International Conference on Machine Learning and Computational Intelligence-2017
International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2017 IJSRCSEIT | Volume 2 | Issue 7 | ISSN : 2456-3307

Static Malware Analysis : A Case Study

Satya Narayan Tripathy, Sisira Kumar Kapat, M. Soujanya, Susanta Kumar Das
Department of Computer Science Berhampur University, Odisha, India

ABSTRACT

In the arena of digitized era, everyone needs internet connectivity for seeking and sharing of information. Starting from sharing information to social networking, each task requires internet. Some of the malware take advantage of this, and use user activities to activate. Hence the vector will be SDN (Software Defined Network) and SNS (Social Networking Sites). In both the cases, the user cannot be pretended to be a malware specialist or a computer professional who can detect the malicious activity easily. Although a lot of anti-malware tools are available, but it is good if the user can predict the malware. This paper focuses to analyze a malware easily and effectively, which a normal user can capture.

Keywords : Malware, Static Analysis, Case Study, Portable Executable, Common Strings.

I. INTRODUCTION

Malware infection is trending in SNS and SDN with the growth of new technologies. The recent topics in malware infection include Ransomware, ad fraud malware, android malware, botnets, banking Trojans and adware. This paper focuses on the static analysis of malware; including analysis of three malwares as examples. This research activity is based on simple concepts which a normal user can understand. The complete research is to detect (or can predict) malware existence in user computer. This paper considers the user is not a computer professional so we do not consider the malware family rather we focus on the files or process collected from user computer. Here three examples are given to make a general understanding about the malwares.

Malware is a computer program which directly or indirectly affects another computer program [9]. To be a malware a computer program must satisfy either of the malware criteria as, follows.

1. It must replicate itself and/or
2. Copy, remove or modify other files or programs.

In most of the cases, the computer malware is a PE (Portable Executable) file. The portable executable file has several sections. Out of the several sections, PE header is the most common one. The user can open any suspected file in an editor (like notepad). If the first two letters are found to be ‘MZ’, then the user may confirm that, the file is a PE file. One example is presented in Figure 3 and Figure 4. The file which is placed in the ‘startup’ will be initialized first when the computer system starts. So the most common classic place of malware is ‘startup’. The user can search the location before searching any other location in the system.

The existing malware analysis is based on some tools and techniques. The most common tools used for malware analysis are x32dbg/x64dbg, API monitor, PE explorer etc. and virtual machine is the most common environment for malware analysis. Some malware uses anti-VME (anti virtual machine environment) technique [8]. This implies the malware cannot be executed successfully in a virtual machine for analysis. There are two methods available for malware analysis, such as static malware analysis and dynamic malware analysis.

In static malware analysis [10], the malware is analyzed before it runs in the device. It includes, disassembling the source program which is not possible for each case. In dynamic analysis, the malware is executed and monitored the behavior. Dynamic analysis can monitor the instructions by executing the malware in a virtual environment such that, it won't affect the host operating system. The malware is then passed through behavioral study. In both the cases some merits and demerits are there but still static analysis beats dynamic analysis in speed [11,12].

II. RELATED WORK

According to Andrew & Srinivas[1], signature based detection was good. So they focused on modifying the existing signature detection technique. They analyzed the malware and collected the API call sequence. Each windows API is mapped into 32 bit integer id number. The obtained signature is used for similarity measurement with the existing signature database. For similarity measurement Euclidian distance, sequence alignment and similarity function methods is used. The analysis is based on static scanning means no sandboxing, proxy testing or code de-obfuscation. This technique holds good for polymorphic and metamorphic malware.

Madhu et al.[2], presented a methodology for composing signature of malware codes from Portable Executable is presented. They presented two methods for malware detection such as Static Analyzer for Vicious Executables (SAVE) and Malware Examiner using Disassembled Code (MEDiC). MEDiC uses assembly calls for analysis and SAVE uses API calls (Static API call sequence and Static API call set) for analysis. According to them, Assembly can be superior to API calls in that it allows a more detailed comparison of executable. API calls, on the other hand, can be superior to Assembly for its speed and its smaller signature.

