AUTOMATED MALWARE ANALYSIS

Shridhar Ravi [1], Sparsh Arora [2], Kaustubh Badrike [3], Sanket Bailmare [4], Aparna Bannore[5]
Department of Computer Engineering
SIES Graduate School of Technology, Navi Mumbai, India

ABSTRACT:
Security is a serious concern in this digital age. Malware disrupts not only the workflow of organisations but also poses the threat of changing the control flow/data structures to the data of the stakeholders involved. A lot of malware pass undetected, because its behaviour in the system is often unknown as it may do the harm to the system without the physical presence in the memory. The risk of analyzing its execution is too high on a live system and may harm the functioning of the system.

A secure solution is sandbox analysis, which mitigates the risks to a virtual environment. It helps in analyzing types of malware in a safe isolated environment that replicates an end user operating environment. Our solution aims at generating the execution of the suspicious file inside the sandbox environment, track its activities and record it in a report classifying files accordingly, using application-run based malware detection.

***KEYWORDS*: Malware, Sandbox, Analysis,**

I. INTRODUCTION

Malware is a file or code, typically delivered over a network, that infects, explores, steals or conducts virtually any behaviour an attacker wants. Software which is specifically designed to disrupt, damage, or gain authorized access to the computer. Malware is a growing concern in the modern connected and machine dependent world. The intent of a malicious file could be to steal confidential data or obtain root privileges. By getting unauthorized access an attacker can easily use the identity of the victim or can damage a corporate with their critical data. The complexity of malware keeps evolving tremendously, with every advancement in technology and computational resource because of which it has become more troublesome to crack a malware. Therefore, malware analysis is a big business, as an attack can cost a company dearly.
Figure [1] depicts the detection rate for various types of malware.

Malware takes control of your computer or network and all the software running on it. It may alter and delete files, including the reformatting of a machine's hard drive, causing a complete loss of any information that is not backed up. It may Steal sensitive information, including proprietary information and customer data like credit card and social security numbers. Malware may send malicious emails or network traffic on your behalf or it may install pop-up messages or lock your computer and redirect you to a tech support firm or criminal asking for payment in order to restore the machine or data.

A study suggests that the impact of malware is getting worse day by day especially in the financial section, the complexity of malware keeps evolving tremendously, with every advancement in technology and computational resource because of which it has become more troublesome to crack a malware. Therefore, malware analysis is a big business, as an attack can cost a company dearly. Figure [2] depicts the findings of a study conducted by Kaspersky Lab and B2B International regarding the spread and effects of malware infection.

Following are some different types of malware:

1. Worms - It is a malware program that replicates itself in the computer. It almost causes some network harm, even if it is to consume the bandwidth or space.

2. Virus - It is a malware code that when executed, replicates itself by modifying another computer program and inserting its own code in between which then corrupts or destroys data.

3. Trojan Horse - It is a type of malicious software which disguises itself as legitimate software to gain access to a victim's system. Users get tricked some attractive social media ads who then redirect them to malicious websites.

4. Rootkit - It is defined as malicious computer software that is hidden deep inside a computer and therefore remains undetectable. Worms, bots, malware can be hidden inside them and the attacker can have root access using it.

5. Spyware - It is a software that can be used to gather information about an individual or an organization without their

knowledge, and may send this to another entity without their consent.

6. Keyloggers - It can be used to record the keystrokes of the victim's system which can lead to a leak of confidential data and more

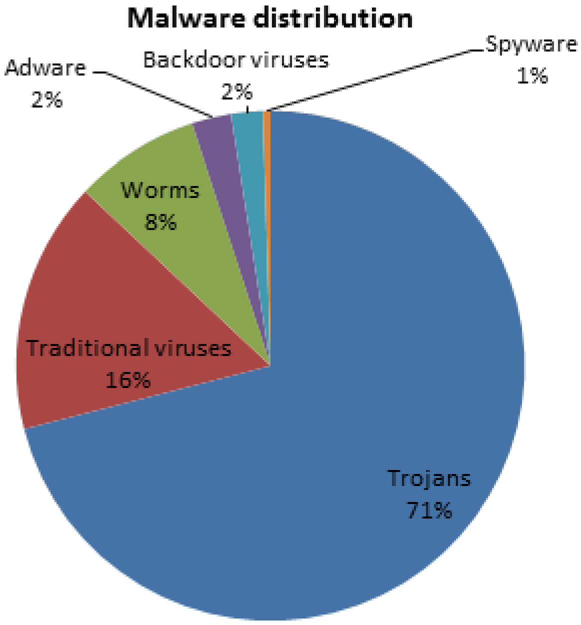


Fig. 1 Detection rate for various types of malware. [8]

