Chapter 2

DEFINITIONS AND TIMELINE

It would be nice to present a clever taxonomy of malicious software, one that clearly shows how each type of malware relates to every other type. However, a taxonomy would give the quaint and totally incorrect impression that there is a scientific basis for the classification of malware.

In fact, there is no universally-accepted definition of terms like "virus" and "worm," much less an agreed-upon taxonomy, even though there have been occasional attempts to impose mathematical formalisms onto malware.[100] Instead of trying to pin down these terms precisely, the common characteristics each type of malware typically has are listed.

2.1 Malware Types

Malware can be roughly broken down into types according to the malware's method of operation. Anti-"virus" software, despite its name, is able to detect all of these types of malware.

There are three characteristics associated with these malware types.

1 *Self-replicating* malware actively attempts to propagate by creating new copies, or *instances*, of itself. Malware may also be propagated passively, by a user copying it accidentally, for example, but this isn't self-replication.

2 The *population growth* of malware describes the overall change in the number of malware instances due to self-replication. Malware that doesn't self-replicate will always have a zero population growth, but malware with a zero population growth may self-replicate.

3 *Parasitic* malware requires some other executable code in order to exist. "Executable" in this context should be taken very broadly to include anything that can be executed, such as boot block code on a disk, binary code

in applications, and interpreted code. It also includes source code, like application scripting languages, and code that may require compilation before being executed.

2.1.1 Logic Bomb

Self-replicating:	no
Population growth:	zero
Parasitic:	possibly

A *logic bomb* is code which consists of two parts:

1 A *payload*, which is an action to perform. The payload can be anything, but has the connotation of having a malicious effect.

2 A *trigger*, a boolean condition that is evaluated and controls when the payload is executed. The exact trigger condition is limited only by the imagination, and could be based on local conditions like the date, the user logged in, or the operating system version. Triggers could also be designed to be set off remotely, or – like the "dead man's switch" on a train – be set off by the absence of an event.

Logic bombs can be inserted into existing code, or could be standalone. A simple parasitic example is shown below, with a payload that crashes the computer using a particular date as a trigger.

```
legitimate code
if date is Friday the 13th:
    crash_computer()
legitimate code
```

Logic bombs can be concise and unobtrusive, especially in millions of lines of source code, and the mere threat of a logic bomb could easily be used to extort money from a company. In one case, a disgruntled employee rigged a logic bomb on his employer's file server to trigger on a date after he was fired from his job, causing files to be deleted with no possibility of recovery. He was later sentenced to 41 months in prison.[101] Another case alleges that an employee installed a logic bomb on 1000 company computers, date-triggered to remove all the files on those machines; the person allegedly tried to profit from the downturn in the company's stock prices that occurred as a result of the damage.[1]

2.1.2 Trojan Horse

Self-replicating:	no
Population growth:	zero
Parasitic:	yes

There was no love lost between the Greeks and the Trojans. The Greeks had besieged the Trojans, holed up in the city of Troy, for ten years. They finally took the city by using a clever ploy: the Greeks built an enormous wooden horse, concealing soldiers inside, and tricked the Trojans into bringing the horse into Troy. When night fell, the soldiers exited the horse and much unpleasantness ensued.[102]

In computing, a *Trojan horse* is a program which purports to do some benign task, but secretly performs some additional malicious task. A classic example is a password-grabbing login program which prints authentic-looking "username" and "password" prompts, and waits for a user to type in the information. When this happens, the password grabber stashes the information away for its creator, then prints out an "invalid password" message before running the *real* login program. The unsuspecting user thinks they made a typing mistake and re-enters the information, none the wiser.

Trojan horses have been known about since at least 1972, when they were mentioned in a well-known report by Anderson, who credited the idea to D. J. Edwards.[103]

2.1.3 Back Door

Self-replicating:	no
Population growth:	zero
Parasitic:	possibly

A *back door* is any mechanism which bypasses a normal security check. Programmers sometimes create back doors for legitimate reasons, such as skipping a time-consuming authentication process when debugging a network server.