Karishma et al.[3] collect some data set from benign files and spyware files and then they used java programming for Hexadecimal dumps for byte sequence Generation. Then they generated n-gram, for that they used hexadecimal dumps to converted into 'n' of fixed size and stored in HashMap, for later updation in the database. These are used for feature extraction by using Frequency-based Feature Extraction (FBFE) approach. The features with high frequency are being considered for training the classifier. The features with low frequency are ignored after this step the classifier is trained for model training they used Naïve Bayes Algorithm for classifying. The limitation of their approach is, regular explicitly searches for a process.

Ankur[4] discusses about basic outline of malicious codes and especially spywares and their detection using different techniques and also they told that the installation of spyware normally involves Internet Explorer. The main reason is the popularity of internet explorer that has made it target of spywares. Its deep integration with the Windows environment makes it prone to attack into the Windows operating-system. Internet Explorer also serves easy environment for spyware in the form of Browser Helper Objects, which modify the browser's behavior to add toolbars or to redirect traffic.

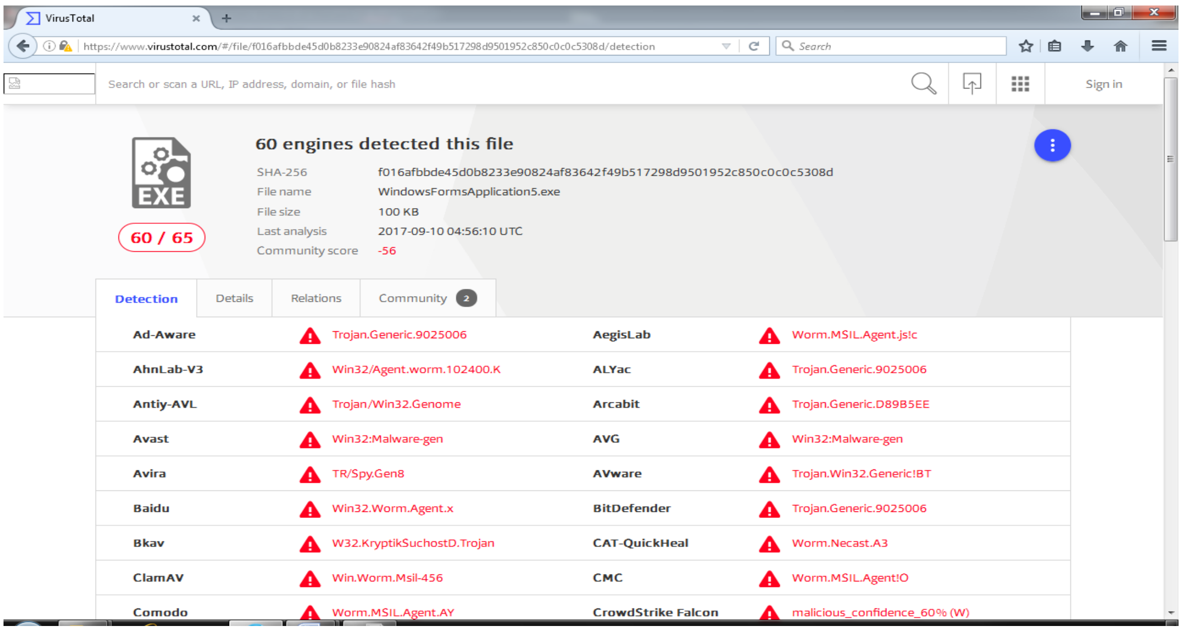
Gerardo et.al.[5] introduces a detection technique that assume that a side effect of the most common metamorphic engines it is the dissemination of a high number of repeated instruction in the body part of virus. Also they evaluate their technique. That method is more effective for static analysis rather than dynamic analysis. They used frequency analysis of instruction presetting disassemble code to detect the malware.

Case study

This paper considers three cases, where the malwares considered are, “New folder.exe”, “tongji.js” and “suchost..exe”. These malwares were in one of the infected device which was left infected purposefully. The files which are collected from the device are analyzed and discussed below as the case studies.

Case-1: suchost..exe (svchost..exe)

The ‘suchost..exe’ file is a PE (Portable Executable) file. This file runs another process called ‘svchost..exe’ which sounds similar to svchost.exe. The svchost.exe is a system file which is well visible in the task manager of any computer system. To confirm this file as a malware, this file if processed in virustotal assuming that, all the anti-malware engines used in virustotal is updated. The below Figure 1. (showing suchost..exe is a malware, virustotal output) shows that out of 65, 60 malware engines confirms this file to be a malware.



