Journal of Computer Science 6 (4): 457-460, 2010
ISSN 1549-3636
© 2010 Science Publications

A Trust System Based on Multi Level Virus Detection

Yasmine H. Abdul-Amir
Department of Computer, Collage of Sciences, Al-Mustansiriya University, Iraq

Abstract: Problem statement: As these detection methods were developed and implemented, the virus developers adapted to the new detectors in ways intended to defeat them. **Approach:** This study introduced new multilevel virus detection (MDS). **Results:** This system model depended on an advance behavior blocking technology. It detected virus-code by a behavior approach monitors and determined a virus activity at several protection system levels. **Conclusion:** This system simultaneously provided smart memory resident monitor, integrity checker and activity virus file (.BAT) checker.

Key words: Computer security, system security, computer virus

INTRODUCTION

As the number of viruses grew, the old scanning methods had to include larger and larger signature databases and scanning became intolerably slow. Consequently, the developers began to streamline scanners. Instead of scanning the entire file, the entry point is examined for any pointers that would point to a virus if infected. Generic decryptions for encrypted viruses were developed and actions that reflect virus behavior, like writing to the boot block of a disk were trapped and examined (Shea, 2003).

Viruses generally have two phases: Infection and attack. When a virus is released it infect available programs and files, then depending on the virus, searches for other victims each time those programs and files are opened. Other viruses wait for a target before they become infectious. This could be anything, a date, a time or specific event like the deletion of an employee’s payroll record. The attack phase too often waits on a target, so a virus can inhabit the system for days, months, even years before it attacks. Then depending on its instructions, it may slow your computer down, change files name or incapacitate the system. The manner in which a virus spreads and what it does depend on the type of virus (Lin, 2008). For string and algorithmic scanning, virus makers inserted wild card instructions, like no operations, that had no function, but changed the length of the code and the signature. These were inserted in the virus byte string at random locations as the virus replicated itself.

This led anti-virus developers to introduce wild card scanners and heuristic rules to search for sub-strings typical of virus behavior. In this study multilevel virus detector was added such that the virus scanner calculated a checksum for every executable program and appended it to the file (i.e., vaccination).

MATERIALS AND METHODS

A computer virus is a small piece of programming created to inter computer systems and infect files. Like its counterpart in nature a computer virus infects healthy files in its host computer and then spreads its infection to other healthy computers. Typically a virus will replicate itself and try to infect as many files and systems as it can (Shea, 2003). As viruses have become more sophisticated, so have virus detection and eradication programs. Virus detection typically includes one, or more, of the following methods:

String searches: The earliest scanners relied on simple string scanning and pattern Recognition. All memory and secondary storage locations including disk boot records were scanned looking for specific bit sequences and/or string lengths that identified a unique block of data or program code specific to a particular virus. Scanners typically relied on known lengths to identify and remove malicious code (Singh and Singh, 2000).

Algorithmic searches: These scanners search for the presence of certain parameters or algorithmic constructs (e.g., control transfers, encryption, decryption algorithms) to detect the presence of a virus in an infected file.

Vaccination methods: These detectors record the characteristics of executable files and append a signature to the file. Each time the file is opened, the signature is checked against the known signature. If the file has been modified, a virus is suspected and further testing is invoked (Hamar and Run, 1998).

Investigation methods: A file can be examined for its capability to replicate and infect other files. In some cases, otherwise unknown viruses can be detected by this method, but it is not fully reliable.

Anti-stealth methods: Often called the sandbox method, the detector observes the behavior of a file invoked for execution in a simulated environment in order to detect behavior that is characteristic of a virus. If the code passes the test it is released to the real (non-simulated) execution environment, otherwise it is further analyzed (Petru and Atkins, 1997).

