通信情報保護學會論文誌
第7卷 第3號, 1997. 9

Managing the Heterogeneous File System for Anti-Virus

Kyung Su Kim*, Seung Jo Han*, Pan Koo Kim*

Abstract

Computer viruses are increasing in number and are continually intellectualized as well. To cope with this problem, anti-virus tools such as a scanner and the monitoring program have been developed. But it is not guaranteed that these softwares will work in safety under MS-DOS' control. If the virus is run first, it can avoid the monitoring of anti-virus software or even can attack the anti-virus software. Therefore, anti-virus programs should be run before the system is infected. This paper presents a new PC starting mechanism which allows the PC system to start from a clean state after booting. For this mechanism, we build a new disk file system different from DOS' file system, and manage the two file systems heterogeneously. Our system is strong against boot viruses and recovers from infections automatically.

1 Introduction

At the beginning of 1994 the number of known MS-DOS viruses was estimated at around 3,000. A year later, in January 1995, the number of viruses was estimated at about 6,000. Several anti-virus experts expect the number of viruses to reach 10,000 by the end of the year 1996. This increase of viruses, which grows rapidly, does cause problems to anti-virus software, especially to scanners. To the present, users mainly depended on scanning programs which have ability to scan virus' character strings and fix the infected program. The rapid growth in the number of viruses means that scanners should be updated frequently enough to cover new viruses. Also, as the number of viruses grows, so does the size of the scanner or its database. In some implementations the scanning speed suffers. This approach is vulnerable to unknown viruses, especially such as polymorphic and stealth viruses which are a big burden to the vaccine programmers.

It was always very tempting to find an ultimate solution to the problem, to create a

† 이 논문은 1995년도 조선대학교 학술연구비의 지원을 받아 연구되었음
* 조선대학교

generic scanner which can detect new viruses automatically, without the need to update its code and/or database. However, heuristics suffer from a moderate false positive rate. This type of software is reactive rather than proactive, in that, a virus attack will be detected after it happens. So, it is necessary for the system to monitor virus activities, and be detected before it is infected. To do that, some softwares have been developed such as virus monitoring systems which install themselves as memory-resident TSR (terminate-stay-resident) programs in a manner similar to some PC utilities like Borland's Sidekick. They intercept and monitor disk I/O, trying either to monitor the systems integrity or to detect virus activity. However, these program themselves have some holes and are not safe because of the characteristics of MS-DOS, which does not have a protection mechanism for the systems resources. A new system mechanism which can start the PC from a clean state(i.e. virus-free) is needed.

This paper presents a new system mechanism which can start the PC system at a clean environment after booting. For this, we build a new disk file system different from DOS's file system and manage the two file system heterogeneously. This new mechanism enables detection and recovery of the boot viruses automatically and also performs the anti-virus program itself with integrity.

2 Anti-Virus Technologies

2.1 Scanners

A scanner relies on the knowledge of the known virus "pattern". When a new virus appears in the wild, it is analysed, and a characteristic pattern of some 10 to 16 bytes is recorded within a database. The virus scanning program will scan all the executable on a disk, including the operating system and the bootstrap sector(s), and then compares their contents with the known virus patterns. Of course, this type of program can only discover known viruses and as such it has to be continually updated with new patterns when new viruses appear. This is the main problem with this type of program. At the moment there is no national or international databank of virus patterns. Also the growing number of viruses means that scanners should be updated frequently enough to cover new viruses. This approach is vulnerable to unknown viruses, especially such as polymorphic and stealth viruses, and it is a big burden to the vaccine programmers.

Computer scientists have been trying to find an ultimate solution to this problem, for creating a generic scanner which can detect new viruses automatically. It is based on analysing a program for features typical or untypical for viruses. The set of features, possibly together with a set of rules, is known as heuristics. Heuristics are added to the anti-virus and attempt to infer whether a file is infected or not. This is mostly often done by looking for a pattern of certain code fragments that occur mostly often in viruses, but not in bona fide programs. Heuristics suffer from a

moderate false positive rate. This type of software is reactive rather than proactive, in that, a virus attack will be detected after it happens. To make matters worse, most scanning programs are slow. And the crucial flaw of scanners is that some viruses bypass or deceive the control of anti-virus programs and attack the anti-virus software itself, if system has been infected already. So, it is necessary that anti-virus program should be run first before it is infected to protect system against virus infection.

