop will not be visible.<br>You can still use Task Manager to run commands, including regedit.exe.<br>A similar System key exists under HKLM\; but the Shell subkey does not get executed. |
| Registry | HKCU or HKLM\ Software\ Microsoft\ Windows\ CurrentVersion\ Run | Runs programs or commands after the user logs on | Works with all versions of Windows 9x and later.<br>Not run in Safe mode unless the value is prefixed by an * (asterisk).<br>Often contains many legitimate programs.<br>*The most popular registry auto-run key for malware, by a huge percentage*.<br>W2K will run any subkey with any program listed under this key.<br>Discovered by Andrew Aronoff of SilentRunners.org.<br>Non-admin users cannot modify HKLM version.<br>Run key also appears in the HK_U\.Default registry profile area, but does not copy over to new profiles.<br>Cannot be disabled by holding down the Shift or Alt keys as sometimes reported. |

| Area | Name | Function | Notes |
|---|---|---|---|
| Registry | HKCU or HKLM\ Software\ Microsoft\ Windows\ CurrentVersion\ RunOnce | Runs programs or commands after the user logs on for the first time only after the key is created. | Works with all versions of Windows 9x and later.<br>HKLM\RunOnce runs entries *synchronously* (in undefined order)—there is a defined order and all other keys and processing must wait for this key to process and clear before they can load. All other Run keys run entries asynchronously, which means they can load on top of each other.<br>HKCU version will run once for any user given the key.<br>HKLM version will only run the value for users with admin permissions to key. Regular users will not run the value, although they can read it.<br>RunOnce key also appears in the HK_U\.Default registry profile area, but does not copy over to new profiles.<br>Non-admin users cannot modify HKLM version.<br>Not run if in Safe mode in W2K and later unless the value name begins with an asterisk.<br>If an exclamation point begins the key value, then the key will not be deleted until successful completion of the program or command.<br>Holding down the Shift key does not prevent execution.<br>W2K will run any subkey with any program listed under this key. Discovered by Andrew Aronoff of SilentRunners.org. |
| Registry | HKLM\ Software\ Microsoft\ Windows\ CurrentVersion\ RunOnce\Setup | Runs programs or commands after Setup's first-boot activities or can be launched by the Add/Remove wizard when the user logs on for the first time. (Can be stored as part of the Default Users profile.) | Works in all versions of Windows.<br>Not run if in Safe mode.<br>Holding down the Shift key does not prevent execution.<br>If an exclamation point begins the key value, then the key will not be deleted until successful completion of the program or command. |

*Table continued on following page*

| Area | Name | Function | Notes |
|---|---|---|---|
| Registry | HKCU or HKLM \ Software\ Microsoft\ Windows\ CurrentVersion\ ShellService ObjectDelayLoad | Runs commands or programs after the user logs on, although typically points to the CLSID of the associated .DLL file. Links programs to explorer.exe process. | Legitimate programs often located here, including Microsoft's webcheck.exe and systray.exe.<br>HKCU is more popular than HKLM.<br>Data value is CLSID of associated program as registered in HKCR\.<br>Download.Ject trojan, Spyware Eblaster (`http://securityresponse.symantec.com/avcenter/venc/data/spyware.eblaster.html`) and the Webber trojan (`www.sophos.com/virusinfo/analyses/trojwebbera.html`) use this key. |
| Registry | HKCU or HKLM\ Software\ Policies\ Microsoft\ Windows\ System\Scripts | Runs scripts on computer startup/shutdown or user logon/logoff | Works with Windows 2000 and later.<br>Scripts may be passed down by group policies and located in different registry keys.<br>Not a common location for malware. |
| Registry | HKLM\ Software\ Classes\ <filettype>\ shell\open\ command<br><br>HKCR\ <filettype>\ shell\open\ command<br><br>Examples: HKLM\Software\ Classes\batfile\ shell\open\ command HKLM\ Software\Classes\ comfile\shell\ open\command HKLM\Software\ Classes\exefile\ shell\open\ command HKLM\ Software\Classes\ htafile\shell\open\ command HKLM\ Software\Classes\ piffile\shell\open\ command | Can be modified to run additional commands or programs when a particular file type is executed | Works on Windows 9x and later.<br>HKLM\Software\Classes\<filetype> and HKCR\<filetype> are aliases of each other. If you change the value in one, you change it in the other.<br>Most common malware modifications listed, although any file type can be modified.<br>Most common modification is made to the exefile type. For example, Value should always be: "%1" %*<br>PrettyPark worm (`http://securityresponse.symantec.com/avcenter/venc/data/prettypark.worm.html`) changed value to: FILES32.VXD "%1" %*<br>Whenever an exe file was executed, it would execute the malicious Files32.vxd worm program, too.<br>If the entire data value is deleted instead of the original value being replaced, it causes execution problems with exe files. In XP, in that HKCR is no longer a simple alias for HKLM\Software\Classes, but an overlay of the per-user Classes over this. This allows per-account file associations to be effected, including that of the Administrator account. Exploits can be made at two levels: at the linkage between .ext and |

