A New Methodology based on Cloud Computing for Efficient Virus Detection

Vasileios A. Memos and Kostas E. Psannis, *Member, IEEE*

***Abstract*—Antivirus software programs use specific techniques to detect computer viruses, malware and other network threats. The basic, most common and oldest antivirus detection technique is "virus signature scanning", whereby antivirus programs use unique byte sequences for each virus so as to identify potential presence of malicious code in each file investigation procedure. Despite its advantages, this technique has many weaknesses that are highlighted in this paper. In lieu, this paper proposes a new hybrid security model for optimized protection and better virus detection, which merges the "Sandboxing Method", "System-Changes-based Signatures" and "Cloud Computing".**

***Index Terms*—antivirus techniques evaluation, cloud technology, sandboxing method, system-changes-based signatures.**

I. Introduction

TODAY viruses and other malicious software – malware – have increased dramatically and spread rapidly every day.

In addition, new unknown malware, known as zero-day threats, appear day by day and in combination with advanced virus concealment methods that are used by sophisticated virus programmers make the work of antivirus software very difficult. The current protection methods which antivirus vendors use are not adequate to solve these problems and produce many false positives. So, they should develop new methods and techniques for their software for more efficient malware detection. The main reason for these antivirus weaknesses is that they are based on "virus signatures" which consist of specific byte sequences to identify malicious code [1].

This problem urged us to make this research to prove the main problems of this technique and suggest a new security model that will not be based on specific byte sequences, but on signatures of system changes which malicious processes cause, in combination with some innovative techniques, such as sandboxing [2] and cloud technology [3] which have been used over the last years in many antivirus programs. In this research, we use a malicious file – Trojan horse type – which is transmitted via social networks, such as Facebook, to make a series of tests to prove the above antivirus weaknesses. We also use a file splitter to divide the Trojan to smaller files, an antivirus program for scans, the windows Command Line and a clear system file. The paper is organized as follows: In Section 2, we cite all the related work about the problem of signature scanning detection method. In Section 3, we extensively analyze the problems of this method. In Section 4, we prove these problems with a series of tests. Section 5 includes our proposed methodology for better virus detection and more efficient protection against network attacks. Section 6 concludes the paper.

This work was supported in part by the Research Committee of the University of Macedonia, Greece, under grant 80749 for the advance of Basic Research.

V. A. Memos is with the Department of Technology Management, School of Information Sciences, University of Macedonia, Greece; e-mails: tm0844@uom.gr; mvasileios89@gmail.com)

K. E. Psannis is with the Department of Applied Informatics, School of Information Sciences, University of Macedonia, Greece; (phone/fax: 302310891737; e-mails: kpsannis@uom.gr ; mobility2net@gmail.com)

II. Related Work

Nowadays antivirus programs are quite advanced and identify viruses by using various static and dynamic techniques, but these techniques are not sufficient. Virus programmers are usually a step ahead of antivirus programs, because they use many techniques, such as space filling, compressing and encryption, to make their viruses avoid the detection by antivirus scanners. In addition, new viruses appear every day, which current antivirus programs cannot identify immediately, because of the fact that they are mainly based on virus detection by using string signatures [4, 5]. In 2006, Seon Yoo and Ulrich Ultes-Nitsche presented a non-signature-based virus detection approach, using Self-Organizing Maps (SOMs). In 2008, Essam Al Daoud, Iqbal H. Jebril and Belal Zaqaibeh indicate the need of a new model that could be able to detect all metamorphic virus variants by using new trusty monitoring techniques by attaching a digital signature and certificate to each new program [6]. Unlike the traditional detection methods, this SOM-based approach would detect infected files without virus signatures. In their tests on 790 different infected files (including polymorphic and encrypted viruses), their model detected 84% of them, with a false positive of 30% [7]. In addition, in 2009, Min Feng and Rajiv Gupta, developed a new algorithm for matching a new type of virus signatures - dynamic signatures - which was based on the run-time behavior of a piece of malware. Tests proved that this algorithm was very effective in recognizing different variants in a malware family with a single dynamic signature and without false positives [8]. In 2011, Madhu K. Shankarapani, Subbu Ramamoorthy, Ram S. Movva and Srinivas Mukkamala developed algorithms which could detect known malware without the use of signatures [9].