The modern scanner has changed significantly over time with several key capabilities common to most scanners today support:

- Memory residence: Scanners are loaded in memory and examine every file before it is loaded for execution
- Virus profiling: Establishes a rule base specific to each known virus as well as for known mutation and polymorphism virus engines. The rules are tested for every known virus while running the suspect file in a simulator (see heuristic-based below)
- On-line virus profile updates: The use of virus profiles means that updates to the profile database are required for each new virus. Today, profiles are maintained on the anti-virus vendor’s web site and can automatically update the profile stored on the user's system so that the most recent viruses are detectable. In order to get updates, the typical home user must visit the vendor’s web site. Many corporations download the updates automatically and distribute them to their users over the internal corporate network (Stere, 2006; CYBEROFT, 2005)
- Signature (string and/or algorithmic) scanning. The virus detection engine scans memory and all attached disks for virus signatures against the profile database.
- Heuristic-based generic scanning (also known as simulation): Some scanners implement system emulation such that executable code is presented to the emulator that executes the code in a virtual machine rather than the real machine. During emulation an encrypted virus would decrypt itself and attempt to execute. Since the emulator can intercept the decrypted code, the virus and/or fragments of the virus, can be recognized by the signature scanner (CYBEROFT, 2005)

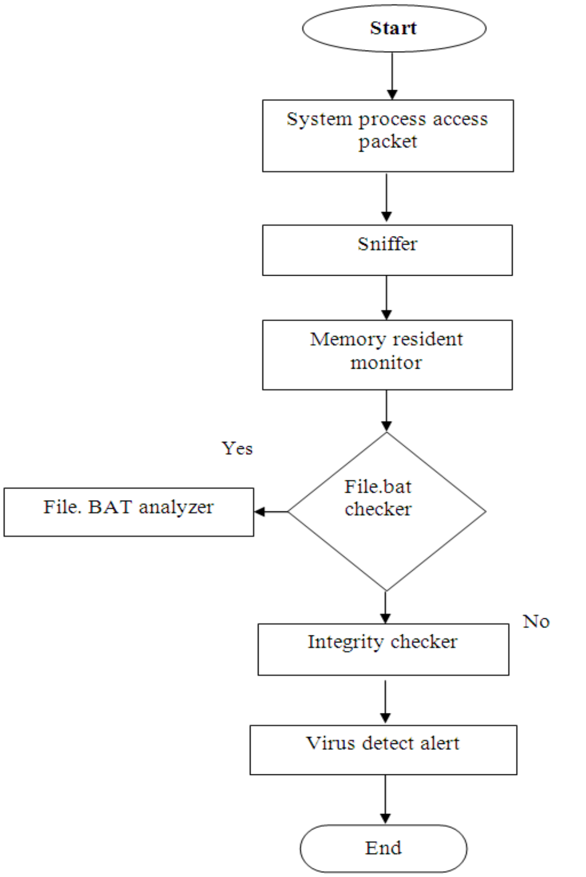


Fig. 1: Startup of (MDS) multi level virus detection system

The proposed model: This research concentrates on the aspect of making the complex networks more vulnerable to various kings of complex virus attacks. This work discusses the application of multi level virus detection system on a practical system and Fig. 1 shows the structure of proposed system. Therefore, a virus detection system is suggested that integrates automatic virus techniques which detect concerted scan activities and derive possible signatures of virus.

RESULTS

The system implemented by based on using the proposed system. We present and explain works of the windows of the system and the relationship between them. There are more than window to display the system as follows: when the computer is start up the system is start and this windows is appears to the user as shown in Fig. 2. When the user opens the file and click skip icon from Fig. 3.

Figure 4 shows if the file is infected and warring the user "not allowed" to open this file.

When the user clicks skip, skip the system, delete the virus and delete the entire virus like this. Figure 5 shows the window upper when the user click the report icon.

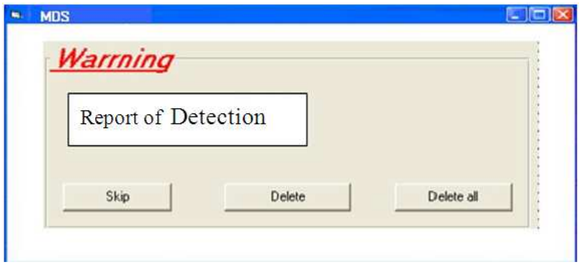
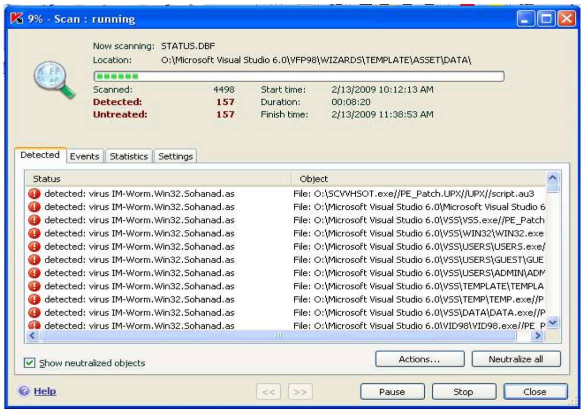