| Area | Name | Function | Notes |
|---|---|---|---|
| | HKLM\Software\Classes\ShellScrap\shell\open\command | | file type (e.g., directing .EXE away from its normal exefile association) or by altering the actions defined within the file type. Some file association contexts default to the action called "open," while others look to which action is named as "default".<br>More elaborate file association intrusions can be crafted via CLSIDs; in addition, other shell extensions can be added that will kick in as part of the namespace (left pane in Explorer), or as "persistent handlers" when the contents of folders are listed (right pane in Explorer). |
| Registry | HKCU or HKLM\Software\Microsoft\Active Setup\Installed Components\<program's name or CLSID><br><br>Loads programs on PC startup | Works with Windows 98 and later.<br>Look for Stubpath= value. | Contains many/mostly legitimate programs.<br>Common method used by malware; for example, Prorat trojan (`www.sophos.com/virusinfo/analyses/trojproratd.html`).<br>HKCU doesn't usually launch anything. The HKLM Version value is compared at launch to the Version value under HKCU. If the HKLM value is greater, the executable is launched and the HKCU Version value is updated. At next boot, the executable doesn't launch again unless the HKCU Version value is deleted or the HKLM value is incremented. (Thanks to Andrew Aronoff of SilentRunners.org)<br>Difficult to discern what is legitimate vs. malicious in this key. |
| Registry | HKCU or HKLM\Software\Microsoft\Command Processor\Autorun | Runs program or command when:<br>Cmd.exe is executed, Windows is started in Safe mode with Command Prompt, or when a batch file (.bat) or command (.cmd) is executed. | Works with NT and later.<br>Replaces previous functionality of Dosstart.bat.<br>Does not run when Command.com is executed.<br>Can be disabled when running cmd.exe manually by typing in **cmd.exe /d**.<br>Modification of this key does not require a reboot to be effective. |

*Table continued on following page*

| Area | Name | Function | Notes |
|---|---|---|---|
| Registry | HKLM\Software\Microsoft\Internet Explorer\Search<br><br>HKLM\Software\Microsoft\Internet Explorer\UrlSearchHooks | Determines how Internet Explorer searches for unknown entries | Works with Internet Explorer 5.x and later.<br>Both keys contain legitimate values, but often commandeered by spyware and adware.<br>Search subkey contains references to `http://ie.search.msn.com` by default. |
| Registry | HKLM\Software\Microsoft\Internet Explorer\Styles | Lists Internet Explorer style sheets | Can be created or manipulated by adware/malware to display malicious web sites or pop-ups. |
| Registry | HKLM\Software\Microsoft\Internet Explorer\Toolbar | Loads new menu bars for Internet Explorer or modifies existing toolbars | Works with all versions of Internet Explorer 5.x and later.<br>Commonly exploited by adware. |
| Registry | HKCU\Software\Internet Explorer\Toolbar\ShellBrowser<br><br>HKCU\Software\Internet Explorer\Toolbar\WebBrowser | Malicious adware\spyware could create new menu bars in Internet Explorer. | Commonly manipulated by adware and spyware.<br>Menu bar will be a CLSID subkey listed under Toolbars. |
| Registry | HKLM\Software\Microsoft\Windows NT\CurrentVersion\Windows\AppInit_DLLs | All the DLLs that are specified in this value are loaded by each Microsoft Windows-based application that is running in the current logon session using the User32.dll API library (which is used by most programs). | Works with Windows NT and later.<br>Not usually populated by legitimate programs, but can be.<br>Common method used by malware and adware; for example, CoolWeb Search Adware (`http://securityresponse.symantec.com/avcenter/venc/data/adware.cwsmsconfd.b.html`). |