III. PROBLEM DEFINITION

Antivirus vendors have developed many methods to improve the protection that their software provides to users. The most common and oldest technique is "virus signature scanning" that antivirus software uses unique byte sequences as a signature for each virus to recognize it. The following Section includes detailed description of this technique.

This signature-based detection model has many advantages, such as the low memory and system resources that it needs and the speed of scanning lots of files per second [10], but it has inevitably many weaknesses, too. It is dangerously easy for sophisticated hackers to change a virus signature and make viruses undetectable by the antivirus programs. In addition, virus signatures dramatically increase the false positive rate of antivirus scanners [11]. False positives can also cause major problems to the system operation. This happens because new viruses and malware are discovered daily, so there are lots of signatures and, in consideration of the amount of files that exist, a signature is often difficult to be absolutely unique.

In addition, antivirus virus database should be constantly updated to remain reliable [12]. However, new viruses are discovered per second [13], "0-days threats" as they are known, so virus identification should be based on other detection techniques too, which would not examine the binary code of the files to identify specific sequences of bytes, but would try to guess the behavior of each file. One such method is "heuristics" which is used additionally by many antivirus programs and are based on looking up for suspicious instruction sequences that may be related to malware existence. Although this method provides better detection capabilities, it gives a lot of false positives when it is adjusted for maximum detection rate [14, 15]. And of course, such behavior detection methods take up many system resources, so they slow down computers' operation and overload networks.

Virus programmers are usually one step ahead of antivirus programs, because of the fact that they first launch malware to attack their targets and the only thing that antivirus vendors can do is to find and defend against these attacks by finding their antidote to launch it as virus definition file update to protect users' computers. Note that often a considerably long time is needed to manage it.

In addition, hackers use many mutation and encryption methods to conceal their viruses and bypass antivirus scanners, such as Stealth technique [16], Self Modification technique [17], Virus Encryption technique (with a variable key) [9], Polymorphic technique [18], Metamorphic technique [18] and Avoiding Bait-Files Technique [19].

All the above problems are considerable vulnerabilities of antivirus software, so new techniques should be discovered to deal with them for better detection rate and lower false positives frequency.

IV. EXPERIMENTS

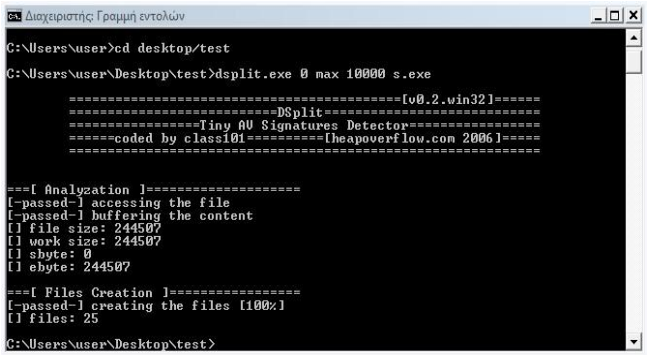
This Section includes three tests that present the main weaknesses of virus signature method based on string scanning by using a known methodology which describes the way antivirus programs operate. This section is organized as follows: In Subsection A we use some tools to find the virus signature of a malicious file. In subsection B, we present a test to show the problem of false positives that are caused by using this method. Subsection C includes another test which indicates the problem of virus concealment.

A. Finding virus signature

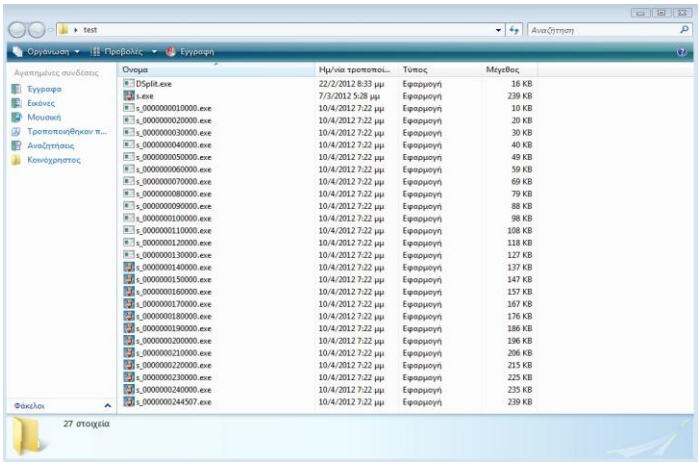
In this test we study and test the malicious file *"s"*, 238KB size, which looks like a picture, but it is an executable file (*s.exe*) which is proved by clearing the "Hide extensions for known file types" check box in the windows folder options control panel. It is actually a Trojan horse and it is transmitted via social networks, such as Facebook. In this test we try to find its virus signature that has been used by an antivirus scanner to identify this threat. Note that each antivirus program uses a different signature to identify a certain threat.

