Computer Viruses - An Introduction

An in-depth overview on what computer viruses are and the different types of virus threats.

A computer virus is a program that explicitly copies itself. This may lead to it spreading from machine to machine and is typically done without the user's knowledge or permission. Viruses, by definition, add their code to your system in such a way that when the infected part of the system executes, the virus does also.

There are various types of viruses:

- Boot viruses place (some of) their code in the disk sector whose code the machine will automatically execute when booting. Thus, when an infected machine boots, the virus loads and runs. After boot viruses are finished loading, they usually load the original boot code, which they have previously moved to another location, or take other measures to ensure the machine appears to boot normally.
- File viruses attach to 'program files' (files containing executable or interpretable code) in such a way that when you run the infected program, the virus code executes. Usually the virus code is added in such a way that it executes first, although this is not strictly necessary. After the virus code has finished loading and executing, it will normally load and execute the original program it has infected, or call the function it intercepted, so as to not arouse the user's suspicion.
- Macro viruses are really just a type of file virus, but a particularly 'successful' type. They copy their macros to templates and/or other application document files. Although 'auto macros' were almost exclusively used by early macro viruses (often to ensure the virus' code is the first to execute when infected templates or documents were opened), several other mechanisms are also available – in fact, some of these, such as taking over standard internal functions of the host application (say the 'File Save' command) and installing default event handlers are probably more commonly used these days.
- Script viruses also became quite successful around the beginning of this century. This was mainly due to the increase in machines running Windows Scripting Host, which was first installed by default in Windows 98 and 2000 and with Internet Explorer 5.0 and later versions. Representing new types of 'program file', but with icons more like that of 'safe' text files, standalone Visual Basic Script (VBS) and JavaScript (JS) programs became a popular target of the writers of mass mailing viruses.
- Companion viruses take advantage of features of the operating system to be executed, rather than directly infecting programs or boot sectors. Under DOS and Windows, when you execute the command 'ABC', the rule is that ABC.COM executes before ABC.EXE (in the rare cases where both files exist). Thus, a companion virus could place its code in a COM file with its first name matching that of an existing EXE file. When the user next executed the 'ABC' command, the virus' ABC.COM program would be run (usually the virus would launch ABC.EXE once its code was finished so as not to arouse suspicion). This is known as the 'execution preference companion' method, but several other forms of companion infection are also possible.
- Worms are described by some antivirus researchers as similar to viruses in that they make copies of themselves, but different in that they need not attach to particular files or sectors at all. Once such a worm is executed, it seeks other systems – rather than parts of systems – to infect, then copies its code to them in such a way as to have the code execute directly from memory. This form of 'classic worm' is still very rare, with the 'Morris worm' (or 'The Internet worm') of November 1988 the best known of a small number of examples. More recently the term 'worm' has been taken to mean 'a virus that replicates across a network link', with the most common usage applied to

viruses that send many copies of themselves out attached to the infected user's e-mail.

Some viruses display obvious symptoms, and some cause damage to files in a system they have infected. While one or both of these features of a virus often capture the attention of the popular media, note from the preceding discussion that neither are essential in the definition of a virus. A non-damaging virus is still a virus, not a prank and, other things being equal, viruses without obvious symptoms are more likely to spread further and persist longer than those that rapidly draw attention to themselves.

There are no 'good' viruses, simply because a virus is code that was not intentionally installed by the user. Users must be able to control their computers, and that requires that they have the power to install and remove software; that no software is installed, modified, or removed without their knowledge and permission. A virus is surreptitiously self-installed. It may modify other software in the system without user awareness, and removal can be difficult and costly.

Many viruses cause intentional damage. But many more cause damage that may not have been intended by the virus writer. For instance, when a virus finds itself in a very different environment than that for which it was written, what was intended to be a non-destructive virus can prove very destructive. A good case in point is the boot virus. Few, if any, boot viruses contain code to damage computers running Windows NT however, with many boot viruses, when they infect an NT machine system recovery can be quite tricky.

Even if a virus causes no direct damage to your computer, your inexperience with viruses can mean that damage occurs during the removal process. Many organizations have shredded floppies, deleted files, and done low-level formats of hard disks in their efforts to remove viruses. Even when removal is done perfectly, with no damage to the infected system or files, it is not normally done when the machine is first infected, and the virus in that machine has had a few weeks to spread. The social costs of infection include a loss of reputation and good will. This last point is increasingly significant recently with the rapid increase in network-aware and data stealing viruses.