| Area | Name | Function | Notes |
|---|---|---|---|
| Registry | HKLM\ Software\ Microsoft\ Windows NT\ CurrentVersion\ Winlogon\ GinaDLL | Loads Windows logon user interface; loaded interface passes interactive user's logon credentials to Winlogon.exe | Works with Windows NT and later. Microsoft's default data value is Msgina.dll. Has been a target of trojan attacks, attempting to capture end user logon credentials. PC Anywhere program will modify the value to be Awgina.dll. The Novell logon client will modify as well. |
| Registry | HKLM\ Software\ Microsoft\ Windows NT\ CurrentVersion\ Winlogon\ Notify | Used to run a particular program when a predefined event (e.g., Screensaver stops or starts, user logs on or off) occurs. | Works with NT and later. Many legitimate programs are stored here. Not a common malware location, but is used. For example, Haxor backdoor trojan rootkit (`http://security response.symantec.com/ avcenter/venc/data/backdoor. haxdoor.b.html`). |
| Registry | HKLM\ Software\ Microsoft\ Windows NT\ CurrentVersion\ Winlogon\ Userinit | Specifies the programs that Winlogon runs when a user logs on | By default, Winlogon runs %Windir\ System32\Userinit.exe, which runs logon scripts, reestablishes network connections, and then starts Explorer .exe, the Windows user interface. Not a common malware startup location; has been exploited in the wild. For example, Petch trojan (`http:// securityresponse.symantec.com/ avcenter/venc/data/w32.petch.b. html`). |
| Registry | HKLM\ Software\ Microsoft\ Windows\ CurrentVersion\ Explorer\ Browser Helper Objects | Programs are loaded when Internet Explorer loads; programs loaded also known as Add-Ons. | Works with an OS that can run Internet Explorer 5.x and above. Commonly exploited key Several programs help list and/or modify BHOs, including IE XP SP2 and above. Note that disabling "third-party browser enhancements" in IE6's Tools, Options, Advanced will not suppress these intrusions into Outlook Express if the BHOs also defined themselves there as well. |
| Registry | HKLM\ Software\ Microsoft\ Windows\ CurrentVersion\ Explorer\ SharedTask Scheduler | Task scheduler programs that are launched when Windows starts | Works with W2K and later. Not a common malware location, but is used. For example, Bookmarker trojan (`http://securityresponse .symantec.com/avcenter/venc/ data/trojan.bookmarker.c.html`) |

*Table continued on following page*

| Area | Name | Function | Notes |
|---|---|---|---|
| Registry | HKLM\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders<br><br>HKLM\Software\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders\Startup<br>\Common Startup | Determines location of Startup folders (i.e., Startup programs) and other common folders (ex. My Documents, My Favorites) for All Users profile | Works with Windows 9x and later. Used by malware to change Startup folder behavior. Malware can place itself in the newly be executed when the user logs on, but if the user checks the normal Startup folders, the malicious program will not be listed. Malware modifying these keys will often then execute programs and commands found in default Startup folders so the user is not suspicious. |
| Registry | HKLM\Software\Microsoft\Windows\CurrentVersion\Explorer\ShellExecuteHooks | Contains the list of the COM objects, listed by GUID, that trap execute commands | Must contain the %Windir%\System32\Shell32.dll API program. Other listed programs must be deemed suspicious. |
| Registry | HKLM\Software\Microsoft\Windows\CurrentVersion\RunOnceEx | Runs programs or commands after the user logs on, in a controlled order. Runs listed value each time any user logs on until a user with admin permissions to the registry key logs on, then it deletes the value after running. | Works with all versions of Windows 9x and later.<br>Not run in Safe mode unless the value is prefixed by an * (asterisk).<br>Only runs values under subkeys (does not run values placed directly under key)<br>Non-admin users cannot normally modify.<br>For more information, see `http://support.microsoft.com/?kbid=232509&sd=RMVP`. |
| Registry | HKLM\Software\Microsoft\Windows\CurrentVersion\RunServices | Runs service after boot up prior to the user logging on. | Works only in the Win 9x family. There is also a HKCU version of the same key, but it doesn't appear to be used or able to launch anything. |