For this test we used four tools: the Windows Command Line, DSplit file splitter, HxD hex-editor program and Avira antivirus as an on-demand scanner for our scans.

Firstly, we create a "test" folder on the computer desktop and put into "s.exe" and "dsplit.exe" files. We type *cd desktop/test* to gain access to the created folder and then give: *dsplit.exe 0 max 10000 s.exe*, as it is depicted in figure 1(a). This command divides *s.exe* file to smaller ones so that each file is 10,000 bytes larger than its previous one and 10,000 bytes smaller than its next one. Figure 1(b) depicts the result of the above command with the 25 new created files.



(a)



(b)

Fig.1 The initial dsplit command **(a)** which divides the s.exe file in 25 new files **(b)**.

We now scan the 25 new created files with Avira antivirus and as it is depicted in figure 2, Avira detects as "TR/Offend.kdv.49932" the last 16 files, from the 100KB file (*s_0000000100000.exe*) to the final one. This actually means that some of the added bytes into the 90KB file make the 100KB one to be detected as malware. To find which of the 100 added KB make the difference, we delete all the created files except these two files.

The new command now is: *dsplit.exe 90000 100000 1000 s.exe*, which creates new files between 90KB and 100KB ones, so that each of them is 1,000 bytes larger than its previous one and 1,000 bytes smaller than its next one. We scan these new files with Avira and observe that 9/11 files detected as Trojan horse, from 92KB file onwards. It means that there is something in 92KB file that is not in the previous 91KB one.

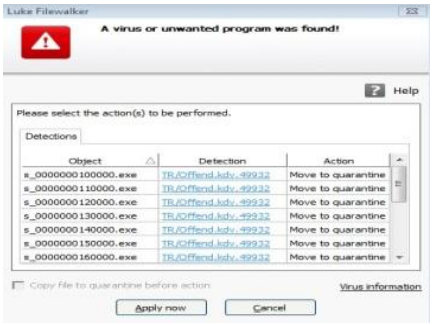


Fig. 2 Avira antivirus scanner detects 16/25 files, from the 100KB file and all files after it.

The process is a routine and as previously we delete all other files and hold these two files and type the proper command, which is now: *dsplit.exe 91000 92000 100 s.exe*. By repeating the above steps, it will be needed to give the commands *dsplit.exe 91000 91100 10 s.exe* and *dsplit.exe 91040 91050 1 s.exe* too. Thus, we will manage to have files with only 1 byte difference between each other. Figure 3 depicts this final step of this loop, where there are 11 created files. By scanning these files again with Avira antivirus, we see only one file detected as Trojan, the 91050B one, as it is depicted in figure 4. It means that it has one more byte than the previous 91049B file, which causes the detection by the antivirus scanner.

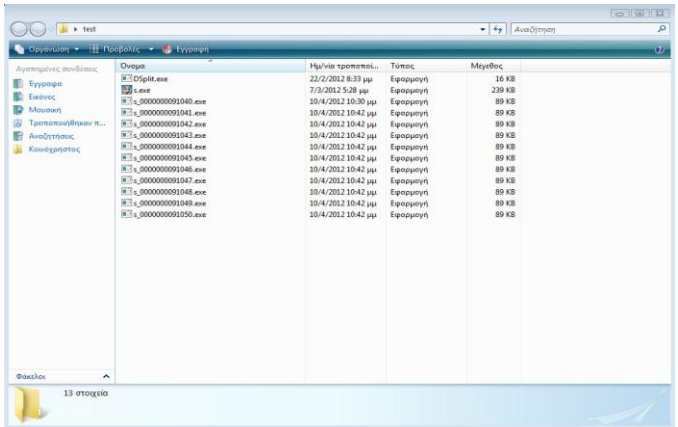


Fig. 3 All files have only 1B difference between each other.