A. Static Analysis: Static analysis refers to the analysis of a file without its execution, static analysis systems are equipped with a database of regular expressions that specify byte or instruction sequences that are considered malicious. A program is declared malware when one of the signatures is identified in the program's code. Limitations of static analysis are due to the fact the hackers try to circumvent the detection techniques which results in unnoticed malicious content. While it is conceivable to improve static analysis to handle more advanced obfuscation techniques, there is a fundamental limit in what can be decided statically.

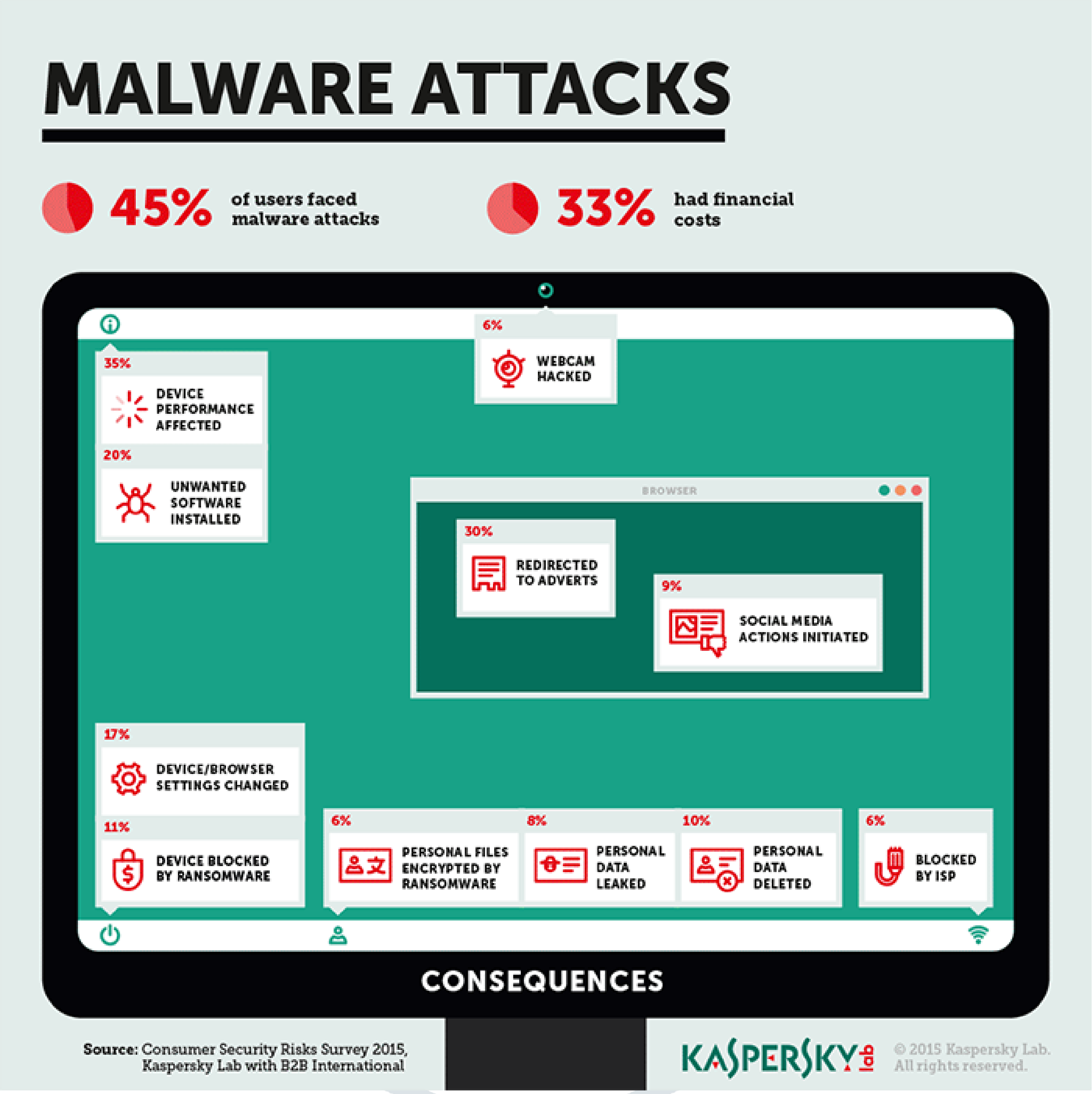


Fig. 2 Impact of Malware attacks [7]