Virus Types

Viruses were traditionally classified into two major categories: 'boot viruses' (which infect the boot area of floppies and hard disks, and become resident and active at the time of booting the machine) and 'file viruses' (which infect one or more types of program files, and activate when the program is run). Until the mid-1990s the most common infections were of boot viruses, even though a huge majority of the known viruses were file viruses. This was because, in order to spread far, a virus needs a means of get from one machine to another. Floppy disks are commonly shared (although nowhere near as commonly now as in the earlier days of the 'PC revolution'), and while not all disks carry program files (reducing the chances for file infectors to spread), a diskette always has a boot sector and thus, potentially, a boot virus. Dependence on the common human practice of exchanging diskettes (and occasionally forgetting to remove them from the default A: boot drive) was an efficient means for a virus to spread when 'sneakernet' was the common means of file sharing and software distribution.

Macro Viruses

A macro virus consists of instructions in a macro language – for example, WordBasic or, more commonly now, Visual Basic for Applications – and generally resides in application document files (such as Word documents and Excel workbooks) or templates of these file types. Traditionally we have tended not to think of 'documents' as capable of being infected – they have generally been considered to be 'just data'. However, any application that supports macros that are embedded (or can be included in some way) in a 'document' file is potentially a rich platform for macro viruses.

Macros do not have to be embedded in document files for the hosting application and/or macro environment to be 'virusable'. For example, early in 1999 the first virus for the CorelScript macro language (included in several major Corel products) was discovered. However, due to CorelScript's reliance on separate macro code files this virus, CSC/CSV.A, and any successors seemed unlikely to be much of a threat to users of applications with CorelScript support. In fact, two years later there had been only one further CorelScript virus discovered.

Macro viruses pose several threats that are not shared by more traditional file infectors. The fact that they can be embedded in 'documents' which have not traditionally been seen as obvious virus carriers has already been discussed. Another threat is that the most popular and most widespread macro language is VBA and this is very easy to write programs in, be they viruses or not. This is, of course, part of the reason why VBA viruses became very common – many people who could not otherwise write a virus have been able to write a VBA macro virus. VBA is also a very powerful macro language (which also contributes to its popularity amongst professional programmers). It allows easy access to Windows APIs (such as Windows Automation (formerly OLE) and DDE) that can be difficult to program in lower-level languages. This also adds to its threat as a virus development platform because the term 'application macro' suggests to many users and administrators the keystroke recording type of 'macro' common in earlier and less powerful applications. If 'macro' means 'keystroke recording' to someone, they are unlikely to imagine much of a threat being possible in a 'macro virus'.

Finally, because documents are now even more widely shared than diskettes, through networks and the Internet (and particularly via e-mail), document-based viruses have become very common and are likely to maintain their high-profile in prevalence statistics in the immediate future.

File Infectors

Traditionally these have been viruses that attach themselves to (or replace) COM and EXE files, although in some cases they infect executable files with SYS, DRV, BIN, OVL and other less common extensions. The most successful of these file infectors are resident viruses, loading into memory the first time an infected file is run, and taking clandestine control of the computer. Such viruses commonly infect additional programs as they are run, or even just as directory listings are made. But there are many non-resident viruses, too, which simply infect one or more files each time an infected file is run. Amongst traditional EXE and/or COM infectors, these non-resident viruses have not been very 'successful' (in terms of prevalence of infection in the wild).

The advent of Windows slowed the development of both boot viruses and EXE infectors. Most boot viruses were incompatible with the 32-bit file-system extensions, first available as an option in Windows 3.1, then usually installed by default with Windows for WorkGroups 3.11 and finally standard fare in Windows 95 and its later replacements. 32-bit NT and Windows 2000 systems also tend to have serious trouble with boot infectors. The new, more complex and (initially) less understood 16-bit Windows 'New Executable' (or NE) and 32-bit

Windows ‘Portable Executable’ (PE) EXE file formats also slowed the early efforts of virus writers considering taking on these newer platforms.