As with logic bombs, back doors can be placed into legitimate code or be standalone programs. The example back door below, shown in gray, circumvents a login authentication process.

```
username = read_username()
password = read_password()
if username is "133t h4ck0r":
    return ALLOW_LOGIN
if username and password are valid:
    return ALLOW_LOGIN
else:
    return DENY_LOGIN
```

One special kind of back door is a *RAT*, which stands for Remote Administration Tool or Remote Access Trojan, depending on who's asked. These programs allow a computer to be monitored and controlled remotely; users may deliberately install these to access a work computer from home, or to allow help desk staff to diagnose and fix a computer problem from afar. However, if malware surreptitiously installs a RAT on a computer, then it opens up a back door into that machine.

2.1.4 Virus

Self-replicating:	yes
Population growth:	positive
Parasitic:	yes

A *virus* is malware that, when executed, tries to replicate itself into other executable code; when it succeeds, the code is said to be *infected.*[2] The infected code, when run, can infect new code in turn. This self-replication into existing executable code is the key defining characteristic of a virus.

When faced with more than one virus to describe, a rather silly problem arises. There's no agreement on the plural form of "virus." The two leading contenders are "viruses" and "virii;" the latter form is often used by virus writers themselves, but it's rare to see this used in the security community, who prefer "viruses."[104]

If viruses sound like something straight out of science fiction, there's a reason for that. They are. The early history of viruses is admittedly fairly murky, but the first mention of a computer virus is in science fiction in the early 1970s, with Gregory Benford's *The Scarred Man* in 1970, and David Gerrold's *When Harlie Was One* in 1972.[105] Both stories also mention a program which acts to counter the virus, so this is the first mention of anti-virus software as well.

The earliest real academic research on viruses was done by Fred Cohen in 1983, with the "virus" name coined by Len Adleman.[106] Cohen is sometimes called the "father of computer viruses," but it turns out that there were viruses written prior to his work. Rich Skrenta's Elk Cloner was circulating in 1982, and Joe Dellinger's viruses were developed between 1981–1983; all of these were for the Apple II platform.[107] Some sources mention a 1980 glitch in Arpanet as the first virus, but this was just a case of legitimate code acting badly; the only thing being propagated was data in network packets.[108] Gregory Benford's viruses were not limited to his science fiction stories; he wrote and released non-malicious viruses in 1969 at what is now the Lawrence Livermore National Laboratory, as well as in the early Arpanet.

Some computer games have featured self-replicating programs attacking one another in a controlled environment. Core War appeared in 1984, where programs written in a simple assembly language called Redcode fought one another; a combatant was assumed to be destroyed if its program counter pointed to an invalid Redcode instruction. Programs in Core War existed only in a virtual machine, but this was not the case for an earlier game, Darwin. Darwin was played in 1961, where a program could hunt and destroy another combatant in a non-virtual environment using a well-defined interface.[109] In terms of strategy, successful combatants in these games were hard-to-find, innovative, and adaptive, qualities that can be used by computer viruses too.[3]

Traditionally, viruses can propagate within a single computer, or may travel from one computer to another using human-transported media, like a floppy disk, CD-ROM, DVD-ROM, or USB flash drive. In other words, viruses don't propagate via computer networks; networks are the domain of worms instead. However, the label "virus" has been applied to malware that would traditionally be considered a worm, and the term has been diluted in common usage to refer to any sort of self-replicating malware.

Viruses can be caught in various stages of self-replication. A *germ* is the original form of a virus, prior to any replication. A virus which fails to replicate is called an *intended*. This may occur as a result of bugs in the virus, or encountering an unexpected version of an operating system. A virus can be *dormant*, where it is present but not yet infecting anything – for example, a Windows virus can reside on a Unix-based file server and have no effect there, but can be exported to Windows machines.[4]

2.1.5 Worm

Self-replicating:	yes
Population growth:	positive
Parasitic:	no

A *worm* shares several characteristics with a virus. The most important characteristic is that worms are self-replicating too, but self-replication of a worm is distinct in two ways. First, worms are standalone,[5] and do not rely on other executable code. Second, worms spread from machine to machine across networks.