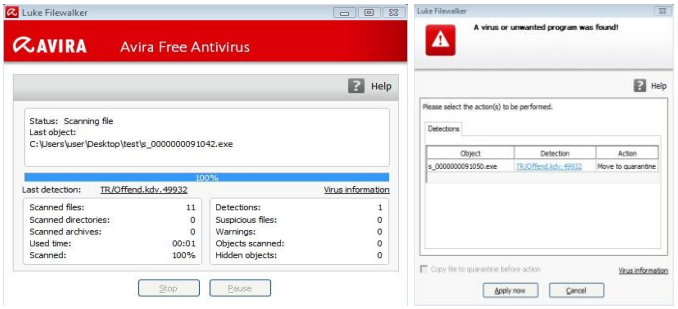


Fig. 4 Avira detects only one of the files.

We now use HxD hex-editor to open these two files and compare their hexadecimal codes, as it is depicted in figure 5.

We note at the end of the codes that this byte which is added and causes the Trojan detection is the hexadecimal *41* or the string *A*. So, we just find the end of the string signature which Avira antivirus use to identify this Trojan horse. To find the start of the signature we need a clear – non-malicious – file. We use *taskmgr.exe* system file, which is the executable part of the Windows Task Manager.

We open *taskmgr.exe* file with HxD, too, and try copying some of the last bytes from the 91050B file into it. Then, we scan it with Avira antivirus. It is a routine again by adding byte-to-byte from the detected file to this clear one, until we can manage to make this clear file detectable as a Trojan, too.

A virus signature must be long enough to generate as fewest false positives as possible. So, choosing a short signature, such as: *55 8B EC 6A FF*, is not a good solution. This is because of the fact that such short byte sequence is possible to be present in many other files that are not malicious. That is why the length should be considerably long, more than 50 bytes [20]. By continuous tests we find and depict in figure 6 that Avira uses in this case a 123-byte length signature to detect the *s.exe* file as "TR/Offend.kdv.49932", which is consisted of the following:

55 8B EC 6A FF 68 90 70 41 00 68 B0 64 41 00 64 A1 00 00 00 00 50 64 89 25 00 00 00 00 83 EC 68 53 56 57 89 65 E8 33 DB 89 5D FC 6A 02 FF 15 30 70 41 00 59 83 0D C4 89 41 00 FF 83 0D C8 89 41 00 FF FF 15 34 70 41 00 8B 0D C0 89 41 00 89 08 FF 15 38 70 41 00 8B 0D BC 89 41 00 89 08 A1 5C 70 41 00 8B 00 A3 CC 89 41 00 E8 11 01 00 00 39 1D 90 89 41 00 75 0C 68 AC 64 41