Script Viruses

Viruses that infect various scripting languages, from the most basic (DOS batch) through to the sophisticated JavaScript (JS) and Visual Basic Script (VBS), are also a form of file infector. DOS batch viruses have always been of interest value only, never posing a serious threat due to the rather limited nature of the language itself. However, the increased complexity of the JS and VBS languages hosted under the Windows Scripting Host (WSH) certainly increased the attractiveness (to virus writers!) of script file infectors. For example, the addition of string manipulation operators makes many tasks that are all but impossible in a DOS batch script trivial in JS and VBS scripts. Further, the integration of the scripting engine with many standard OS functions such as file system, networking and registry interfaces opened up a whole slew of opportunities – much like VBA macro viruses, these scripting languages have allowed ‘non-programmers’ to become virus writers. This effect is exacerbated by the ease with which existing scripts can be changed, as the flood of trivial variants following the initial VBS/LoveLetter outbreak showed. Finally, no discussion of script viruses would be complete without pointing out that any modestly sophisticated scripting language that allows access to its hosts’ file systems is a potential virus host. Thus the scripting languages of popular IRC client software such as mIRC and Pirch are virusable and have been targeted because of the popularity of those applications among naïve (and thus easily tricked) users.

Boot Sector Infectors

Every logical drive, both hard disk and floppy, contains a boot sector. This is true even of disks that do not hold a bootable operating system. This boot sector is the very first sector of the logical drive and contains specific information relating to the formatting of the disk, the data stored there and on PCs is expected to contain a small program called the boot program.

Boot programs are expected to load the appropriate operating system files when an attempt is made to boot from a disk, and typically display the familiar "Non-system Disk or Disk Error" message if the operating system files are not present. It is increasingly common for the boot program on diskettes to simply display a message warning that the diskette does not contain a bootable system – such boot programs will be overwritten with an appropriate one should you reformat the diskette using an option that copies operating system files to the disk.

Regardless of its precise nature, this is also the program that is infected by boot viruses. The most common way to contract a boot virus is by leaving an infected diskette in a drive and rebooting the machine. Most PCs default to attempting to boot from the first diskette drive in the system (the drive normally known as A: under DOS and Windows) if there is a diskette in the drive – if there is not, the attempt to boot continues with the Master Boot Record of the first hard drive. There is very little checking done at this point – the program in the boot sector is read and executed. If it is a virus it is loaded into memory and executed just as if it was any other boot program. If it is a virus, it usually then infects your hard drive.

Remember, because every disk has a boot sector, it is possible (and common) to infect a machine from a ‘data disk’. Boot viruses infect the boot sector of floppy disks; some of them, such as Form, also infect the boot sector of hard disks whereas others, such as Stoned, infect the hard disk’s Master Boot Record.

Master Boot Record Infectors

The first physical sector of every hard disk (Cylinder 0, Head 0, Sector 1) is known as the Master Boot Record (MBR – also as the Master Boot Sector, MBS), which in turn contains the disk's Partition Table. The Master Boot Record, like the boot sector of a diskette, holds a small boot program. However, unlike on a diskette, this boot program is not usually directly concerned with locating and starting the operating system.

Boot programs in Master Boot Records usually locate the active, or boot, partition by inspecting the Partition Table, and then load the contents of the boot sector of that partition. A partition is really just a logical drive, so its very first sector is a boot sector that, much like that of a diskette, contains information relating to the formatting of the disk, the data stored there and a small boot program. Assuming the disk is set up properly, a valid boot sector will be found in the first logical sector of the boot partition and the boot program from the Master Boot Record will load that boot program into memory and execute it. As you may expect, the boot program from the boot partition then tries to locate the operating system files it expects to find on the disk and load and execute them.

Master Boot Record viruses are usually contracted in exactly the same manner as boot sector viruses – by leaving an infected diskette in the A: drive and rebooting the machine. When the boot sector program from the diskette is read and executed, it is the virus being loaded into memory and it will infect the MBR of your hard drive. Again, because every disk has a boot sector, it is possible (and common) to infect a machine from a 'data disk'.

Both boot sector and MBR infectors are largely thwarted by the simple expedient of disabling a PC's ability to boot from its diskette drive(s). Most PCs manufactured from the early/mid 1990s have a provision to configure the BIOS to not attempt to boot from diskette as its first choice. Many even have a configuration option to prevent the BIOS from trying to boot from the diskette drive(s) at all. Surprisingly few system administrators seem to take advantage of this simple, yet effective and free, 'antivirus technique'.