VirusTotal

https://www.virustotal.com/#/file/f016afbbde45d0b8233e90824af83642f49b517298d9501952c850c0c0c5308d/detection

Search or scan a URL, IP address, domain, or file hash

Sign in

60 / 65

60 engines detected this file

SHA-256	f016afbbde45d0b8233e90824af83642f49b517298d9501952c850c0c0c5308d
File name	WindowsFormsApplication5.exe
File size	100 KB
Last analysis	2017-09-10 04:56:10 UTC
Community score	-56

Detection | Details | Relations | Community 2

Ad-Aware	Trojan.Generic.9025006	AegisLab	Worm.MSIL.Agent.js!c
AhnLab-V3	Win32/Agent.worm.102400.K	ALYac	Trojan.Generic.9025006
Antiy-AVL	Trojan/Win32.Genome	Arcabit	Trojan.Generic.D89B5EE
Avast	Win32:Malware-gen	AVG	Win32:Malware-gen
Avira	TR/Spy.Gen8	AVware	Trojan.Win32.Generic!BT
Baidu	Win32.Worm.Agent.x	BitDefender	Trojan.Generic.9025006
Bkav	W32.KryptikSuchostD.Trojan	CAT-QuickHeal	Worm.Necast.A3
ClamAV	Win.Worm.Msil-456	CMC	Worm.MSIL.Agent!O
Comodo	Worm.MSIL.Agent.AY	CrowdStrike Falcon	malicious_confidence_60% (W)

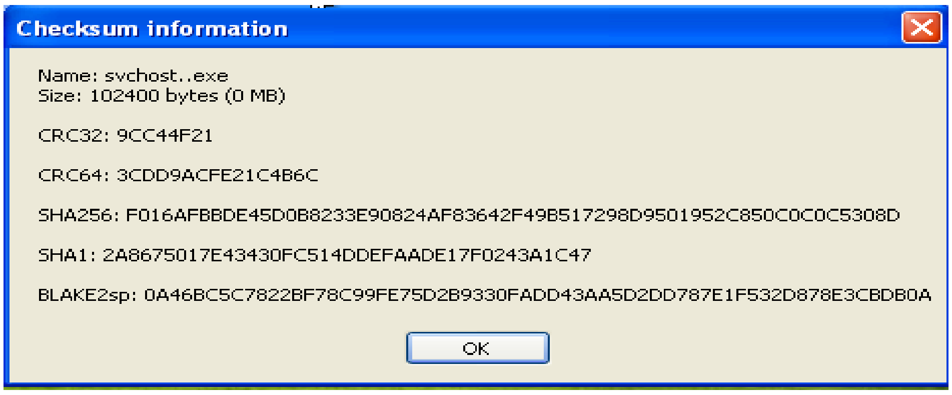
Figure 1. (showing suchost..exe is a malware, virustotal output)

Whenever suchost..exe runs,

1. It copies itself to two positions as (collected from source file)
   a. AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup \svchost..exe
   b. SystemDrive \Documents and Settings \Users \Documents\suchost..exe
2. It creates two processes as suchost..exe and svchost..exe, which can be clearly visible in task manager. It may not run in win-xp environment and it requires .NET framework to run in post win-7 environment.
3. Creates two registry keys
   a. (System drive :\...)\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\svchost..exe
   b. (System drive :\...)\Documents\suchost..exe
4. The two registry keys given above are placed in four registry locations such as,
   a. HKEY_CLASSES_ROOT/Local Settings/Software/Microsoft/windows/ shell/MuiCache

b. HKEY_CURRENT_USER/Software/Classes/Local Settings/Software/ Microsoft/windows/shell/MuiCache
c. HKEY_USERS/s-1-5-21-2528868183-3685655094-3686021654-1000/ Software/Classes/Local Settings/Software/Microsoft/windows/shell/ MuiCache
d. HKEY_USERS/s-1-5-21-2528868183-3685655094-3686021654-1000_Classes/ Local Settings/Software/Microsoft/windows/ shell/MuiCache