| Area | Name | Function | Notes |
|---|---|---|---|
| Registry | HKLM\ Software\ Microsoft\ Windows\ CurrentVersion\ RunServicesOnce | Runs service once after boot up prior to the user logging on, and then deletes itself. | Works only in the Win 9x family.<br>If the value is preceded by an exclamation point, deletion will not occur unless the command is successfully completed.<br>There is also a HKCU version of the same key, but it doesn't appear to be used or able to launch anything. |
| Registry | HKLM\ Software\ Microsoft\ Windows\ CurrentVersion\ Shell Extensions\ Approved | Lists programs that will run with associated file types | Works with Windows 9x and later.<br>Usually contains dozens of legitimate programs.<br>Most programs listed will be located in %Windir%\System32 or C:\ Program Files.<br>Difficult to tell what is and isn't malicious. |
| Registry | HKLM\ System\ CurrentControl Set\Control\ MPRServices | Can be used to launch programs during predefined events | Used by the Win 9x family.<br>Similar to the HKLM\Software\ Microsoft\Windows NT\Current Version\Winlogon\Notify registry key used by NT and later systems.<br>Used by Haxdoor.B backdoor trojan (`http://securityresponse.symantec.com/avcenter/venc/data/backdoor.haxdoor.b.html`). |
| Registry | HKLM\ System\ CurrentControl Set\Control\ SafeBoot | Used by Windows to determine what programs, services, and drivers are loaded in a Safe mode boot | Although not common, can be manipulated by malware to either prevent Safe mode from being run (i.e., values are deleted) or to add malware program to a Safe mode boot sequence.<br>Used by Petch trojan (`http://securityresponse.symantec.com/avcenter/venc/data/w32.petch.b.html`) to delete all Safe mode listings. |
| Registry | HKLM\ Software\ Microsoft\ Windows NT\ CurrentVersion\ Image File Execution Options | Allows another program (or debugger) to be executed instead when another program is started | Key lists all the programs that have been defined to have alternative programs start instead.<br>Normal to have dozens of legitimate entries here.<br>Used by a few malware programs, including the Zellome worm and StartPage.O trojan.<br>Thanks to Andrew Aronoff of Silient-Runners.org for the hint. |

*Table continued on following page*

| Area | Name | Function | Notes |
|---|---|---|---|
| Registry | HKLM\System\CurrentControl Set\Control\Session Manager\BootExecute | Programs or commands will be executed upon next reboot | Works with NT and later.<br>Replaces some of the functionality of Wininit.ini of earlier Windows versions. |
| Registry | HKLM\System\CurrentControl Set\Control\Session Manager\Environment\Path | Determines what directories to check for commands or programs typed in without a specific PATH statement (i.e., Frog.exe vs. C:\Program Files\Frog.exe) | Some malware programs rely on defects in the way Windows searches for and executes files when only the file name (ex. Frog.exe vs. C:\Program Files\Frog.exe) is given.<br><br>The PATH statement can be set by the DOS PATH command (located under Environmental variables in NT family) or by the registry key.<br>Should contain by default: %SystemRoot%\system32;%SystemRoot%;%SystemRoot%\System32\Wbem;<br>Can contain other legitimate non-default entries (ex. C:\Program Files\Network Associates;)<br>Not commonly used by malware, but can still be a problem today. For example, some malware places itself in default application directories, which the application executes instead of the legitimate program executable.<br>One trojan placed its malicious code in the user's My Documents folder. Because the malware was named after a legitimate MS-Word executable, MS-Word would always load it first instead of the legitimate version located under Program Files. |
| Registry | HKLM\System\CurrentControl Set\Control\Session Manager\Environment\PathExt | Determines what file extensions are tried if the program name is typed in without an extension (e.g., Frog vs. Frog.exe) | Some malware programs (ex. Spawner or twin viruses) rely on defects in the way Windows executes files when only the file name (ex. Frog.exe vs. C:\Program Files\Frog.exe) is given.<br>Not commonly used by malware today.<br>Should be the following by default: .COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH |