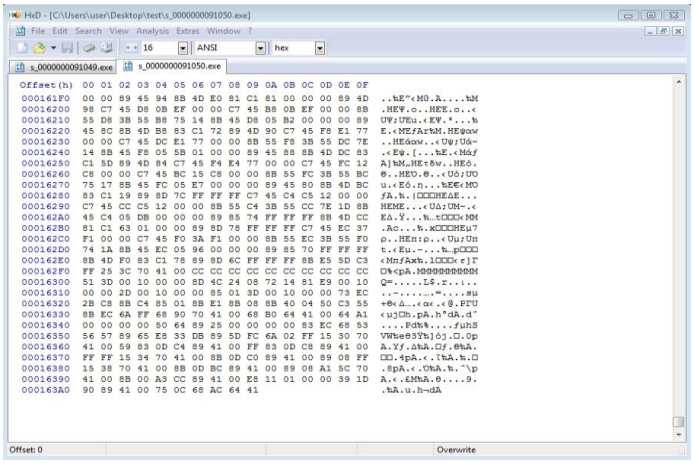
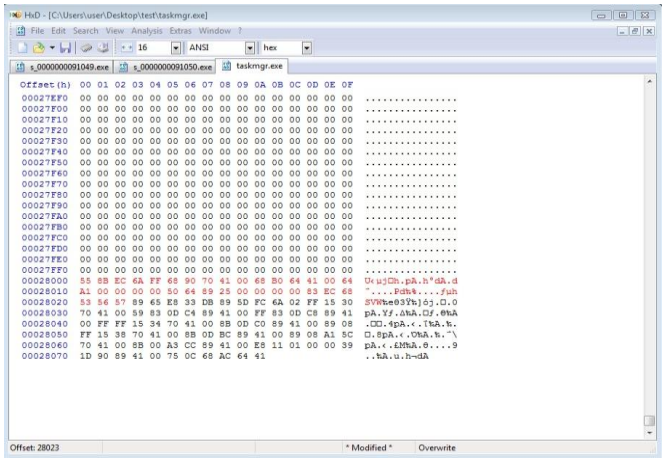
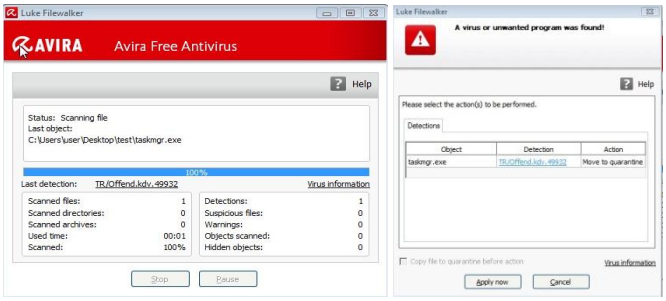


Fig. 5 HxD editor with hexadecimal code of '91049' file and '91050' one.



(a)



(b)

Fig. 6 The signature that is added in clear file taskmgr.exe **(a)** and makes it detected falsely as Trojan by Avira **(b)**.

B. The problem of false positives

In the previous subsection, in our try to find the virus signature of the malicious file, we make the clear *taskmgr.exe* file to be detected by the antivirus as a Trojan. But is this file actually "TR/Offend.kdv.49932" as Avira says? The answer is no. The 123 additional bytes in the original code of the *taskmgr.exe* are no more than a useless code and it neither affect the operation of the task manager file, nor makes it malicious. This conclusion can be verified by uploading the modified *taskmgr.exe* file to VirusTotal site, which includes 42 online antivirus scanners. As it is depicted in figure 7, only Avira finds this file as malicious. It is a false positive by Avira, because of the 123B sequence existence inside the *taskmgr.exe* code. Same false positives we would also have if we made our tests with other antivirus scanners.

The above problem is more important than it seems, because of the fact that Avira and each other antivirus often recommends user to set this file to "quarantine" by isolating and making it inactive or to "delete" it, when it is unable to "repair" it. But system files are important and necessary for the proper system operation, and therefore this action will cause inconstancy to the system, making it unbootable or crash.

C. Virus concealment

In this subsection, we present a simple way to conceal a virus from detection by antivirus programs. To manage it, we need to change properly the bytes sequence which antivirus use as virus signature to identify - in our example - *s.exe* file as Trojan horse variant, without change its malicious operation.

Figure 8 (a) depicts the highlighting of the signature we found in the previous section in the original *s.exe* malicious file.

By changing properly an alphabet string, such as "U" to "u" (55 to 75 in hexadecimal code), signature will change, but not the file's destructive function. If we scan the new changed s.exe file now, we will observe that Avira does not detect it as Trojan (Fig. 8(b)).