2.2 Monitoring System

Computer viruses must replicate to be viruses. This means that a virus is observable by its mechanism of replication. A monitor is a memory-resident(TSR) program which monitors system activity and looks for virus-like behaviour. In order to replicate a virus needs a copy of itself. Most often viruses modify existing executable files to achieve this. So, in most cases, monitors try to intercept system requests which lead to modifying executable files. When a suspicious request is intercepted, a monitor typically alerts a user and, based on the user's decision, can prohibit such a request from being executed.

While this approach is attractive in theory, unfortunately there are no fixed sets of rules regarding what a virus should do or should not do. As a result, false alarms can result from legitimate program activity which is misinterpreted by the anti-virus program. Conversely, any virus which does not comply with the monitoring program's concept of virus activity will be ignored. The monitoring activity also degrades system performance and can be incompatible with network software, certain application programs and so on. The greatest drawback of memory-resident programs is that any intelligent virus program can easily bypass or disable them if it has been infected already. The mechanism used by anti-virus programs for intercepting disk read and writes, i. e. to change the vectors in the DOS interrupt table, is exactly that used by most virus programs. Also, some virus' use quite effective and sophisticated techniques, such as tunneling, to bypass possibly present monitors. So, it is requested that monitors should be started first before system is infected.

2.3 Integrity Checker

This relies on the calculation of a checksum of any executable on the system followed by periodic recalculations in order to verify that the checksum has not changed. If a virus attacks an executable, it will have to change at least one bit inside the executable, which will result in a completely different checksum. This means, when an integrity checker is first installed to your system, you need to run it to create a database of all files on your system. Then, during subsequent runs the integrity checker compares files on your system to the data stored in the database and detects any changes made to the files. Also, the results of the checksumming algorithm must not be easily reproducible. Unlike a monitoring program, it is much more difficult for a virus to bypass an integrity checker,

provided you run your integrity checker in a virus clean environment, i. e. having booted your PC from a virus free system.

3 A New System Mechanism

Since the computer virus is one of the programs, it can act free under DOS' permission. Observing the bounds of virus' activity, we notice all virus programs can access any part of disk and change or ruin them. Also, because of the absence of a protecting mechanism for use of memory, virus can control any program at its will. So, it is not possible to set a perfect program to protect from virus attack. To prevent the onset of a virus, the monitoring program or checking program should work out of virus' interruption. However it is not ensured that programs in DOS are processed in safety for there is no protection mechanism in DOS. Consequently, if we won't modify the DOS, we have to protect them with external system or installation.

The purpose of this study is to make it difficult for virus' to access, while maintaining information for an integrity checking by using a different file system from a DOS file's, instead of using other operating system or without any modification of DOS.

Now, how can it be possible to check or prevent infection out of DOS control? The answer is that virus checking or monitoring mechanism can run free from DOS control. In the sequence of DOS booting, the master boot loader is the program which runs first before DOS' loading. If this loader makes the DOS booting virus-free, it is possible that an anti-virus program can be run from a clear environment. If there are anti-virus programs which run in DOS control, those programs can start from the clear state and can be secured. All programs that are needed for monitoring or checking can be secured before DOS runs. To do this work, we divided the disk into two parts and made one part a safe zone to maintain security information. This partition is treated with different way from the DOS disk file system. To manage the new disk file system, we modify a master boot loader and implement an installation program which sets up the security zone.

3.1 Disk File System

We divide the disk into two parts, one for DOS partition and the other for security part(zone). Partition I is same as DOS's disk file system. Partition II is a simple disk file system different from DOS's that has different structure. The content of all files in the second partition is encrypted to be veiled by virus programers. Partition II includes all information to be secured and files for backing up. And also this area can be only accessed by our set-up program which knows its structure. Therefore, any other programs could not access this partition for they do not know the structure of file system and the position of data, thus do not infer the content of disk by using Disk Editor, due to the encrypted data. Consquently, partition II is a safe area. Figure 1 shows the whole situation in partitioned disk file system.