| Area | Name | Function | Notes |
|---|---|---|---|
| Registry | HKLM\System\CurrentControlSet\Control\Session Manager\FileRenameOperations | Contains pending file operations (e.g., rename, copy, etc.) to be executed on the next reboot of Windows | Works with NT and later.<br>Replaced the older Wininit.ini file. |
| Registry | HKLM\System\CurrentControlSet\Control\Session Manager\StartPage | Configures Internet Explorer's Startup page | Commonly manipulated by adware and spyware |
| Registry | HKLM\System\CurrentControlSet\Enum\Root | Used to registry legacy Windows services | Not normally used by legitimate programs today.<br>Not commonly used by malware.<br>Used by Wallz worm (`http://securityresponse.symantec.com/avcenter/venc/data/w32.wallz.html`). |
| Registry | HKLM\System\CurrentControlSet\Services | Will load program as service (i.e., prior to user being logged in) | Works with NT and later.<br>Common malware vector.<br>Difficult to determine what is and isn't malicious using this key alone. |
| Registry | HKCR\Protocols\Filters or HKLM\Software\Classes\Protocols\Filters | Malware program can load itself when a MIME file attachment (ex. Text/xml) is executed | For example, can be used so malicious program is loaded each time a text file is viewed in IE instead of Notepad.<br>Frequently used by spyware and adware.<br>Programs listed by CLSID below keys.<br>Used by StartPage.I trojan.<br>Both keys are just aliases for each other.<br>Thanks to Andrew Aronoff of Silent Runners.org for this hint. |
| Registry | HKLM\System\CurrentControlSet\Control\Class\{4D36E96B-E325-11CE-BFC1-08002BE10318}\UpperFilters | Malware program can modify I/O from input devices | Used by some keylogging trojans (ex.InvisibleKey Spyware) to capture data from the keyboard driver.<br>By default, several of the same keys will exist.<br>Do not delete or manipulate this value, because it often contains legitimate information, without backing up registry first.<br>Thanks to Andrew Aronoff of Silent Runners.org for this hint. |

*Table continued on following page*

| Area | Name | Function | Notes |
|---|---|---|---|
| Registry | HKLM\System\CurrentControlSet\Services\Winsock2\Parameters\NameSpace_Catalog5\Catalog_Entries<br><br>HKLM\System\CurrentControlSet\Services\Winsock2\Parameters\Protocol_Catalog9\Catalog_Entries | Allows trojan or worm to install itself as a Layered Service Provider so that it can monitor network traffic | Used by many trojans, spyware, and adware programs.<br>Many legitimate keys are located here. Can be difficult to find unauthorized programs.<br>Commercial Guardian Monitor spyware program and Redfall trojan uses this method.<br>Thanks to Andrew Aronoff of SilentRunners.org for this hint. |
| Registry | HKLM\Software\Microsoft\Office\Outlook\Addins | Malware can add itself as an Outlook Add-in and manipulate incoming or outgoing e-mail | May contain legitimate entries, such as anti-spam or antivirus software plug-ins.<br>A common malicious example is Hotbar adware. |
| Registry | HKCU\Identities\<Identity>\Software\Microsoft\Outlook Express\<version>\Signatures | Malware can add a malicious script to Outlook Express e-mail signatures that retrieves malware automatically when opened by the recipient. | Documented in Outlook Express, but may be able to be exploited in Outlook and other e-mail clients as well.<br>Used by the Kak and JS.Fortnight worms. |
| Registry | HKCU\Software\Microsoft\Internet Explorer\Desktop\Components\<#>\Flags<br>\Source<br>\SubscribedURL | Can be hijacked by adwareto redirect IE to unauthorized locations and malware | Source and Subscribed values are set to About:Home by default.<br>Hint provided by Andrew Aronoff of SilentRunners.org. |
| Registry | HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\ShellState | The registry value controls many aspects of the desktop environment. | Including whether Active Desktop is enabled, and whether file extensions are visible.<br>Not very commonly manipulated by malware presently.<br>Hint provided by Andrew Aronoff of SilentRunners.org. |

| Area | Name | Function | Notes |
|---|---|---|---|
| Registry | HKCU\Software\Microsoft\Windows\CurrentVersion\Policies\Active Desktop | Controls Active Desktop settings | Active Desktop, if enabled, opens up more potential attack vectors.<br>Note that selecting particular types of display media (e.g., a .JPG as wallpaper) will enable Active Desktop in some versions of Windows, whereas disabling Active Desktop while a .JPG is set as wallpaper will cause an "Are you sure?" prompt that many users will back out of in order to use their "nice" wallpaper. Not present by default on most systems.<br>Not very commonly manipulated by malware presently.<br>Hint provided by Andrew Aronoff of SilentRunners.org. |
| Registry | HKCU\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer | Controls Windows Explorer settings | Not very commonly manipulated by malware presently.<br>Hint provided by Andrew Aronoff of SilentRunners.org. |
| Registry | HKCU\Software\Microsoft\Windows\CurrentVersion\Policies\System | Allows control of desktop system and some administrative tools | Often used to disable Task Manager (DisableTaskMgr=0x1), Registry Editor (DisableRegistryTools = 0x1), and Control Panel (NoDispCPL= 0x1).<br>Key not present by default on most systems.<br>Commonly manipulated by malware. Examples include HackerWacker keystroke logger spyware, Ronoper worm, and Ting adware.<br>Hint provided by Andrew Aronoff of SilentRunners.org. |
| Registry | HKLM\Software\Microsoft\Windows\CurrentVersion\URL\Default Prefix | Adds any string value as a prefix for any URL typed in the browser, effectively redirecting all typed-in URLs to the unauthorized web site first | Commonly used by Adware. Examples include SmartSearch and WorldSearch adware, the JS.Fornight adware worm, and Popdis Trojan.<br>Default values are supposed to be *http://*.<br>Hint provided by Andrew Aronoff of SilentRunners.org. |

