Viruses And Antiviral Programs

Shahram Faridani

19.11.2007

Contents

1 Preference

Computer viruses are a major problem in modem day computing. In A world, that data and communication is may be the most valuable thing and information is power and authority losing data regardless in which scale is very critical. For this reason are viruses a real problem. This artikel talks on one hand about viruses, particularly their development and on the other hand about antiviral programs and their functionality.

2 A brief history of CIH

CIH is a hardware destructive virus, it was discoverd first in 1998 and has been among the ten most common viruses for several months. It was written by Chen Ing Hau from Taiwan as he was a student. According to China Times he has never been punished for his act, thus there is a case against him, which is pendding at the time. The name Chernobyl was coined because of the complete coincidence of the payload trigger date in some variants of the virus, actually the virus writer's birthday and the Chernobylaccident, which happened in Ukraine on April 26, 1986. It is at the same time the birthday date of the writer. The name Spacefiller is driven from the virus behavior, it checks the uninfected file for free space and get in there, so it stays unrecognizeable. CIH is obviously not the first hardware destructive virus. There is another similar virus called Aragon first discovered in 1992. It infects boot records and has the payload of resetting the BIOS. On computers running outdated BIOSes, this can result in hardware damage (the CIH virus being famous for this).

3 Propagation and technical specification

Any program you receive from outside your computer could potentially be infected. Once you are infected, the virus will soon spread throughout your computer, and so the chance of your passing an infected file to someone else is high.

Even though the first reports of CIH appeared only around the middle of 1998, the virus reached the Number Two spot on the Sophos Virus Top Ten for the whole of 1998. This means it is very common indeed. Programs infected with CIH have been seen on a number of cover CDs from reputable magazines, and on a number of reputable websites. This has certainly helped the virus achieve wide distribution. There were also several commercial sources, which were infected such as:

- Yamaha shipped an infected version of a firmware update software for their CD-R400 drives.
- Widely spread demo version of the Activision game SiN was infected as well.
- IBM shipped a batch of new Aptiva PCs with the CIH virus pre-installed during March 1999, just a month before the virus activates destructively.

Normally, CIH simply spreads itself. But on certain trigger dates, it detonates its warhead. The warhead wipes out your hard disk, and then tries to overwrite the computer's BIOS chip.

Once the BIOS is overwritten, you will be unable to use your computer at all. Repair involves physically removing the BIOS chip and replacing it with a fresh one. On some computers, the BIOS chip is not removable, so it can only be replaced by swapping the entire motherboard. There are several variants of CIH, with different trigger conditions. The best known, and most widespread, variant will detonate on 26 April. Other variants detonate on 26 June, or even on the 26th of any month. CIH spreads under Windows 95 and Windows 98. DOS and Windows 3.x cannot spread CIH because they cannot run Windows 95/98 programs. Windows NT cannot spread CIH because the virus uses programming tricks that do not work under NT. The virus can infect Windows NT programs, but such programs will no longer run, and will therefore not be infectious themselves .

Due to its infection mechanism, most antivirus software can deactivate the CIH virus but cannot completely clean infected files.

- Ramifications:

    1. infected files cannot be restored to their original state, and will therefore produce different hashes or checksums than the original file, which could cause the file to fail integrity checks because the virus signature is still present within the file, the antivirus software will continue to flag infected files, usually as "CIH (inactive)".
    2. The only way to get completely rid of CIH is to replace the affected files with copies of untouched originals. For systems that were roughly infected, this likely entails a complete reinstallation of the operating system and software.

4 Functionality of the virus

This is a descreption of first version, which has been released on 4.April 1998.

1. Create the virus program.
2. The virus modification IDT to get $Ring0$ privilege.
3. Virus code does not reload into system.
4. Call $IFSMgr\_InstallFileSystemApiHook$ to hook file system.
5. Modifies entry point of $IFSMgr\_InstallFileApiHook$.
6. When system opens existing PE file, the file will be infected, and the file does not need to be reinfected.
7. It is also infected, even the file is $READ - ONLY$.
8. When the file is infected, the modification date and time of the file also do not be changed.
9. When my virus uses $IFSMgr\_Ring0\_FileI0$, it will not call previous FileSystemApiHook, it will call the function that the IFS manager would normally call to implement this particular I/O request.

There are some other recent variations of the virus in the wild right now. The virus got a new look.It scans fot the security holes in host OS. This version caused: Ip conflicts,Font removal, System Netbios Conflicts. it actually does not harm a system, but prompt conflicts on port 139 of the windows systems.

5 Coping with malicious software

In this section we are going to talk about viruses and the different ways, how we can deal with them.

5.1 Halting problem

Fact is a number of real-world problems that would be nice to be able to solve are not solvable.

5.1.1 Definition

The problem of determining in advance whether a particular program or algorithm will terminate or run forever. The halting problem is the canonical example of a provablyunsolvable problem. Obviously any attempt to answer the question by actually executing the algorithm or simulating each step of its execution will only give an answer if the algorithm under consideration does terminate, otherwise the algorithm attempting to answer the question will itself run forever.It can be represented as a set called halting set: $k := (i, x)|programiwilleventuallyhaltifrunwithinputx$.