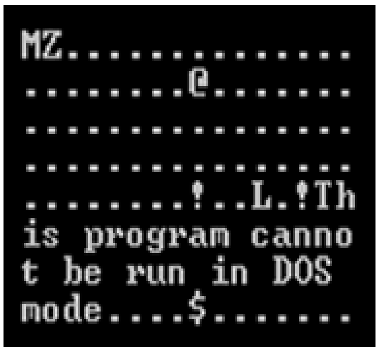
5. The Checksum information collected from win-xp sp2, 32 bit environment is as shown in Figure 2 . (Showing Checksum information of suchost..exe):



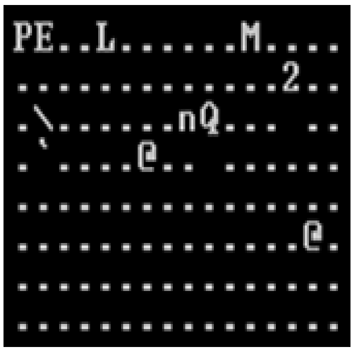
```
Checksum information

Name: svchost..exe
Size: 102400 bytes (0 MB)

CRC32: 9CC44F21

CRC64: 3CDD9ACFE21C4B6C

SHA256: F016AFBBDE45D0B8233E90824AF83642F49B517298D9501952C850C0C0C5308D

SHA1: 2A8675017E43430FC514DDEFAADE17F0243A1C47

BLAKE2sp: 0A46BC5C7822BF78C99FE75D2B9330FADD43AA5D2DD787E1F532D878E3CBDB0A
```

Figure 2 . (Showing Checksum information of suchost..exe)

To be clear about this file, the user may open this file in a notepad or the user may use some external software for analysis. Some portions of 'suchost..exe' is presented in the figures. From the Figure 3: (Shows 'MZ' are the first two letters) and Figure 4: (Shows PE header of suchost..exe) it is clear that the file starts with letter 'MZ' and the file is a PE (Portable Executable) file.



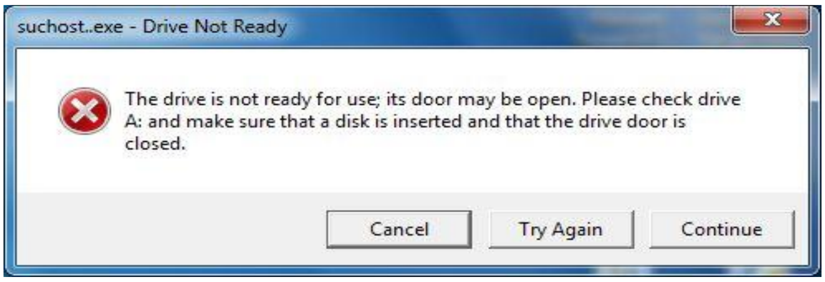
```
MZ..............
........@.......
................
................
........!..L.!Th
is program canno
t be run in DOS
mode....$.......
```



```
PE..L......M....
.............2..
.\......nQ... ..
.`....@.. ......
................
..............@.
................
................
```

Figure 3. (Shows 'MZ' are the first two letters) **Figure 4.** (Shows PE header of suchost..exe)

In 32 bit environment the file shows a message dialog box, whose message body is "drive not ready" and this is an infinite loop hence the user will be unable to close the dialogue box. The message box is shown in Figure 5. (Drive not ready message by suchost..exe) below.


suchost..exe - Drive Not Ready

The drive is not ready for use; its door may be open. Please check drive A: and make sure that a disk is inserted and that the drive door is closed.

Figure 5. (Drive not ready message by suchost..exe)

Case-2: tongji.js

```
<script type="text/javascript" src="http://web.nba1001.net:8888/tj/tongji.js">
</script>
```