B. Antivirus: Antivirus software scans the file comparing specific bits of code against information in its database and if it finds a pattern duplicating one in the database, it is considered a virus, and it will quarantine or delete that particular file. Antivirus software relies on virus definitions to detect malware. Due to which it automatically downloads the new, updated definition files – once a day or even more often. The definition files contain signatures for viruses and other malware that have been encountered in the wild. Some genuine software might do a system call that may raise a flag of an antivirus causing the computer to crash. Hence Antivirus is not always a feasible solution while dealing with malicious content.

II. LITERATURE REVIEW

Sujyothi, Akshatha, and Shreenath Acharya [1] identified that traditional hierarchical classification approach is slow and gets ineffective with the increased complexity of malicious activity and is unable to target this high scale of production of new malware. Therefore, the dynamic analysis of malware is proposed. The system captures the execution trace of the file, thereby identifying the malicious trace. The system mainly consists of the analysis module and the classification module. The analysis module is used for mostly doing the preprocessing task such as generating data suitable for the classification module. Classification module then differentiates these samples into their corresponding neighbours. The main aim of this paper is to reduce the rate of false identification of malware by isolating malicious traces from within the execution traces generated during analysis. Figure [3] depicts the flow diagram of system proposed by the authors.

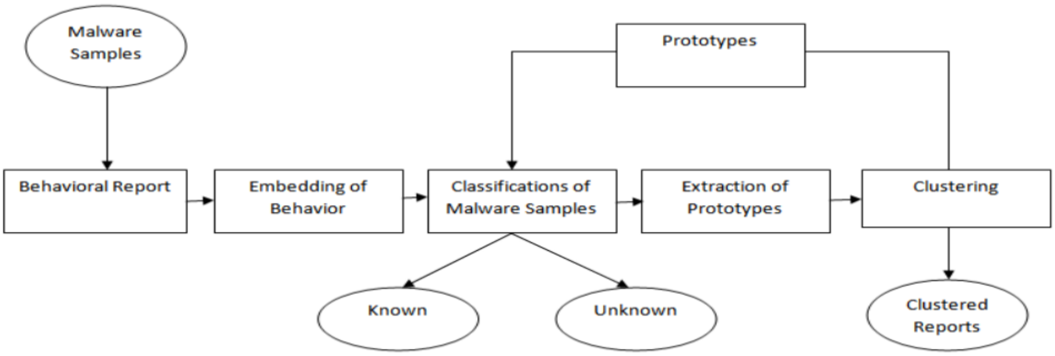


Fig. 3 Flow diagram showing overall activities of the system [1]

Branco, Rodrigo Rubira, and Udi Shamir [2] focus on the challenges faced in malware analysis stating that thousands of samples need to be analyzed per week and to the complexity of some malware to avoid automated analysis. They propose a scheduler which determines the availability of the machine and also runs the sample in a dedicated machine if it refuses to run in a virtualized environment. A packet sniffer is used to detect traffic generated and dissector analyze captured packet and generate a report about network behaviour. Then the collected information is used to group together the instance of malware based on the fact that they all use the same library call. A statistical view of the analysed sample is also provided. This paper also focuses on how the machine should be selected for efficient and faster analysis. The paper proposes that they are able to perform faster analysis than compared to the result of multiple different anti-viruses.

“Practical malware analysis based on sandboxing.” [3] provides that an automated malware analysis submits a similar kind of results that of manual analysis but in a considerable amount of time. This paper states the alarming low percentage of malware detection. They proposed an idea of a distributed firewall, called Distfw, implemented using IPtables for filtering traffic and IPsec for securing communication. Cuckoo sandbox solution was integrated to automatically analyse the malicious application. The paper emphasises that by the use of not only the time to is reduced but also this technique can help increase the security of a system.