Fig. 2 the MDS s/w



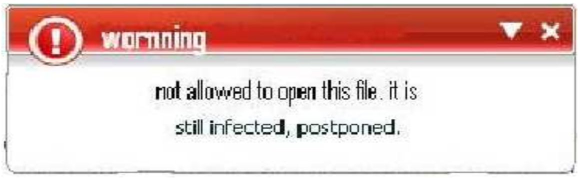
```
9% - Scan : running

Now scanning:  STATUS.DBF
Location:      O:\Microsoft Visual Studio 6.0\VFP98\WIZARDS\TEMPLATE\ASSET\DATA\

Scanned:        4498     Start time:    2/13/2009 10:12:13 AM
Detected:        157     Duration:      00:08:20
Untreated:       157     Finish time:   2/13/2009 11:38:53 AM

Status                                   Object
detected: virus IM-Worm.Win32.Sohanad.as File: O:\SCVVHSOT.exe//PE_Patch.UPX//UPX//script.au3
detected: virus IM-Worm.Win32.Sohanad.as File: O:\Microsoft Visual Studio 6.0\Microsoft Visual Studio 6
detected: virus IM-Worm.Win32.Sohanad.as File: O:\Microsoft Visual Studio 6.0\VSS\VSS.exe//PE_Patch
detected: virus IM-Worm.Win32.Sohanad.as File: O:\Microsoft Visual Studio 6.0\VSS\WIN32\WIN32.exe
detected: virus IM-Worm.Win32.Sohanad.as File: O:\Microsoft Visual Studio 6.0\VSS\USERS\USERS.exe/
detected: virus IM-Worm.Win32.Sohanad.as File: O:\Microsoft Visual Studio 6.0\VSS\USERS\GUEST\GUE
detected: virus IM-Worm.Win32.Sohanad.as File: O:\Microsoft Visual Studio 6.0\VSS\USERS\ADMIN\ADM
detected: virus IM-Worm.Win32.Sohanad.as File: O:\Microsoft Visual Studio 6.0\VSS\TEMPLATE\TEMPLA
detected: virus IM-Worm.Win32.Sohanad.as File: O:\Microsoft Visual Studio 6.0\VSS\TEMP\TEMP.exe//P
detected: virus IM-Worm.Win32.Sohanad.as File: O:\Microsoft Visual Studio 6.0\VSS\DATA\DATA.exe//P
detected: virus IM-Worm.Win32.Sohanad.as File: O:\Microsoft Visual Studio 6.0\VID98\VID98.exe//PE_P
```

Fig. 3: User opens the file



not allowed to open this file. it is still infected, postponed.

Fig. 4: Warning

On the other hand the system call when the user calls it by right clicks optioned as shown in Fig. 5.

DISCUSSION

Network Sniffers: Networks Sniffers are tools that simply collect data from a live network. They generally include a means of storing the information in a particular format on disk and a means of viewing or browsing the captured network packets as shown in Table 1.

File viruses: The memory resident module monitors the system for specifically viral behavior in executable files. Viral behavior includes: These three activities are initiated simultaneously when a virus activates:

- Modifying the code of an executable file
- Attaching additional code to an executable file
- Executing the attached code

Smart memory-resident monitoring is active whenever the computer is running and operates in the background. When file modification activity is detected, the MDS detects the modification and stops it, displaying an alert to the user. If the user chooses to