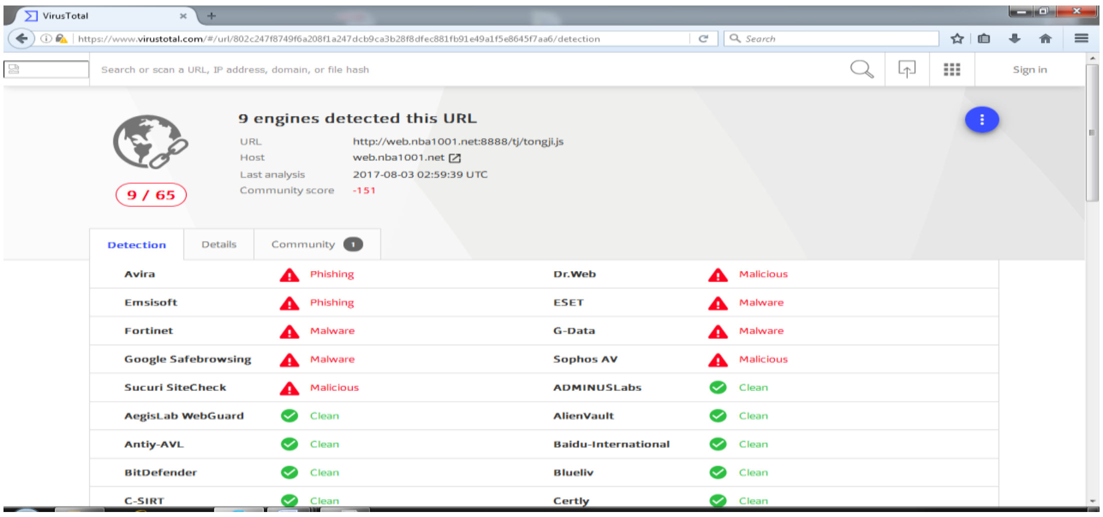
Figure 6. (Embeded malicious source in an HTML file)

The code statement is collected from one of the infected device. This is a malware which appends its execution code statement to an HTML file. This statement calls the original javascript program using the URL and executes it whenever the html program is connected to the network. The architecture of this malware is shown in Figure 7. (Architecture of 'tongji' malware).

HTML Document
Malware Code (Append)

Figure 7. (Architecture of 'tongji' malware)

This URL is being analyzed with virustotal and the result is shown in Figure 8. (showing tongji link given above is malicious, virustotal output).


VirusTotal
https://www.virustotal.com/#/url/802c247f8749f6a208f1a247dcb9ca3b28f8dfec881fb91e49a1f5e8645f7aa6/detection
Search
Search or scan a URL, IP address, domain, or file hash
Sign in
Detection
Details
Community

9 engines detected this URL

9 / 65

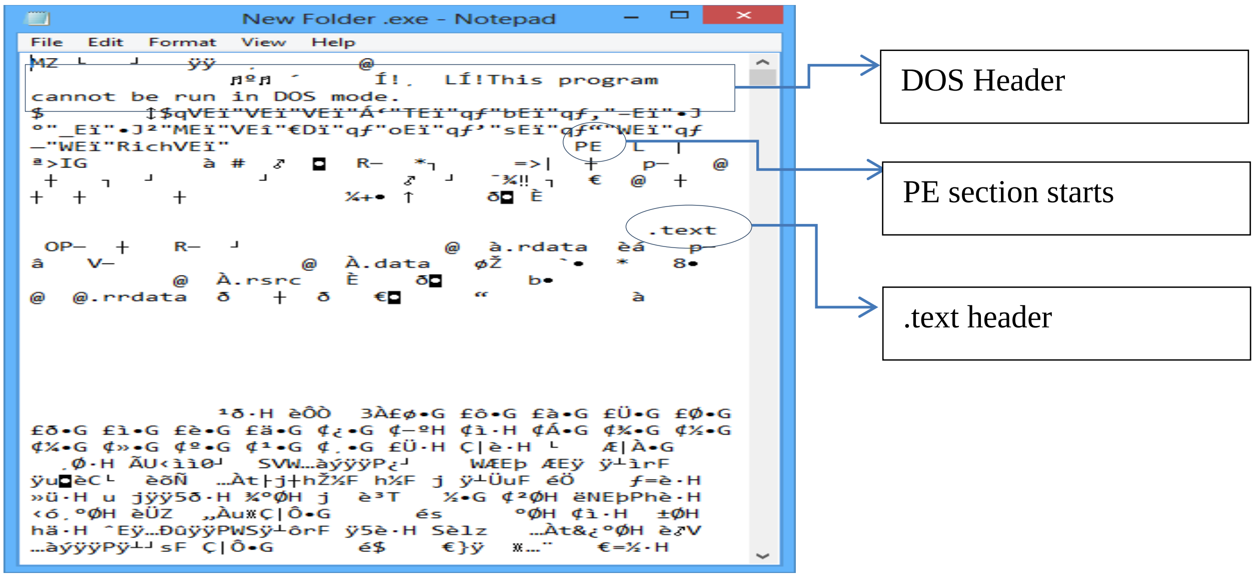
URL	http://web.nba1001.net:8888/tj/tongji.js
Host	web.nba1001.net
Last analysis	2017-08-03 02:59:39 UTC
Community score	-151