	Region Name	Contents	Remarks
Partition I	Modified Master Boot Sector	Modified master bootstrap loader Check-sum informations Partition table	Safe
	Boot Sector	DOS bootstrap loader DOS related informations DOS versions, etc	Safe
	FAT 1	Disk usage(bad,use,free)	·
	FAT 2	Copy of FAT1	·
	Root Directory	Root directory file information	·
	Data Region	Data storage area	·
Partition II	Security Zone(Region)	Simple disk file system (different from DOS) Files to be secured Backup files	Safe (veiled)

Figure 1. Partitioned, new disk file system

3.2 Modified Master Boot Loader

The function of the modified master boot loader is as follows:

- Checks the integrity of the RAM-resident program, as comparing the file in DOS with that in security zone, and then loads it into memory.
- Before loading DOS boot loader in DOS boot sector, checks the integrity of DOS boot sector or DOS system programs(IO.SYS, MSDOS.SYS, COMMAND.COM).
- Checks the integrity of the other important programs and data files

To guarantee the safety in that function, we save and maintain information in a different format from DOS' in partition II of the Figure 1. It also keeps check-sum information to check contents of DOS boot sector and checks whether the files are infected before booting or not. Adding to this, it makes integrity checking possible by keeping check-summing information on DOS files such as IO.SYS, MSDOS.SYS, COMMAND.COM. Because virus programs don't know the mechanism of partition II, it can not infer security information within the partition II.

The state of the system can be a safe one after booting with these functions. The master boot loader can be changed by a virus program, but the routine for protection is added to the ROM BIOS that has been made recently. If users fix "virus enable" by using Setup program in booting, they can have warning message when the master boot sector is going to be changed. By this way, any boot virus can not change the master boot sector. Consequently, this modified loader cannot be

changed once it occupies the master boot sector without user's permission. If master boot loader works connected with ROM BIOS, system can be booted in a safe state for virus can not attack ROM BIOS. Functions for the modified master boot loader are like Algorithm 1.

Algorithm 1. Modified Master Boot Loader

```
Algorithm : Modified Master Boot Loader
 /* Integrity checking for all system files   */
 /* Loading RAM-resident programs into memory */
 /* Loading DOS boot loader into memory */
 {
   Read security information(from part.II) into mem.;
   if ( there exists DOS boot record)
   {
     if ( check-sum of DOS boot record
                          != stored check-sum)
     {/*DOS boot sector is infected by boot virus */
       Alert the modification of boot record to user;
       goto recv;
     }
     else
     { /* Integrity checking : compare security
      inform.(partitionII) with check-sum inform.
      for the system files and
      some important files*/
       IO.SYS file is modified or not?;
       MSDOS.SYS file is modified or not?;
       COMMAND.COM file is modified?;
       RAM-resident programs are modified ?;
       Load RAM-resident programs into mem.;
       Load DOS boot loader into memory;
       Control over the DOS boot loader;
     }
   }
   else
   {/*Recover the destructed DOS boot regions*/
recv:  Copy DOS system files of Part.II to part. I;
       Rebooting;
   }
 }
```

3.3 The Management of Security Zone

After we divide the disk into two parts, we install a simple file system for the security zone and move some files or information there. Our installation program functions as follows;

① It puts a modified master boot loader into the master boot sector.

② It registers some files at the security zone such as IO.SYS, MSDOS.SYS, COMMAND.COM, installation program itself and so on, encrypts their content, and copies the encrypted file to the security zone. It can also add or delete some files at any time.

③ It calculates the check-sums for all registered files and writes them at the fixed area of the master boot sector.

4 Experiments

In experiments, we focus on whether or not the PC is booting safely. So, we have modified the existing master boot loader to check whether DOS's system files(IO.SYS, MSDOS.SYS, COMMAND.COM) are changed by virus programs or not. The modified master boot loader reads some information from security zone(partition II) and checks the changes of the DOS's system files. To do these, we first have partitioned the hard disk into two parts using the free software(what we call, FIPS[19]), and then, moved some security information to the security zone. Second, we have replaced the existing master boot loader with our modified master boot

loader to the first sector of hard disk. Third, we just add some strings to the DOS's system files. Lastly, when we have rebooted the PC system, we have found the master boot loader's message that system files have been changed by any other programs. Our system is coded by assembly language on MS-DOS environment. Up to now, we have implemented our system to the extent that booting related files are safe in booting time. Especially, our system is strong against boot viruses. More study is under implementation for the perfect system of anti-virus programs.

5 Conclusion

Scanning programs and monitoring programs are under development to cope with the increasing number of computer viruses. However, it is not guaranteed that these programs perform in a satisfactory manners because the program, which run first, controls the other program due to the absence of protection mechanism for DOS. We present a technique to start the PC booting safely and then could be run anti-virus programs based on a secure environment without modification of DOS. We modified the master boot loader for this system. Our system is strong against boot viruses. More study is under implementation for the perfect system of anti-virus programs with possible new developments because of this study.