Like viruses, the first worms were fictional. The term "worm" was first used in 1975 by John Brunner in his science fiction novel *The Shockwave Rider*. (Interestingly, he used the term "virus" in the book too.)[6] Experiments with worms performing (non-malicious) distributed computations were done at Xerox PARC around 1980, but there were earlier examples. A worm called Creeper crawled around the Arpanet in the 1970s, pursued by another called Reaper which hunted and killed off Creepers.[7]

A watershed event for the Internet happened on November 2, 1988, when a worm incapacitated the fledgling Internet. This worm is now called the Internet worm, or the Morris worm after its creator, Robert Morris, Jr. At the time, Morris had just started a Ph.D. at Cornell University. He had been intending for his worm to propagate slowly and unobtrusively, but what happened was just the opposite. Morris was later convicted for his worm's unauthorized computer access and the costs incurred to clean up from it. He was fined, and sentenced to probation and community service.[8] Chapter 7 looks at this worm in detail.

2.1.6 Rabbit

Self-replicating:	yes
Population growth:	zero
Parasitic:	no

Rabbit is the term used to describe malware that multiplies rapidly. Rabbits may also be called *bacteria*, for largely the same reason.

There are actually two kinds of rabbit.[110] The first is a program which tries to consume all of some system resource, like disk space. A "fork bomb," a program which creates new processes in an infinite loop, is a classic example of this kind of rabbit. These tend to leave painfully obvious trails pointing to the perpetrator, and are not of particular interest.

The second kind of rabbit, which the characteristics above describe, is a special case of a worm. This kind of rabbit is a standalone program which replicates itself across a network from machine to machine, *but* deletes the original copy of itself after replication. In other words, there is only one copy of a given rabbit on a network; it just hops from one computer to another.[9] Rabbits are rarely seen in practice.

2.1.7 Spyware

Self-replicating:	no
Population growth:	zero
Parasitic:	no

Spyware is software which collects information from a computer and transmits it to someone else. Prior to its emergence in recent years as a threat, the term "spyware" was used in 1995 as part of a joke, and in a 1994 Usenet posting looking for "spy-ware" information.[111]

The exact information spyware gathers may vary, but can include anything which potentially has value:

1 Usernames and passwords. These might be harvested from files on the machine, or by recording what the user types using a *keylogger*. A keylogger differs from a Trojan horse in that a keylogger passively captures keystrokes only; no active deception is involved.

2 Email addresses, which would have value to a spammer.

3 Bank account and credit card numbers.

4 Software license keys, to facilitate software pirating.

Viruses and worms may collect similar information, but are not considered spyware, because spyware doesn't self-replicate.[112] Spyware may arrive on a machine in a variety of ways, such as bundled with other software that the user installs, or exploiting technical flaws in web browsers. The latter method causes the spyware to be installed simply by visiting a web page, and is sometimes called a *drive-by download.*

2.1.8 Adware

Self-replicating:	no
Population growth:	zero
Parasitic:	no

Adware has similarities to spyware in that both are gathering information about the user and their habits. Adware is more marketing-focused, and may pop up advertisements or redirect a user's web browser to certain web sites in the hopes of making a sale. Some adware will attempt to target the advertisement to fit the context of what the user is doing. For example, a search for "Calgary" may result in an unsolicited pop-up advertisement for "books about Calgary."

Adware may also gather and transmit information about users which can be used for marketing purposes. As with spyware, adware does not self-replicate.

2.1.9 Hybrids, Droppers, and Blended Threats

The exact type of malware encountered in practice is not necessarily easy to determine, even given these loose definitions of malware types. The nature of software makes it easy to create hybrid malware which has characteristics belonging to several different types.[10]

A classic hybrid example was presented by Ken Thompson in his ACM Turing award lecture.[11] He prepared a special C compiler executable which, besides compiling C code, had two additional features:

1 When compiling the login source code, his compiler would insert a back door to bypass password authentication.

2 When compiling the compiler's source code, it would produce a special compiler executable with these same two features.

His special compiler was thus a Trojan horse, which replicated like a virus, and created back doors. This also demonstrated the vulnerability of the compiler tool chain: since the original source code for the compiler and login programs wasn't changed, none of this nefarious activity was apparent.