Avira	Phishing	Dr.Web	Malicious
Emsisoft	Phishing	ESET	Malware
Fortinet	Malware	G-Data	Malware
Google Safebrowsing	Malware	Sophos AV	Malicious
Sucuri SiteCheck	Malicious	ADMINUSLabs	Clean
AegisLab WebGuard	Clean	AlienVault	Clean
Antiy-AVL	Clean	Baidu-International	Clean
BitDefender	Clean	Blueliv	Clean
C-SIRT	Clean	Certly	Clean

Figure 8. (showing tongji link given above is malicious, virustotal output)

Case-3: NewFolder.exe

This is another malware whose labeled sample Figure 9. (Sample of 'NewFolder.exe' malware) is as follows.



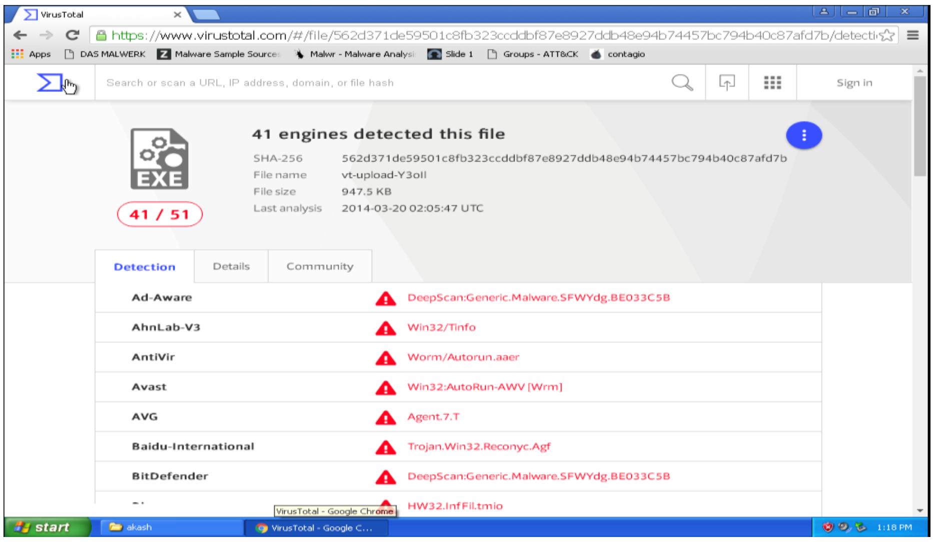
New Folder .exe - Notepad

File Edit Format View Help

```
MZ                ÿÿ          @
          ¤º´    ´            Í!¸  LÍ!This program
cannot be run in DOS mode.
$         ‡$qVEï"VEï"VEï"Ä‹"TEï"qf‚"bEï"qf‚"—Eï"•J
º"_Eï"•J²"MEï"VEï"€Dï"qf‚"oEï"qf‚"sEï"qf‚"WEï"qf
—"WEï"RichVEï"
ª>IG              à  #    PE     L
                                                  .text
 OP—   +    R—                @   à.rdata   èá
à    V—           @   À.data    ØŽ
           @   À.rsrc     È     ð
@   @.rrdata   ð    +    ð    €
```

Figure 9. (Sample of 'NewFolder.exe' malware)

Another virus like NewFolder.exe places its original file somewhere else and the host file which calls the original file to be executed is placed in the memory like pen drive or any other location in the system. This virus gets executed in the background in hidden mode. Hence it is difficult to be traced. The system will slow down, if this virus is executed. Virustotal depicts out of 51malware engines, 41 malware engines shows the 'NewFolder.exe' is a malware which is shown in Figure 10. (showing NewFolder.exe is malicious, virustotal output).



https://www.virustotal.com/#/file/562d371de59501c8fb323ccddbf87e8927ddb48e94b74457bc794b40c87afd7b/detection