Shijo, P. V., and A. Salim [4] mention how most of the anti-virus use signature-based detection which has become inefficient in present scenario due to the rapid increase in the number of variants of malware. This paper proposes an integrated static and dynamic analysis method to analyse and classify an executable file. This paper discusses both the advantages and disadvantages of both static and dynamic malware analysis, therefore a solution is provided by integrating both of them. Machine learning is used in which known malware and benign programs are used as training data. The result of this experiment shows an accuracy of 95.8% using static, 97.1% using dynamic and 98.7% using an integrated method. This paper aims to implement a system that is able to use the advantages of both static and dynamic malware analysis.

“Cuckoo Sandbox Book” [5] provides valuable insight regarding using sandbox architecture for implementations, such as architecture and use cases. The key information for the effectiveness of the solution comes from factors such as the types of files, the volume of analysis handled, a platform for analysis, the desired parameters from analysis and the setup to study particular exploits. It also discusses some of the problems with virtualization traces, however, they can be undermined due to the availability of countermeasures.

III. PROPOSED SYSTEM

This paper gives the dynamic and static approach of analysis of the malicious files. The file would be analysed by matching the analysis rules and the result/score of it would be noted. The files will be executed in a virtual lab so that these files are isolated from the host environment and decrease the risk of potential harm that may cause to the system. The virtual lab consists of a machine which has a minimum firewall against the malware which allows our file to execute without any constraints.The behaviour of this file would be monitored at the execution and the remarkable changes in the system would be monitored and recorded. A readable report would be presented as an output giving all the details. Figure [4] represents the block diagram of our proposed system.