Another hybrid example was a game called Animal, which played twenty questions with a user. John Walker modified it in 1975, so that it would copy the most up-to-date version of itself into all user-accessible directories whenever it was run. Eventually, Animals could be found roaming in every directory in the system.[113] The copying behavior was unknown to the game's user, so it would be considered a Trojan horse. The copying could also be seen as self-replication, and although it didn't infect other code, it didn't use a network either – not really a worm, not really a virus, but certainly exhibiting viral behavior.

There are other combinations of malware too. For example, a *dropper* is malware which leaves behind, or *drops*, other malware.[12] A worm can propagate itself, depositing a Trojan horse on all computers it compromises; a virus can leave a back door in its wake.

A *blended threat* is a virus that exploits a technical vulnerability to propagate itself, in addition to exhibiting "traditional" characteristics. This has considerable overlap with the definition of a worm, especially since many worms exploit technical vulnerabilities. These technical vulnerabilities have historically required precautions and defenses distinct from those that anti-virus vendors provided, and this rift may account for the duplication in terms.[114] The Internet worm was a blended threat, according to this definition.

2.1.10 Zombies

Computers that have been compromised can be used by an attacker for a variety of tasks, unbeknownst to the legitimate owner; computers used in this way are called *zombies*. The most common tasks for zombies are sending spam and participating in coordinated, large-scale denial-of-service attacks.

Sending spam violates the acceptable use policy of many Internet service providers, not to mention violating laws in some jurisdictions. Sites known to send spam are also blacklisted, marking sites that engage in spam-related activity so that incoming email from them can be summarily rejected. It is therefore ill-advised for spammers to send spam directly, in such a way that it can be traced back to them and their machines. Zombies provide a windfall for spammers, because they are a free, throwaway resource: spam can be relayed through zombies, which obscures the spammer's trail, and a blacklisted zombie machine presents no hardship to the spammer.[13]

As for denials of service, one type of denial-of-service attack involves either flooding a victim's network with traffic, or overwhelming a legitimate service on the victim's network with requests. Launching this kind of attack from a single machine would be pointless, since one machine's onslaught is unlikely to generate enough traffic to take out a large target site, and traffic from one machine can be easily blocked by the intended victim. On the other hand, a large number of zombies all targeting a site at the same time can cause grief. A coordinated, network-based denial-of-service attack that is mounted from a large number of machines is called a *distributed denial-of-service* attack, or *DDoS* attack.

Networks of zombies need not be amassed by the person that uses them; the use of zombie networks can be bought for a price.[14] Another issue is how to control zombie networks. One method involves zombies listening for commands on Internet Relay Chat (IRC) channels, which provides a relatively anonymous, scalable means of control. When this is used, the zombie networks are referred to as *botnets*, named after automated IRC client programs called *bots*.[15]

2.2 Naming

When a new piece of malware is spreading, the top priority of anti-virus companies is to provide an effective defense, quickly. Coming up with a catchy name for the malware is a secondary concern.

Typically the primary, human-readable name of a piece of malware is decided by the anti-virus researcher[16] who first analyzes the malware.[115] Names are often based on unique characteristics that malware has, either some feature of its code or some effect that it has. For example, a virus' name may be derived from some distinctive string that is found inside it, like "`Your PC is now Stoned!`"[17] Virus writers, knowing this, may leave such clues deliberately in the hopes that their creation is given a particular name. Anti-virus researchers, knowing *this*, will ignore obvious naming clues so as not to play into the virus writer's hand.[18]

There is no central naming authority for malware, and the result is that a piece of malware will often have several different names. Needless to say, this is confusing for users of anti-virus software, trying to reconcile names heard in alerts and media reports with the names used by their own anti-virus software. To compound the problem, some sites use anti-virus software from multiple different vendors, each of whom may have different names for the same piece of malware.[19] Common naming would benefit anti-virus researchers talking to one another too.[20]

Unfortunately, there isn't likely to be any central naming authority in the near future, for two reasons.[21] First, the current speed of malware propagation precludes checking with a central authority in a timely manner.[22] Second, it isn't always clear what would need to be checked, since one distinct piece of malware may manifest itself in a practically infinite number of ways.

Recommendations for malware naming do exist, but in practice are not usually followed,[23] and anti-virus vendors maintain their own separately-named databases of malware that they have detected. It would, in theory, be possible to manually map malware names between vendors using the information in these databases, but this would be a tedious and error-prone task.