Multi-partite Viruses

Multi-partite viruses are 'combination infectors', infecting more than one class of basic target listed above. Thus, a virus with code parts that infect both files and boot sectors is multi-partite. Before the rise of macro viruses, several of the most common file infectors (for example Junkie) were actually the file infector parts of multi-partite viruses that had leveraged the distribution advantage attributable to their boot infector components. These viruses became common because of their boot virus components. More recently we have seen complex forms of multi-partism with, for example, viruses that infect EXE files and insert droppers as macros in suitable document files.

Note however, that there is no hard distinction between multi-partite viruses and 'non-multi-partite' viruses. For example, a macro virus that infects Word and Excel (e.g. O97M/Jerk), Word and Access (097M/Cross), Word and Project (O97M/Corner), Word, Excel and PowerPoint (e.g. O97M/Tristate) or other combinations of Microsoft Office and related products is generally not considered multi-partite. Because all parts of these viruses depend on the Visual Basic for Applications macro 'platform' (albeit to a version of that platform tailored to each of the hosting application's requirements), these viruses are usually not considered as infecting more than one target type. Such viruses are sometimes referred to as cross-platform viruses, but that is also contentious as the platform is essentially the same (as it is for a Word VBA virus that works equally well under Word 97 and Word 2000 for Windows and Word 98 and Word 2001 for Macintosh). Such viruses that work 'between' Office applications are often called cross-infectors or cross-application-infectors which is a suitable term.

Multi-partite viruses are rare, although in the past the file infectors most commonly seen in the wild were the file infecting components of file and boot multi-partites.

Worms

As mentioned above, the term ‘worm’ does not have a firm definition. However, in the late 1990s it was widely adopted as meaning something like ‘a virus that spreads via network connections’. One immediate weakness of this, as a definition, is that most file infectors blithely infect suitable host files on any drive available to them, including files on mapped network drives. Thus, given an environment where client machines commonly map drives to network shares (i.e. most corporate LANs), most file infectors would be worms. As the point of those adopting the term ‘worm’ was to highlight viruses that would spread rapidly outside their initial hosts or initial host networks, the informal definition was changed to something like ‘a virus that overtly spreads via network connections’ or ‘a virus that overtly spreads via external network connections’.

Worms, under this definition, really came to the fore with the release and widespread distribution of W97M/Melissa.A in late March 1999 (and the apparently inevitable rash of copycats soon thereafter). Many anti-virus researchers at the time were not surprised by what Melissa did, but rather by the fact that Melissa’s writer had the temerity to release the virus. He was subsequently tracked down, arrested and pleaded guilty (however, his sentencing seems to be in some perpetual-delay machine).

Since W97M/Melissa, many virus writers have produced hundreds of viruses that use Microsoft Outlook’s Automation interface to open each address list available to Outlook then send e-mail messages with copies of the virus to the first n addresses in each list (where n is anything from 1 to the number of addresses in the list). This scheme has been implemented in VBA macro viruses (such as Melissa itself), script viruses (the VBA code for this functionality can almost be cut’n’pasted from Melissa’s source into a VBS virus), and executables (again, functioning VBA or VBS code can be taken as a close template for use in Visual Basic code to be compiled to EXE format).

The other major variation on this theme has been the patching of a core Windows networking component to do the worm’s dirty work. This was pioneered by Win95/Ska, which patched WSOCK32.DLL so as to intercept SMTP (e-mail) and NNTP (Usenet News) communications. When Ska detects new messages being sent to either an e-mail or news server it saves address and Subject: information from the headers of those messages. It then sends its own message, containing an attachment (which is a copy of its installer) and the original message to the addressee or newsgroup the original message was intended for. Ska became one of the most common and widespread of viruses ever, and by mid-2000 several more viruses had been written that copied the idea of intercepting network communications at the Winsock level.

Other forms of (mainly) e-mail worms have also been seen, but have mainly not succeeded due to targeting less popular or less sophisticated e-mail clients than Microsoft Outlook. Two exceptions to this are JS/Kak and several implementations of the 'open share crawler' attack.