Is there an algorithm to determine whether any arbitrary program halts? Turing proved the answer is, no. Since many questions can be recast to this problem, we can conclude that some programs are absolutely impossible, although heuristic or partial solutions are possible. The first step in coping with a malicious software is, we should find out if an arbitrary program is infected indeed. This problem is known as undecidability of virus detection. Adleman proved: If an algorithm exists that can prove the presence of viruses in general case than exists an algorithm that solves the halting problem (Reductio ad absurdum - proof by contradiction).

6 Virus identification and obfuscation

There are several challenges surrounding the design of robust computer viruses and designing algorithms to detect viruses. For a virus writer it is very imortant to write a virus, which is difficult to detect and a antiviral program developer tries to find a concept which can detect viruses as much as possible.

6.1 Virus

A simple virus is a code that can make a copy of itself over and over again, it is relatively easy to produce(also called first gen. viruses). Even such a simple virus is dangerous because it will quickly use all available memory and bring the system to a halt. The modern viruses can even more. They are capable of transmitting themselves across networks and bypassing security systems.

6.2 String matching

String-matching consists in finding one, or more generally, all the occurrences of a string (more generally called a pattern) in a text. Idea: take a string of bytes from the virus that does not change from generation to generation and use it to find other instances of the virus. It is the most easiest method to find a virial code. Pattern matching is also a method used by viruses.Also a method that is use by viruses to identify themselves.Viral self-identification is necessary to prevent overpopulation and to prevent a host being infected several of times. String matching is very usefull by detecting first gen. viruses. The algorithm searches for a certain signature. Virus writer use an algorithm to prevent multi-infection of a file. At the same time it is the weak spot of the virus. It gives the antiviral analyst a detection algorithm, when the virus is found. Once the virus is find, it has the benefit of identifying it in the wild. The antiviral programs has a data base at their disposal, they performe a scanning process to find any simular signature. The result of such a scanning is not always trustworthy. It produces sometimes some false positive result for instance by new patches for programs.

7 developed viruses

Virus writers are getting smarter, they find more countermeasures to scanners. A very effective alternative is using a code that has the capability of changing its own code, this allows the virus to have hundreds, sometimes thousands, of different variants, making it much more difficult to notice and/or detect. These types of viruses are called Polymorphic viruses or evolutionary viruses. Polymorphism of a virus code may make it harder, and sometimes even nearly impossible, to use virus string detection to identify an infected file

7.1 Polymorphic Viruses

A polymorphic virus consists of two parts:

1. Header
2. Body

the header decrypts the body in memory .Once the body is decrypted the header transfers control to the body. The body performs the normal viral operations. When the body is finished sends control to the host program.

A virus code may be encrypted and then attached to a file in its encrypted form. The encryption technique may be polymorphic (for example, vary from one generation to another, so that the byte stream of each new viral infection is different from its predecessor). Each time the virus runs it will decrypt its own code in memory before control is transferred to it. Only a small part of a virus code, which is known as a "decryptor", may be constant from one generation to another.

Polymorphism of a virus code may make it harder, and sometimes even nearly impossible, to use virus string detection to identify an infected file. In this case The decryptor may be generated by the virus code in such a way that it contains different code for each new generation

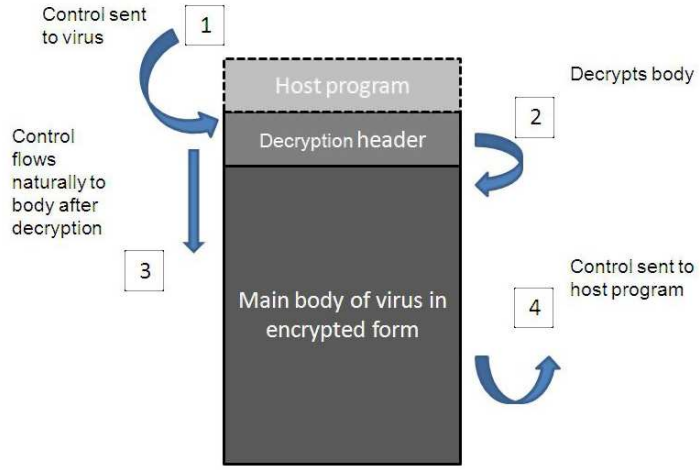

Figure 1: Typical Polymorphic Virus

of the virus. The decryptor's byte stream may be different for every new infection of the same virus. Under such circumstances, there are no constant byte streams that can be used for virus identification in an entire virus code. It is a very easy and at the same time a very effektive way to counter measure the string matching, in this case the antiviral program has no chance to identify the string, and the virus keeps its funtionality totaly.

7.2 Metamorphic viruses

A Polymorphic virus changes its appearance in host programs.The virus uses a decryption routine (known as the "decryptor") in order to do this. A metamorphic virus, by comparison, is a virus that also changes its appearance in host programs, however it does so without necessarily depending on encryption. The difference in appearance comes from changes made by the virus to its own body.