41 engines detected this file

41 / 51

SHA-256 562d371de59501c8fb323ccddbf87e8927ddb48e94b74457bc794b40c87afd7b
File name vt-upload-Y3oll
File size 947.5 KB
Last analysis 2014-03-20 02:05:47 UTC

Detection | Details | Community

Ad-Aware	DeepScan:Generic.Malware.SFWYdg.BE033C5B
AhnLab-V3	Win32/Tinfo
AntiVir	Worm/Autorun.aaer
Avast	Win32:AutoRun-AWV [Wrm]
AVG	Agent.7.T
Baidu-International	Trojan.Win32.Reconyc.Agf
BitDefender	DeepScan:Generic.Malware.SFWYdg.BE033C5B
	HW32.InfFil.tmio

Figure 10. (showing NewFolder.exe is malicious, virustotal output)

III. CONCLUSION

This paper is based on non-debugging and non-disassembling technique, which is quite easy for a normal user. With a little knowledge about the text strings observed in the editor (when a file is opened), a malware can be detected. If there is a little knowledge of API is there, then it is very much easy for a normal user. Different authors used static methodology but, they mean to discuss the malware before execution. Here the files are analyzed before execution. If at all the file is executed, then the user can analyze the malware. Here www.virustotal.com is used for malware confirmation. For the analysis, the user can use the common strings which are quite helpful for most of the cases. The examples given are real-time examples and based on the view of a normal user.

IV. REFERENCES

[1]. A.H.Sung, J. Xu, P.Chavez, S.Mukkamala, "Static Analyzer of Vicious Executables (SAVE)", Conference Paper, DOI: 10.1109/CSAC.2004.37, Source: IEEE Xplore, https://www.researchgate.net/publication/4115464 , 2005

[2]. Madhu K. Shankarapani, Subbu Ramamoorthy, Ram S. Movva, Srinivas Mukkamala, "Malware Detection using assembly and API call sequences", J ComputVirol (2011) 7:107-119, DOI 10.1007/s11416-010-0141-5, 2010

[3]. Karishma Pandey, Madhura Naik, Junaid Qamar , Mahendra Patil.," Spyware Detection Using Data Mining", International Journal for Research in Applied Science &Engineering,Technology(IJRASET) Volume 3 Issue III, March 2015

[4]. Ankur Singh Bist, "Spyware Detection Techniques", INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY Bist, 3(2): February, 2014

[5]. Gerardo Canfora, Antonio Niccolò Iannaccone, Corrado Aaron Visaggio, "Static analysis for the detection of metamorphic computer virusesusing repeated-instructions counting heuristics", J ComputVirol Hack Tech, DOI 10.1007/s11416-013-0189-0, 2013

[6]. Andreas Moser, Christopher Kruegel, EnginKirda, "Limits of Static Analysis for Malware Detection", 23rd Annual Computer security applications conference, 2007, http://rosaec.snu.ac.kr/meet/file/20090204paperc.pdf

[7]. Maryann Gong, Uma Girkar, Benjamin Xie, "Classifying Windows Malware with Static Analysis", https://courses.csail.mit.edu/6.857/2016/files/5.pdf

[8]. Norkhushaini Awang, Arifin Salleh, Mohamad Yusof Darus, "Manual Malware Analysis Using Static Method", International Journal of Computer Networks and Communications Security, 1(7), ISSN 2308-9830, pp. 324-328, 2013

[9]. Simon Kramer, Julian C.Bradfield, "A general definition of malware", DOI 10.1007/s11416-009-0137-1, J Comput Virol (2010) 6:105–114

[10]. B. Jaya Prasad, Haritha Annangi, Krishna Sastry Pendyala, "Basic Static Malware Analysis using open source tools", https://securitycommunity.tcs.com/infosecsoapbox/sites/default/files/Static%20Malware%20Analysis%20Techniques%20.pdf

[11]. Mohd. Ishrat, Manish Saxena, Dr. Mohd. Alamgir, "Comparison of Static and Dynamic Analysis for Runtime Monitoring", International Journal of Computer Science & Communication Networks, ISSN:2249-5789, Vol 2(5), 615-617

[12]. Ronghua Tian, "An Integrated Malware Detection and Classification System", Ph.D. Thesis, Deakin University, 2011