JS/Kak was the first e-mail worm to use Microsoft's Outlook Express in its distribution mechanism. Although probably used significantly less in corporate settings (which tend to use Outlook because it is part of Microsoft Office, or a third-party e-mail program), Outlook Express is the default e-mail client software for millions of personal and small business computer users because it is installed by default with Internet Explorer. Kak modifies its victim's Outlook Express configuration so a default signature is attached to outgoing HTML-format messages. The HTML file used for this signature contains an embedded JavaScript program which is a copy of the virus. Kak has a further distinguishing feature – it is one of a small number of viruses that depends on a security flaw in a product for its infection procedure to work. You can read more about this in the Computer Associates Virus Encyclopedia entry. Its infection and distribution mechanism means Kak is 'silent' to its victims, as default Internet Explorer and Outlook Express security settings are lenient, allowing active code in e-mail messages to run without warning the user. The large, and mostly naïve, userbase of the affected versions of Internet Explorer and Outlook Express constitutes a huge potential infection base for Kak and during much of 2000, Kak ranked near or at the top of most virus prevalence charts, and managed that despite being (or because it was) most densely concentrated in the sector of the computer-using population with the lowest uptake of virus detection software.

The 'open share crawler' attack was probably first successfully implemented in VBS/Netlog. This attack uses the simple expedient of randomly selecting tracts of the IP network address-space then attempting to connect to a Microsoft Network share named 'C' on whatever machine (if any) happens to be on each of those addresses. Doing this, VBS/Netlog was surprisingly successful at finding Windows users that not only were sharing their whole C: drive with the Internet, but were doing so without any password-protection in place. Netlog thus found thousands upon thousands of PCs onto which it copied itself. Choosing what seemed to be the startup directory of these machines, the virus was then assured of a new spurt of activity when the victim machine was next restarted. Many virus writers seem to like nothing better than other virus writers' success, and this simple attack has been copied in many other script and executable viruses. A minor modification of it appeared in ExploreZip and a few later viruses/worms, whereby the virus specifically interrogates Windows networking APIs to find all the machines the host explicitly knows on the network and these are then attacked by the virus' infection and/or payload code. This more directed form of the attack has the advantage of almost guaranteed connectivity over spending a huge amount of time finding the sparsely distributed vulnerable machines on the Internet.

Non-viral Malware

Aside from the issues and virus trends discussed above, since early-1999 there has been substantial growth in the development of certain kinds of non-viral malware. In particular, the class of program that has come to be known as 'remote access Trojans' (and occasionally 'remote access trapdoors' and 'backdoors') or RATs has burgeoned to the point that there is now something of a cottage industry devoted to producing them.

The basic idea behind a RAT is that an ‘attacker’ attempts to trick or ‘social engineer’ a victim (who may just be a random Internet user) into running a program on their computer. This program (the RAT itself) then opens up a ‘backdoor’ or ‘trapdoor’ into the victim’s computer whereby the attacker (or anyone else that discovers the listening network port) can attach to the victim’s computer and execute whatever commands they wish, copy files to and from the computer, send the user messages or pretty much anything else you can imagine a computer program being able to do. Usually a special client program is needed to connect to the RAT, and these are different for each RAT. Early, well-publicized RATs such as NetBus and BackOrifice have more recently given way to dozens of other RATs that have become very common. The extent of the problem caused by RATs is difficult to gauge as they tend to be launched against personal and small business users. Corporate networks often block their effectiveness through the use of firewalls and/or through intrusion detection systems noticing their telltale network activity. However, it seems there must be a reasonable number of compromised machines around the Internet (or at least a belief that there is) as there is evidence of considerable scanning of dial-up, cable and DSL IP address-spaces for machines with open ports commonly associated with well-known RATs. Further evidence suggesting RATs may be commonly accessible at random is that by mid-1999 some viruses started to include code to search for the default port of the most common RATs, copy themselves to the new machine then run the infected file so as to infect that machine as well.

RATs and the network crawler type worms have become so ‘successful’ because of the expansion of ‘always on’ Internet connection technologies such as cable-modems and DSL. The tendency to keep such connections up, coupled with their increased bandwidth and the increasing propensity of naïve users to implement Microsoft networking without taking the most rudimentary of security precautions is bound to see these problems get worse before they get better.