References

[1] S.R.Lee, Power Hacking Techniques, PowerBook, 1995, in Korean.

[2] M.S.Park, Computer Viruses - Analysis, Production and Prevention -, Kihanje, 1992, in Korean.

[3] C.S.Ahn, Virus Analysis and Vaccine Production, Chungbosidae, 1995, in Korean.

[4] C.H.Yoo, PC System Programming, Chungbomunhwasa, 1993, in Korean.

[5] S.K.Han, IBM PC Description Dictionary, Gipmundang, 1991, in Korean.

[6] D. Russell & G.T. Gangemi Sr, Computer Security Basics, O'Reilly & Associates Inc, 1992.

[7] F.B.Cohen, Protection and Security on the Information Superhighway, John Wiley & Sons Inc, 1995.

[8] J. Hruska, Computer Viruses and Anti-Virus Warfare, Ellis Horwood, 1990.

[9] Morton Swimmer, "A Virus Intrusion Detection Expert System", Proceeding of the eicar Conference '95, Zuerich, Nov. 1995.

[10] Igor G. Muttik, "Viruses against Antivirus World", Proceeding of the eicar Conference '95, Zuerich, Nov. 1995.

[11] Fridrik Skulason, "The Evolution of Polymorphic Viruses", Proceeding of the eicar Conference '95, Zuerich, Nov. 1995.

[12] Igor Daniloff, "New Polymorphic Random Decoding Algorithm in Viruses", Proceeding of the eicar Conference '95, Zuerich, Nov. 1995.

[13] Dmitry O. Gryaznov, "Scanners of The Year 2000: Heuristics", Proceeding of the eicar Conference '95, Zuerich, Nov. 1995.

[14] Marko Helenius, "Automatic and Controlled Virus Code Execution

System", Proceeding of the eicar Conference '95, Zuerich, Nov. 1995.

[15] Pan Koo Kim, S.C.Kwon, B.M.Lee, "A new PC Starting Method to cope with Computer Viruses", Proceeding of the eicar Conference '96, Linz Austria, Nov. 1996.

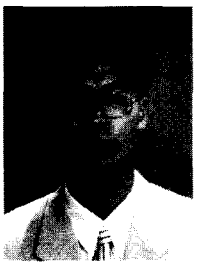
[16] T.Tachibana, K.Mochizuki, "Implementation of Anti-Viral Integrity Check System Based on Digital Signature", Proceeding of the eicar Conference '96, Linz Austria, Nov. 1996.

[17] Morton Swimmer, Jeff Kephart, "The shifting antivirus technology", Proceeding of the eicar Conference '96, Linz Austria, Nov. 1996.

[18] M. Lardschneider, "Virus Protection for Software Distribution in PC-Networks", Proceeding of the eicar Conference '96, Linz Austria, Nov. 1996.

[19] Free Software Foundation, "The First nondestructive Interactive Partition Splitting", WWW page at http://www.student.informatik. th-darmstadt.de/~schafer/fips.html.

□ 著者紹介



김 경 수

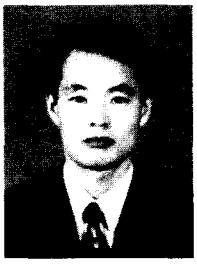
1992년 2월 조선대학교 컴퓨터공학과 학사
1994년 8월 중앙대학교 컴퓨터공학과 석사
1996년 9월 ~ 현재 조선대학교 전자계산학과 박사과정

※ 주관심 분야 : 컴퓨터시스템 보안, 컴퓨터바이러스, 영상처리, 컴퓨터 그래픽스

한 승 조

1980년 2월 조선대학교 전자공학과 학사
1982년 2월 조선대학교 대학원 전자공학과 석사
1994년 2월 충북대학교 대학원 전자계산학과 박사
1997년 현재 조선대학교 전자정보통신공학부 교수

※ 주관심 분야 : 통신보안, 음성합성, ASIC 설계



김 판 구

1988년 2월 조선대학교 컴퓨터공학과 학사
1990년 2월 서울대학교 컴퓨터공학과 석사
1994년 8월 서울대학교 컴퓨터공학과 박사
1995년 2월 ~ 현재 조선대학교 전자계산학과 조교수

※ 주관심 분야 : 운영체제, 컴퓨터시스템보안, 컴퓨터바이러스, 멀티미디어시스템 등