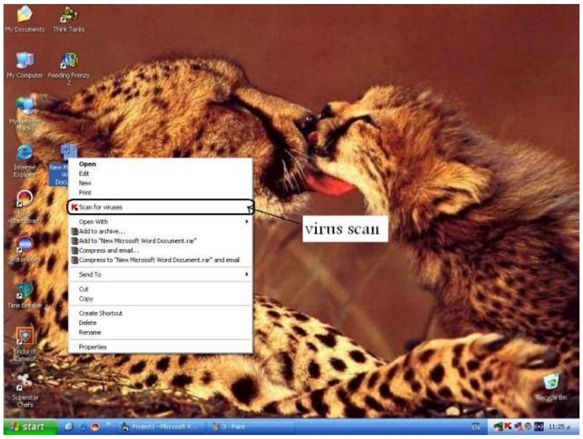


Fig. 5: Virus detection right click

Table 1: Network sniffers

Product	License	Description
Tcpdump	Open source	Tcpdump is a tool to print out the headers of packets network interface that match a Boolean expression. It can also save the packet data to a file for later analysis and/or read from a saved packet file.
Ethereal	Open Source	Ethereal is a multi platform network protocol analyzer that allows users to browse captured traffic. Ethereal includes sophisticated filters and can dissect many protocols.
Cflowd caida.org	Open source	Cflowd is traffic analysis tool to collect data from Cisco's net flow export feature. The product guide lists its uses as trends analysis, characterization of workloads usage tracking, accounting and network monitoring.
winPcap	Open source	winPcap is a tool for packet capture and network analysis for the Win32 platforms. winPcap adds to windows the ability to capture and send raw data from a network card, with the possibility to filter and buffer the captured packets. winPcap provides an API that exports a set of high level capture primitives that are compatibility with libpcap, the popular Unix capture library.

deny the operation, the offending process is killed and the user is promoted and must decide whether or not to remove the offending program. If the user decides to allow the modification, the process is allowed to continue and the event is logged. If MDS detects an operation that modifies a non-critical area of an executable file-and nothing else- it takes no action because the activity does not meet the critical area of an executable file- and nothing else- it takes no action because the activity does not meet the critical for being viral.

After that level of MDS is check level by checking the file type.

Is the file (.BAT)? The analyzer try to find the viruses by searching for characteristics of virus often have infect objects. It searches for way viruses getting in to the Notepad analyzer uses this indications:

```
Cd%winDir%\system|
Dettree/y*_dll
Cd\deltree/y*.sys
```

This code .BAT that mean create Batch file and create virus will delete all (dll) files and system files “deltree”. This command special use to delete all files and after that this code deletes all file has extension-dll and sys.

Basically the code will start in C and go to all windows file according to this command: (cd% w/sys) And this code can convert to delete all extension files like:

Y*.jpg .... photo files.
*. Mp3…….. musical files
*.ast ….. and so on.

Integrity checker/vaccinator: During installation, integrity checker/vaccinator takes a snap-shot of all system and executable file types and stores the information in a small data file in each of the directories where those file type are found. The vaccination files average around 100 bytes from the beginning to the ending of the file. When an executable is accessed or run, before closing the file, the Real-Time monitor checks the active file information against the information for that file stored in the vaccination file. If a modification is detected in a section of the file where viruses tend to infect, a backup of the unmodified state is made and the user is alerted and prompted to accept or deny the change. If the user denies the change, the file reverts back to its original state.

CONCLUSION

In this study modern methods for detection virus it by install the multilevel of virus detection process was introduce. Some virus code may use more than one virus mechanism. It is not enough searching in the few bytes from the beginning of the documents for suspicious instructions.

REFERENCES

CYBEROFT, 2005. Protection against Trojans horses cyber city. http://www.cyber.com

Hamar, S. and K. Run, 1998. A professional virus detection and elimination internet worm. http://www.viruslist.com.

Lin, J., 2008. On malicious software classification. Proceeding of the International Symposium on Intelligent Information Technology Application Workshops, Dec. 21-22, IEEE Xplore Press, Shanghai, pp: 368-371. DOI: 10.1109/IITA.Workshops.2008.106

Petru, T. and D. Atkins, 1997. Internet Security Professional References. 2nd Edn., New Riders Publishing, ISBN: 10: 156205760X, pp: 916.

Shea, J.J., 2003. Hacking exposed-network security secrets and solutions. Elect. Insulat. Mag., 19: 73-74. DOI: 10.1109/MEI.2003.1238725

Singh, M. and S. Singh, 2000. Network security (security in large networks). Proceedings of the 25th Annual IEEE Conference on Local Computer Networks, IEEE Xplore Press, USA., pp: 88-93. DOI: 10.1109/LCN.2000.891012

Stere, D.C., 2006. An undetectable of virus. http://www.researchibm.com