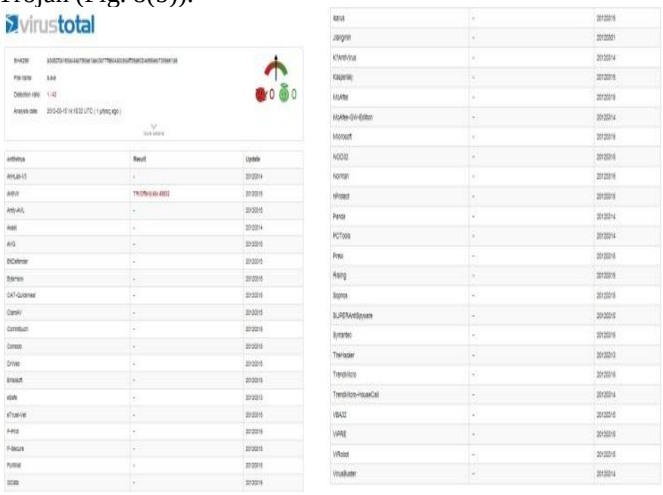
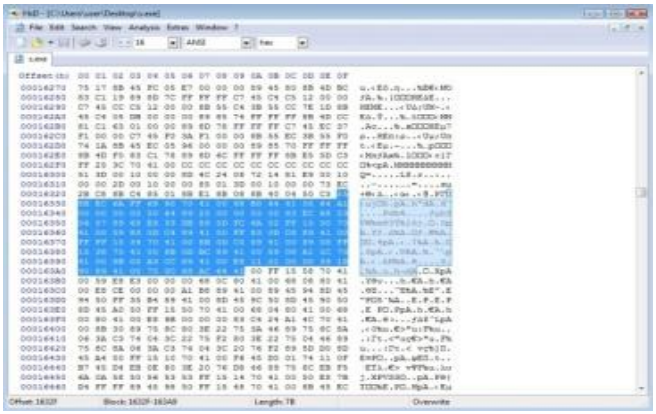


Fig. 7 Detection results of 42 antivirus by uploading the new taskmgr.exe file in VirusTotal site.



(a)

(b)

Fig. 8 The finding and change of the Trojan signature (a) and the scan process and scan results by Avira **(**b**)**.

Note that this process is only a simple, but important example to indicate how easily a virus can be concealed from an antivirus scanner. The problem is much more serious if we

consider that there are polymorphic viruses too, whose code is self-replicated and self-concealed. In addition, there are many specialized programs – crypters – that use various complicated encryption methods to make a virus file non-detectable, easily and quickly. These tools are programmed and released by hackers over the Internet.

V. PROPOSED APPROACH – METHODOLOGY

A. Model description

As we describe and analyze with experiments in the previous Sections, the already existing methods which antivirus software use, have many weaknesses. For optimized virus protection, we propose the development of a new hybrid security model that will merge "Cloud Computing" and "Sandboxing Technology" – methods which are already implemented gradually in some antivirus programs – with a new creation methodology of Virus Signatures that will be based on "System-Changes" which a malware causes to the infected computer.

This new cloud-based antivirus software will be used as a client and will monitor in real-time the activity of computer files in the cloud, such as any changes they cause to hard disk, system directory, registry etc, and if it recognizes malicious changes, they will not become permanent, thanks to the sandbox. If the changes are not clearly malicious, the antivirus will alert the user to decide whether the suspicious file will continue to run sandboxed or outside the sandbox feature. For a better performance, cloud technology would come in handy to contribute again, as the user will have the access to see in real-time collecting statistics, marks and comments by other users connected to the cloud for the suspicious file activity.

B. General description of the used methods

Sandboxing technology: A sandbox provides an additional protection level against infection by harmful viruses. It is an entirely isolated environment which the programs that are running inside it have not access to the local computer drivers, so there is no risk for the system files. So, if the user runs a file that is malicious, the changes it will cause to the system will not be permanent. In this case, the installed antivirus program blocks the malicious process of the file and the system resets to its previous mode [2].

System-changes-based virus signatures: As we proved in the previous section, the "string-based" signature detection method has many weaknesses. Signatures change dangerously easily, making viruses undetectable and also this method has many false positives. In addition, new unknown malware are discovered, so it is necessary to change the way that signatures are created. This new method will not be based on specific byte strings, but on changes which malicious files make on the system computer. This technique is similar to heuristics; the main difference, however, is that each malicious behavior will have its different signature. So, in this new method the antivirus scanner will not look up for specific "bytes sequences", but for specific "system-changes sequences" to recognize viruses.

Cloud Computing technology: Cloud computing is a new approach for quicker response to new threats, which spread constantly on the internet. This architecture is based on communication between servers that are somewhere on the Internet –"cloud" – and computers that are connected to this "cloud". The connected computers have installed a small program - in our case the antivirus program - that is used as a client. Most processes of the program take part in the connected server by the web service, which is running in the cloud. Thus, the computer does not need to process and store a large amount of data – in our case, virus signatures. At regular intervals, the client automatically scans the computer for virus existence, using the information it takes from the web service's database of the cloud server that it's connected to. In addition, because of the fact that web service runs in the cloud and not locally in the computer, cloud antivirus programs consume low memory and system resources, without supercharge the computer even if it is not meet the minimum requirements that have the most current computers [21].