A tool called VGrep automates this process of mapping names.[116] First, a machine is populated with the malware of interest. Then, as shown in Figure 2.1, each anti-virus product examines each file on the machine, and outputs what (if any) malware it detects. VGrep gathers all this anti-virus output and collates

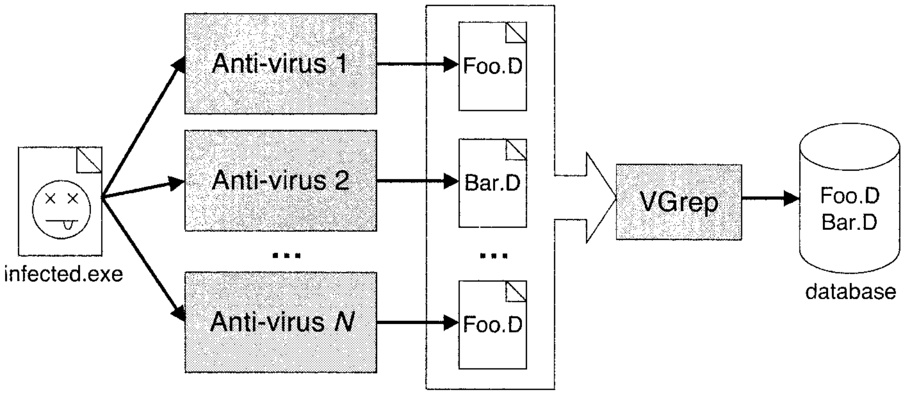

Figure 2.1. VGrep operation

it for later searching. The real technical challenge is not collating the data, but simply getting usable, consistent output from a wide range of anti-virus products.

The naming problem and the need for tools like VGrep can be demonstrated using an example. Using VGrep and cross-referencing vendor's virus databases, the partial list of names below for the same worm can be found.[24]

Bagle.C
Email-worm.Win32.Bagle.c
W32/Bagle.c@MM
W32.Beagle.C@mm
WORM_BAGLE.C
Worm.Bagle.A3

These results highlight some of the key identifiers used for naming malware:[117]

Malware type. This is the type of the threat which, for this example, is a worm.

Platform specifier. The environment in which the malware runs; this worm needs the Windows 32-bit operating system API ("W32" and "Win32").[25] More generally, the platform specifier could be any execution environment, such as an application's programming language (e.g., "VBS" for "Visual Basic Script"), or may even need to specify a combination of hardware and software platform.

Family name. The family name is the "human-readable" name of the malware that is usually chosen by the anti-virus researcher performing the analysis. This example shows several different, but obviously related, names. The relationship is not always obvious: "Nachi" and "Welchia" are the same worm, for instance.

Variant. Not unlike legitimate software, a piece of malware tends to be released multiple times with minor changes.[26] This change is referred to as the malware's *variant* or, following the biological analogy, the *strain* of the malware.

Variants are usually assigned letters in increasing order of discovery, so this "C" variant is the third B[e]agle found. Particularly persistent families with many variants will have multiple letters, as "Z" gives way to "AA." Unfortunately, this is not unusual – some malware has dozens of variants.[27]

Modifiers. Modifiers supply additional information about the malware, such as its primary means of propagation. For example, "mm" stands for "mass mailing."

The results also highlight the fact that not all vendors supply all these identifiers for every piece of malware, that there is no common agreement on the specific identifiers used, and that there is no common syntax used for names.

Besides VGrep, there are online services where a suspect file can be uploaded and examined by multiple anti-virus products. Output from a service like this also illustrates the variety in malware naming:[28]

```
Worm/Mydoom.BC         Win32:Mytob-D      I-Worm/Mydoom
Win32.Worm.Mytob.C     Worm.Mytob.C       Win32.HLLM.MyDoom.22
W32/Mytob.D@mm         W32/Mytob.C-mm     Net-Worm.Win32.Mytob.c
Win32/Mytob.D          Mytob.D
```

Ultimately, however, the biggest concern is that the malware is detected and eliminated, not what it's called.