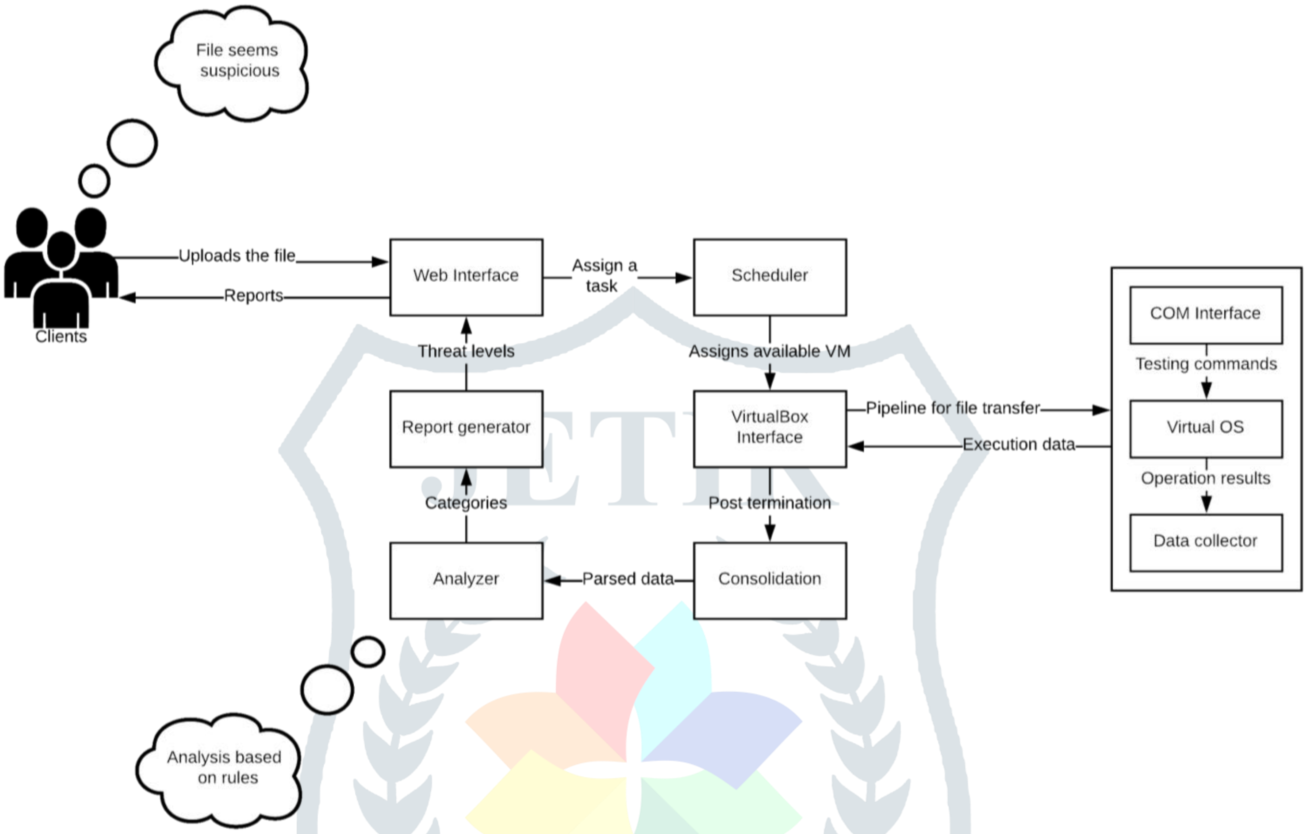


Fig. 4: Block Diagram of Proposed System

The system consists of a host machine, a virtual network and several analysis guests. The host machine performs various responsibilities, including:

I. Web server for human interface
II. Managing the virtual network
III. Assign files to virtual machines
IV. Analysis of result data
V. Report generation
VI. Maintain records of client tasks to extract information to prevent future potential attacks.

To perform these activities, the host utilizes the following technologies:

I. Oracle VirtualBox SDK
II. Python 2.7
III. Linux distribution

The analysis guest is a virtual machine, which executes and monitors the incoming file, and transfers the result logs back to the host. It can be created as needed by the host without any setup, from a predefined safe snapshot. It is hence dispensable, however, it must maintain the integrity of results. The guest utilizes the following technologies:

I. Windows Component Object Model
II. Python 2.7

IV. IMPLEMENTATION DETAILS

1. The web interface, built on a flask framework, accepts the suspicious file to be analyzed.

2. The file is sent to the Scheduler to be assigned to a virtual machine. The scheduler implements a Round Robin Scheduling Algorithm which uses historical data and the current state of the system and then computes ahead the influence it will have on the system after the deployment of the needed VM resources and then chooses the least-affecting solution, through which it achieves the best load balancing and reduces or avoids dynamic migration. Using this algorithm, the Scheduler allocates one VM for one file in a cyclic manner. This algorithm is similar to the Round Robin Scheduling for Process Scheduling.

3. The host system communicates with the virtual machine via the VirtualBox SDK, which has specific interfaces for data transfer. A pipeline is created with the assigned VM. The Network Address Translation (NAT) network on the virtual machine allows us to control what gets in or out of the virtual machine's network.

4. The file begins its execution in the virtual machine. The machine is monitored during this process for unauthorized network access, memory operations, file read-write operations, etc. All the data generated during this execution is gathered for analysing the type of file.

5. The gathered data is then transferred back to the host after the execution is complete, or the VM crashes. This data, which contains a lot of information, will help in the classification of the file as malicious or benign. It is analyzed using certain rules to determine the type of file.