By using cloud technology, antivirus analysts can collect easily and quickly all the suspicious processes from users' computers to analyze them. So, if they find malicious process, they launch the suitable antidote signature in the cloud, so automatically within minutes, all the computers that are connected to it, are protected by this threat, without the need of downloading by the antivirus program often in a day large maybe patches to stay protected from the latest malware that spreads rapidly via the Internet all over the world [21].

Generally, the cloud method requires a 24/7 Internet connection, but it is not indispensably needed. Cloud antivirus always keeps a cache of malware information on the local computer for offline access, too. This cache doesn't include all the virus database of the cloud server, but only the basic and most common threat signatures, and it is updated every time it finds an Internet connection [21].

C. Proposed model analysis

As it is depicted in Figure 9, our proposed model is a hybrid security system that will consist of:

m cloud virtual sub-servers that constitute the Home Cloud Server

n terminals connected to *m* cloud virtual sub-servers

i sandboxes contained in *n* terminals

k files inserted into *n* terminals

l virus signatures that are collected and contained to the Home Cloud Server's Virus Database

l virus signatures that are transmitted from the Home Cloud Server's Virus Database to *m* cloud virtual sub-servers' Virus Databases, where: i=1,2.., n=1,2.., m=1,2.., k=1,2.., l=1,2.., $i=n$.

Example: In the Home Cloud Server Virus Database, there have been collected all the present *l virus signatures* from the specialized analysts who work in the cloud. *L virus signatures* transmitted from Home Cloud Server Virus Database to *sub-server 1* (virus database) and each other sub-server, connected to Home Cloud Server. So, *sub-server 1* database has *l virus signatures. File 1* is inserted to *terminal 1* (e.g. by

downloading). Firstly, the file is running in the terminal's 1 sandbox – *sandbox 1* – in the background, while antivirus (client) of *terminal 1* communicate with a cloud virtual sub-server of the Home Cloud Server, *sub-server 1*.

.

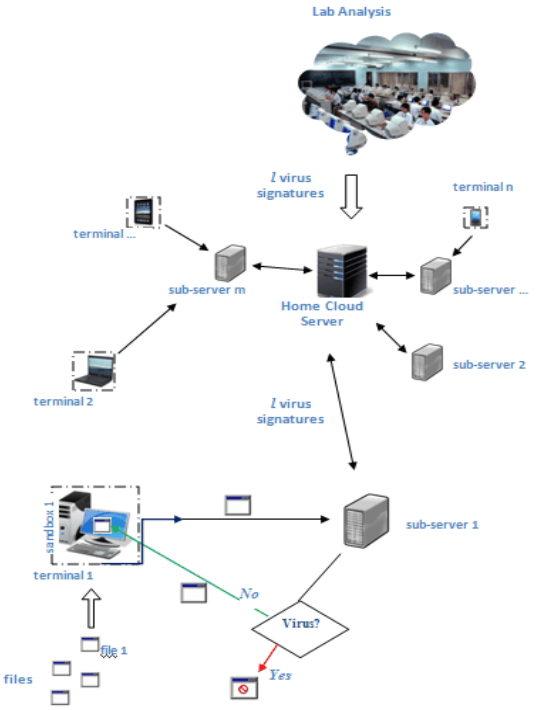


Fig. 9 The proposed model

A deep analysis takes part in the cloud *sub-server 1* to ascertain if the changes that cause the *file 1* inside the *sandbox 1* of the *terminal 1* match to any signature of the *l virus signatures* contained in the *sub-server 1* virus database. If they match to a virus signature, the process of the *file 1* is automatically stopped and the file is blocked. If they do not match to any signature, the *file 1* continues to run, but outside the *sandbox 1* therefore now.