There are several techniques that can produce such an effect. The viruses use several techniques to morphe the host program. One of these morphing techniques used by metamorphic viruses is with the insertion and removal of "garbage" instructions. These are instructions that have no effect on the function of the virus, but simply take up space and which can make analysis more difficult when they appear in large quantities. Another technique is to change the basic encoding of instructions at the opcode level. That is, switching between two different opcodes that are functionally-equivalent. Perhaps the most complex transformation of a metamorphic virus is the replacement of entire blocks of logic with functionally-equivalent blocks of logic. Consider the task of adding x to y and z. One expression of this is "$(x + y) + z$". However, an alternative expression is to replace the single adition with a repeated addition instead: "$t_1 = x + y$"*and*"$t_2 = t_1 + z$. Both expressions will result in the same answer, yet they look very different. Here is an example of using garbage instructions.

8 Smart virus scanner

The Viruses are getting smarter and smarter,actually it is because the virus writers are always at least one step ahead, and it makes the whole thing much difficult. It is obvious that antiviral programs should neccesserly get smarter. It is already very difficult to detect the new gen-viruses, it means the conventional methods are not very helpful in this case. There are some very aspects, which can be used for better virus detection. Better virus detection tool:

- Analyze the program structure (instead of signature matching).
- Check for irrelevant instructions.

  What are irrelevant instructions?

  1. NOPs
  2. Control flow instructions that do not change the control flow.
  3. E.g. jump/branches to the next instructions.
  4. Instruction that modify dead (are not in use) register.
  5. Sequence of instructions which do not modify the architectural state, for instance. $add\ ebx, 1\ sub\ ebx, 1$

- Check whether viral properties are present in a program
  E.g. program writes a executable.
  E.g. program monitors as executables are loaded into memory and changes them.
  E.g. program behaves just like virus xyz.

- Check if the program exhibits those properties.
  If yes $\longrightarrow$ is infected.

8.1 Heuristic virus detection

This is a generic method of virus detection. Anti-virus software makers develop a set of rules to distinguish viruses from non-viruses. Abnormal program behavior plays a very important role here.Viruses perform actions that are not typical of normal programs. Result is there are several heuristics that can be used to detect them.

8.2 Abnormal program behavior

Abnormality of a program behavior can be an indicator for melicious attack. For example interrupts.

8.3 Iterrupt

Application programs access operating system routines using software interrupts. E.g. keystroke from the keyboard. The program places various parameters in CPU registers and then execute a software interrupt instruction (interrupt handler). CPU (temporarily) suspends the operation and look up the address of the operating system routine for that interrupt in the interrupt vector table. It's a very effective way to modify the operating system service routines to be more suspicious of the applications that call them. There is a standard systems programming technique to add such functionality, it is called interrupt patching. Interrupt patching has to major advantages:

1. can monitor the activities.
2. Know the data flow, it is easier to distingiush a melicious activity from a normal one.

provided that, monitoring the activities is possible. Depends on the ISA of the machine it is sometimes very difficult or even impossible to use this method. There some other complementation techniques wich can improve our defence mechanism against viral attacks.For instance visualizing executable viruses using self-organizing maps. This is a very complicated method and i'm going to mention it's functionality very briefly. Each virus has it's own character and can not hide it through SOMs. Like DNA , it's an unique generic code. SOMs can figure out the family without knowing it's signature, which is used by disposing the virus.
Aim:
design the SOM in a way that neurons will flag the presence of peculiar patterns in Windows executable files and that the position of the active neurons will reflect the position of potentially malicious content in the file.

8.4 Actual state of affairs

The main problem by scanners are that, Anti-virus software suffers from more problems than not being able to detect viruses. Many copies of anti-virus software are unable to detect even old viruses, because end users frequently forget or simply don't update their virus scanner's virus databases until it's too late. On-demand scans are rarely performed because they're slow and hog resources while running, so dormant viruses tend to have a rather long life. On-access scanners aren't free of troubles, either - some consume too many resources, so many users are tempted to disable them if they're on a slower machine. Finally, while anti-virus software may become extremely good at sensing virus activity, there are always new security holes to exploit in operating system and networking software that would give viruses another entry point that bypasses the anti-virus software. It means an updated anti-virus software is not a 100% security against the viruses in the wild, but it takes the bulk of responsibilty in the unfair fight against the melicious software, which lurk to do something awful.

9 References

1. Malicious cryptography exposing cryptovirology, Dr. Moti Yung, ISBN: 0-7645-4975-8 Published by wiley publishing, Inc., Indianapolis, Indiana.

2. http://www.f-secure.com, F-Secure Corporation

3. Virus Scanning as Model Checking, Mihai Christodorescu, University of Wisconsin, Madison

4. Computer organization and design, the hardware/software interface. David A.

5. Patterson and John L. Hennessy. Second edition, Morgan Kaufmann Publishers. ISBN: L-55860-428-b.

6. Visualizing Windows Executable Viruses Using Self-Organizing Maps, InSeon Yoo, Telecommunications, Networks and Security Research Group, Department of Informatics, University of Fribourg.

7. www.geot.com.

8. http://www.mirrors.wiretapped.net/security/info/textfiles/DOJ/DoJ5−08.txt