*Table continued on following page*

| Area | Name | Function | Notes |
|---|---|---|---|
| | HKLM\<br>Software\<br>Microsoft\<br>Windows\<br>CurrentVersion\<br>URL\Prefixes\<br>Search<br><br>HKLM\<br>Software\<br>Microsoft\<br>Windows\<br>CurrentVersion\<br>URL\Prefixes\<br>Search | | |
| Registry | HKLM\System\<br>CurrentControl<br>Set\Services\<br>Tcpip\<br>Parameters\<br>DataBasePath | Can be used to point to a new, unauthorized HOSTS file instead of the HOSTS file in the normal location (i.e., \%SystemRoot%\<br>Drivers\Etc) | Used by trojans (ex. Qhosts) and adware (ex. TMKSoft.XPlugin).<br>Value is also added to ControlSet001 and ControlSet002 by some trojans (ex. Qhosts).<br>Hint provided by Andrew Aronoff of SilentRunners.org. |
| Registry | HKLM\System\<br>CurrentControl<br>Set\Services\<br>Tcpip\<br>Parameters\<br>NameServer | Can be used to point to a new, unauthorized DNS server | Used by a few malware programs, including Qhosts trojan. |
| Registry | HKLM\System\<br>CurrentControl<br>Set\Services\<br>Tcpip\<br>Parameters\ | Sets overall TCP/IP communications values, including DHCP, DNS, and TCP/IP stack. These values are used unless a specific value is set under the \Interfaces subkeys on a particular interface. | Many subvalues on this key could be changed to cause problems — for example, to set a new default gateway, to change normal DNS resolution order, etc.<br>Many legitimate settings are present by default.<br>Many values can be modified to strengthen a Windows computer against denial-of-service attacks. |
| Registry | HKLM\System\<br>CurrentControl<br>Set\Services\<br>Tcpip\<br>Parameters\<br>Interfaces\<br><interface CLSID> | Controls all TCP/IP communications, including DHCP, DNS, and TCP/IP stack. | Many subvalues on this key could be changed to cause problems — for example, to set a new default gateway, to change normal DNS resolution order, etc.<br>Many legitimate settings are present by default. |

| Area | Name | Function | Notes |
|---|---|---|---|
| | | | Used by Qhosts and Flush.D trojans. Look at CurrentControlSet001 and 002, as some trojans modify those values to (ex. Qhosts). |
| Registry | HKLM\System\CurrentControlSet\Services\VxD\MSTCP\NameServer | Can be used to force a client to use an unauthorized DNS server | Key not present by default. Used by Qhosts and Flush.D trojans. |

New exploits methods are added every month. Go to `www.wrox.com` to get an updated list.

If you felt this comprehensive list was overwhelming, you can find some comfort in the fact that almost all automated malware hides in the `HKLM\Software\Microsoft\Windows\CurrentVersion\Run registry` key. If you suspect a malware program, go there first, but Table 1-1 will help you locate malware when that particular registry key does not reveal the rogue program.

# Summary

Most malicious attacks can be classified into four categories: Automated, Dedicated Attacker, Remote, or Local Execution. A key point of this book and the success of your network defense depend on you understanding that the most common threats come from automated malware, where security-by-obscurity has value as part of a computer defense plan. Automated malware includes viruses, worms, trojans, and hybrid, blended programs. Remote attacks include buffer overflows, denial-of-service, obscurity, and sniffing attacks. Other attack types, such as social engineering, spam, and insider attacks also deserve consideration. There are more than 100,000 different malware programs and they can hide in more than a hundred different places in Windows. Chapter 1 ended with a comprehensive listing of where malware can hide. The details it provides will lead to the defenses covered in the forthcoming chapters.

Now that we understand what threats we are up against, we can begin to concentrate on the defenses. Chapter 2, "Conventional and Unconventional Defenses," summarizes the overall steps of a successful computer security defense plan. Conventional defenses, such as patch management and antivirus protection, will be discussed along with unconventional but efficient defenses not covered elsewhere.