2.3 Authorship

People whose computers are affected by malware typically have a variety of colorful terms to describe the person who created the malware. This book will use the comparatively bland terms *malware author* and *malware writer* to describe people who create malware; when appropriate, more specific terms like *virus writer* may be used too.

There's a distinction to be made between the malware author and the malware distributor. Writing malware doesn't imply distributing malware, and vice versa, and there have been cases where the two roles are known to have been played by different people.[29] Having said that, the malware author and distributor will be assumed to be the same person throughout this book, for simplicity.

Is a malware author a "hacker?" Yes and no. The term *hacker* has been distorted by the media and popular usage to refer to a person who breaks into computers, especially when some kind of malicious intent is involved. Strictly speaking, a person who breaks into computers is a *cracker*, not a hacker,[118] and there may be a variety of motivations for doing so. In geek parlance, being called a hacker actually has a positive connotation, and means a person who is skilled at computer programming; hacking has nothing to do with computer intrusion or malware.

Hacking (in the popular sense of the word) also implies a manual component, whereas the study of malware is the study of large-scale, automated forms of attack. Because of this distinction and the general confusion over the term, this book will not use it in relation to malware.

2.4 Timeline

Figure 2.2 puts some important events in context. With the exception of adware and spyware, which appeared in the late 1990s, all of the different types of malware were known about in the early 1970s. The prevalence of virus, worms, and other malware has been gradually building steam since the mid-1980s, leaving us with lots of threats – no matter how they're counted.

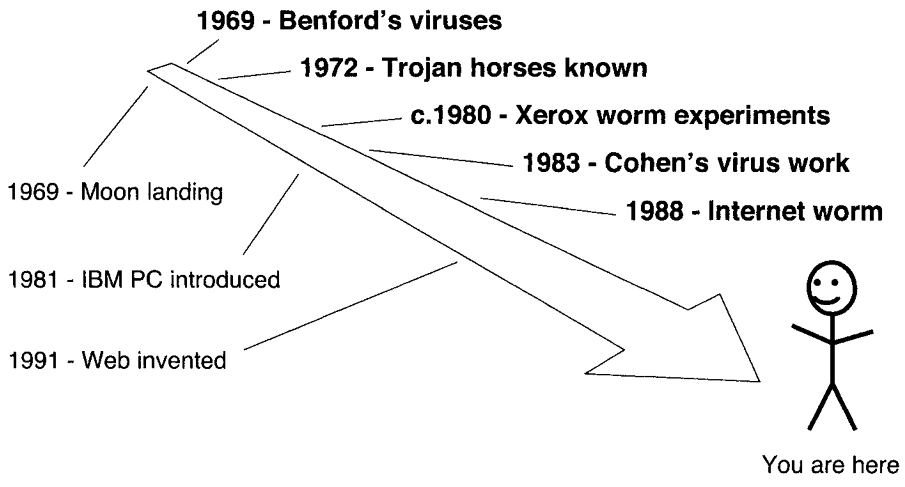

Figure 2.2. Timeline of events

Notes for Chapter 2

1 This case doesn't appear to have gone to trial yet, so the person may yet be found not guilty. Regardless, the charges in the indictment [327] serve as an example of how a logic bomb can be used maliciously.

2 The term "computer virus" is preferable if there's any possibility of confusion with biological viruses.

3 Bassham and Polk [28] note that innovation is important for the longevity of computer viruses, especially if the result is something that hasn't yet been seen by anti-virus software. They also point out that non-destructive viruses have an increased chance of survival, by not drawing attention to themselves.

4 These three definitions are based on Harley et al. [137]; Radatti [258] talks about viruses passing through unaffected platforms, which he calls 'Typhoid Mary Syndrome.'

5 Insofar as a worm can be said to stand.

6 This farsighted book also included ideas about an internet and laser printers [50].

7 The Xerox work is described in Shoch and Hupp [287], and both they and Dewdney [91] mention Creeper and Reaper. There were two versions of Creeper, of which the first would be better called a rabbit, the second a worm.

8 This version of the event is from [329]. An interesting historical twist: Morris, Jr.'s father was one of the people playing Darwin in the early 1960s at Bell Labs, and created 'The species which eventually wiped out all opposition. . . ' [9, page 95].