VI. Conclusion

In this paper, we have analyzed the problems of traditional antivirus software which are mainly based on the string signature detection technique. This method presents major problems, such as many false positives, and thus constitutes an easy method for specialized hackers to fool and bypass. The proposed method merges System-Changes-based Virus Signatures, Cloud Computing and Sandboxing techniques. This new hybrid security model will not be based on string signatures - specific byte sequences - for virus identification, but on specific system-changes sequences that will be made by malicious processes to the computer, in combination with the two innovative technologies, cloud technology and sandboxing method. The former is applicable for faster detection of new, unknown types of malware and lighter antivirus software, whereas the latter is aimed at offering enhanced computer protection by running malicious processes in an isolated virtual environment until the their functions are verified. Future work will include deep a study and analysis of the capability to adjust to the new security model in four areas: a. capability of creation signatures based on specific system-changes, b. capability of improvement the response time of sandbox mode for action in collaboration with the cloud server, c. the user's data that will be collected for analysis by the cloud server, d. how cloud technology architecture could be redesigned to eliminate the high false positive rate problem as far as possible. This is also a prospective research direction.

References

[1] Alisa Shevchenko, *"Malicious Code Detection Technologies"*, Kaspersky Lab, 2008.

[2] Neamtu Iosif Mircea, *"Software Tools to Detect Files"*, Dept. Of Informatics, Faculty of Science, Lucian Blaga University of Sibiu, Sibiu, 2011.

[3] Ionut Ilascu, *"The Insides of Panda Cloud Antivirus"*, May 2009.

[4] P. Szor, "*The Art of Computer Virus Research and Defense*", Addison-Wesley Professional, Boston, MA (2005).

[5] E. Filiol, "*Computer Viruses: from theory to applications*", Springer-Velag France 2005.

[6] Essam Al Daoud, Iqbal H. Jebril and Belal Zaqaibeh, *"Computer Virus Strategies and Detection Methods"*, Int. J. Open Problems Compt. Math., Vol. 1, No. 2, September 2008.

[7] In Seon Yoo and Ulrich Ultes-Nitsche, *"Non-signature based virus detection Towards establishing a unknown virus detection technique using SOM"*, Journal in Computer Virology, 2006, Volume 2, Number 3, Pages 163-186.

[8] Min Feng and Rajiv Gupta, *"Detecting Virus Mutations Via Dynamic Matching"*, CSE Dept., University of California, Riverside, IEEE International Conference on Software Maintenance, September 2009.

[9] Madhu K. Shankarapani, Subbu Ramamoorthy, Ram S. Movva, Srinivas Mukkamala, "*Malware detection using assembly and API call sequences*", Journal in Computer Virology, Vol. 7, Issue 2, pp 107-119, May 2011.

[10] Sunita Kanaujiya, Dr. S. P. Tripathi, N. C. Sharma, *"Improving Speed of the Signature Scanner using BMH Algorithm"*, Vol. 11, No. 4, International Journal of Computer Applications (0975-8887), December 2010.

[11] Umakant Mishra, *"Overcoming limitations of Signature scanning – Applying TRIZ to Improve Anti-Virus Programs"*, TRIZsite Journal, April 2007.

[12] Babak Bashari Rad, Maslin Masrom and Suhaimi Ibrahim, *"Evolution of Computer Virus Concealment and Anti-Virus Techniques: A Short Survey"*, IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 1, January 2011.

[13] Liam Tung, *"Anti-virus can't keep up with threat onslaught"*, April 2012.

[14] Umakant Mishra, *"Eliminating False Positives in Virus Scanning"*, Bangalore, India, 2013.

[15] Randy Abrams, *"Understanding Heuristics"*, AVAR Conference, Seoul, 2007.

[16] Margaret Rouse, *"Stealth Virus"*, SearchSecurity TechTarget, September 2005.

[17] Bertrand Anckaert, Matias Madou, Koen De Bosschere, *"A Model for Self-Modifying Code*", Electronics and Information Systems Dept, Ghent University, Ghent, 2006.

[18] Carey Nachenberg, *"Computer Virus-Coevolution"*, Communications of the ACM, Vol. 40, No. 1, January 1997.

[19] Evgenios Konstantinou, Stefen Wolthusen, *"Metamorphic Virus: Analysis and Detection"*, University of London, TechTarget, 2008.

[20] Sam Rash, Dan Gusfield, *"String Barcoding – Uncovering Optimal Virus Signatures"*, University of California, Davis, 2002.

[21] Stephanie Crawford, *"How a Cloud Antivirus Works"*, Computer HowStuffWorks, 2013.