9 Nazario [229] calls this second kind of rabbit a "jumping executable worm."

10 "Hybrid" is used in a generic sense here; Harley et al. [137] use the term "hybrid viruses" to describe viruses that execute concurrently with the infected code.

11 From Thompson [322]; he simply calls it a Trojan horse.

12 This differs from Harley et al. [137], who define a dropper to be a program that installs malware. However, this term is so often applied to malware that this narrower definition is used here.

13 There are many other spamming techniques besides this; Spammer-X [300, Chapter 3] has more information. Back-door functionality left behind by worms has been used for sending spam in this manner [188].

14 Acohido and Swartz [2] mention a $2000–$3000 rental fee for 20,000 zombies, but prices have been dropping [300].

15 Cooke et al. [79] looks at botnet evolution, and takes the more general view that botnets are just zombie armies, and need a controlling communication channel, but that channel doesn't have to be IRC. There are also a wide variety of additional uses for botnets beyond those listed here [319].

16 In the anti-virus industry, people who analyze malware for anti-virus companies are referred to as "researchers." This is different from the academic use of the term.

17 This was one suggested way to find the Stoned virus [290].

18 Lyman [189], but this is common knowledge in the anti-virus community.

19 Diversity is usually a good thing when it comes to defense, and large sites will often use different anti-virus software on desktop machines than they use on their gateway machines. In a panel discussion at the 2003 *Virus Bulletin* conference, one company revealed that they used *eleven* different anti-virus products.

20 While the vast majority of interested parties want common naming, their motivations for wanting this may be different, and they may treat different parts of the name as being significant [182].

21 Having said this, an effort has been announced recently to provide uniform names for malware. The "Common Malware Enumeration" will issue a unique identifier for malware causing major outbreaks, so users can refer to highly mneumonic names like "CME-42," which intuitively may have been issued before "CME-40" and "CME-41" [176].

22 Of course, this begs the question of why such a central authority wasn't established in the early days of malware prevalence, when there was less malware and the propagation speeds tended to be much, much slower.

23 CARO, the Computer Antivirus Research Organization, produced virus-naming guidelines in 1991 [53], which have since been updated [109].

24 Vendor names have been removed from the results.

25 "API" stands for "application programming interface."

26 Not all variants necessarily come from the same source. For example, the "B" variant of the Blaster worm was released by someone who had acquired a copy of the "A" variant and modified it [330].

27 A few, like Gaobot, have hundreds of variants, and require three letters to describe their variant!

28 This example is from [47], again with vendor information removed.

29 Dellinger's "Virus 2" spread courtesy of the virus writer's friends [87], and secondhand stories indicate that Stoned was spread by someone besides its author [119, 137, 290]. Malware writers are rarely caught or come forward, so discovering these details is unusual.

100 For example, Adleman [3] and Cohen [75].

101 The details of the case may be found in [328]; [326] has sentencing information.

102 Paraphrased liberally from Virgil's Aeneid, Book II [336].

103 Anderson [12].

104 A sidebar in Harley et al. [137, page 60] has an amusing collection of suggested plural forms that didn't make the cut.

105 Benford [33] and Gerrold [118], respectively. Benford talks about his real computer viruses in this collection of reprinted stories.

106 As told in Cohen [74].

107 Skrenta [289] and Dellinger [87].

108 The whole sordid tale is in Rosen [267].

109 The original Core War article is Dewdney [91]; Darwin is described in [9, 201].

110 Bontchev [46].

111 Vossen [338] and van het Groenewoud [331], respectively.

112 This definition of spyware and adware follows Gordon [124].

113 Walker wrote a letter to Dewdney [340], correcting Dewdney's explanation of Animal in his column [92] (this column also mentions Skrenta's virus).

114 Chien and Ször [70] explain blended threats and the historical context of the anti-virus industry with respect to them.

115 Bontchev [44] and Lyman [189] describe the process by which a name is assigned.

116 VGrep was originally by Ian Whalley; this discussion of its operation is based on its online documentation [333].

117 This description is based on the CARO identifiers and terminology [109].

118 The Jargon File lists the many nuances of "hacker," along with a hitch-hiker's guide to the hacker subculture [260].