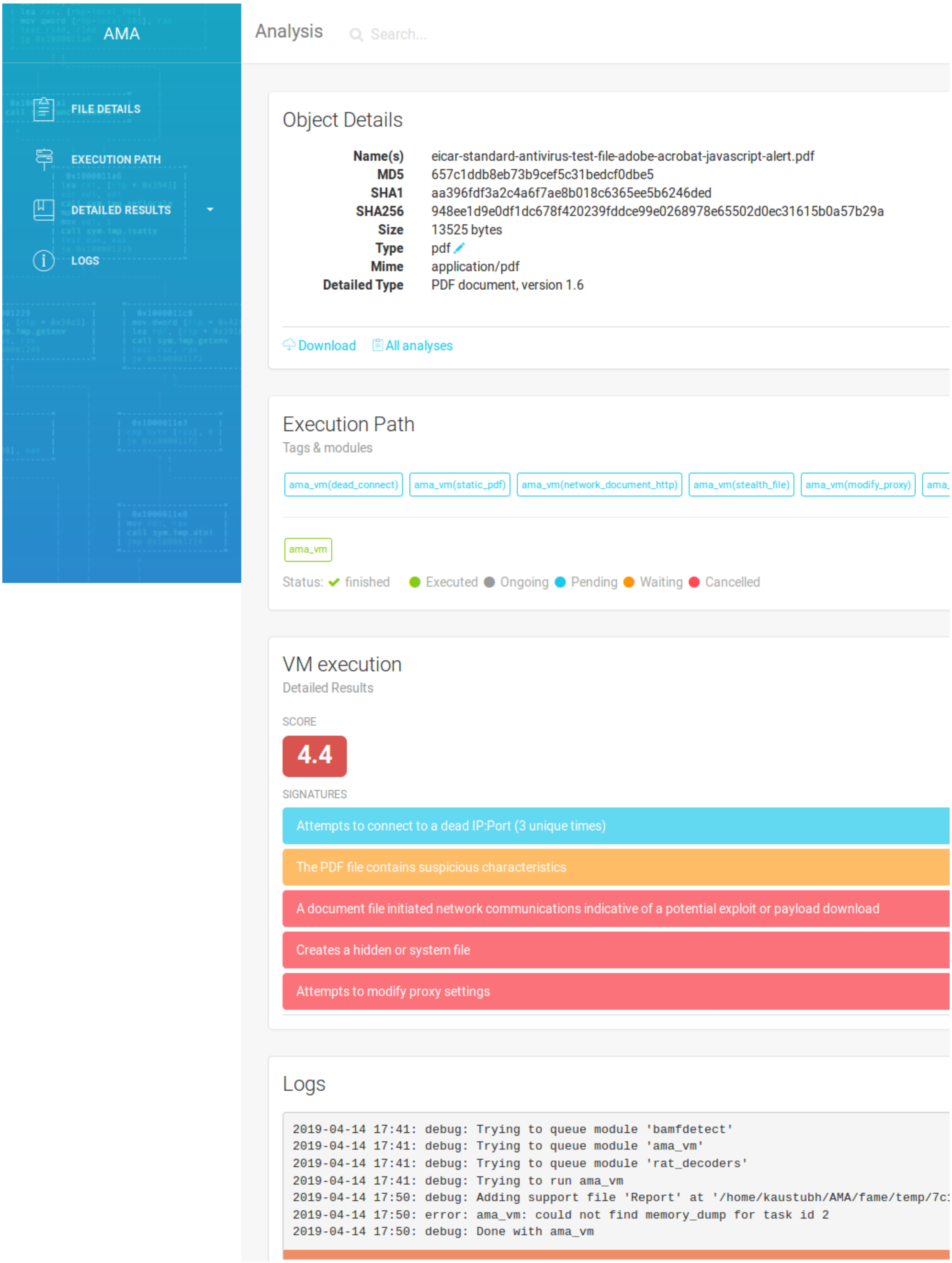
6. The web interface displays a detailed report of the file.

JETIR

V. RESULTS AND DISCUSSION

The primary purpose of the generated report is to serve as an indicator of the potential impact of the suspect file, and quantify it in meaningful but easily understandable parameters. These are represented as an overall threat score as well as a listing of identified Indicators of Compromise. Results from additional analysis modules may be queued, both conditionally and as routine. To emphasize the integrity of our results, we integrated our analysis process with an existing database of known threats. Standing on the shoulder of giants, this intelligence comes from over 50 antivirus research bases, aggregated by VirusTotal. The final score is provided in the interval [0-10], directly proportional to malevolent impact, with 0 indicative of completely benign sample

1) Figure [5] represents the report generated by the system on a standard eicar PDF file, with javascript alerts. The system gave a score of 4.4 out of 10, with the threat ranging from low to moderate, listing out the potential threats on running the file.



Object Details

Name(s)	eicar-standard-antivirus-test-file-adobe-acrobat-javascript-alert.pdf
MD5	657c1ddb8eb73b9cef5c31bedcf0dbe5
SHA1	aa396fdf3a2c4a6f7ae8b018c6365ee5b6246ded
SHA256	948ee1d9e0df1dc678f420239fddce99e0268978e65502d0ec31615b0a57b29a
Size	13525 bytes
Type	pdf
Mime	application/pdf
Detailed Type	PDF document, version 1.6

Download All analyses

Execution Path

Tags & modules

ama_vm(dead_connect) ama_vm(static_pdf) ama_vm(network_document_http) ama_vm(stealth_file) ama_vm(modify_proxy) ama_

ama_vm

Status: ✓ finished ● Executed ● Ongoing ● Pending ● Waiting ● Cancelled

VM execution

Detailed Results

SCORE

4.4

SIGNATURES

Attempts to connect to a dead IP:Port (3 unique times)

The PDF file contains suspicious characteristics

A document file initiated network communications indicative of a potential exploit or payload download

Creates a hidden or system file

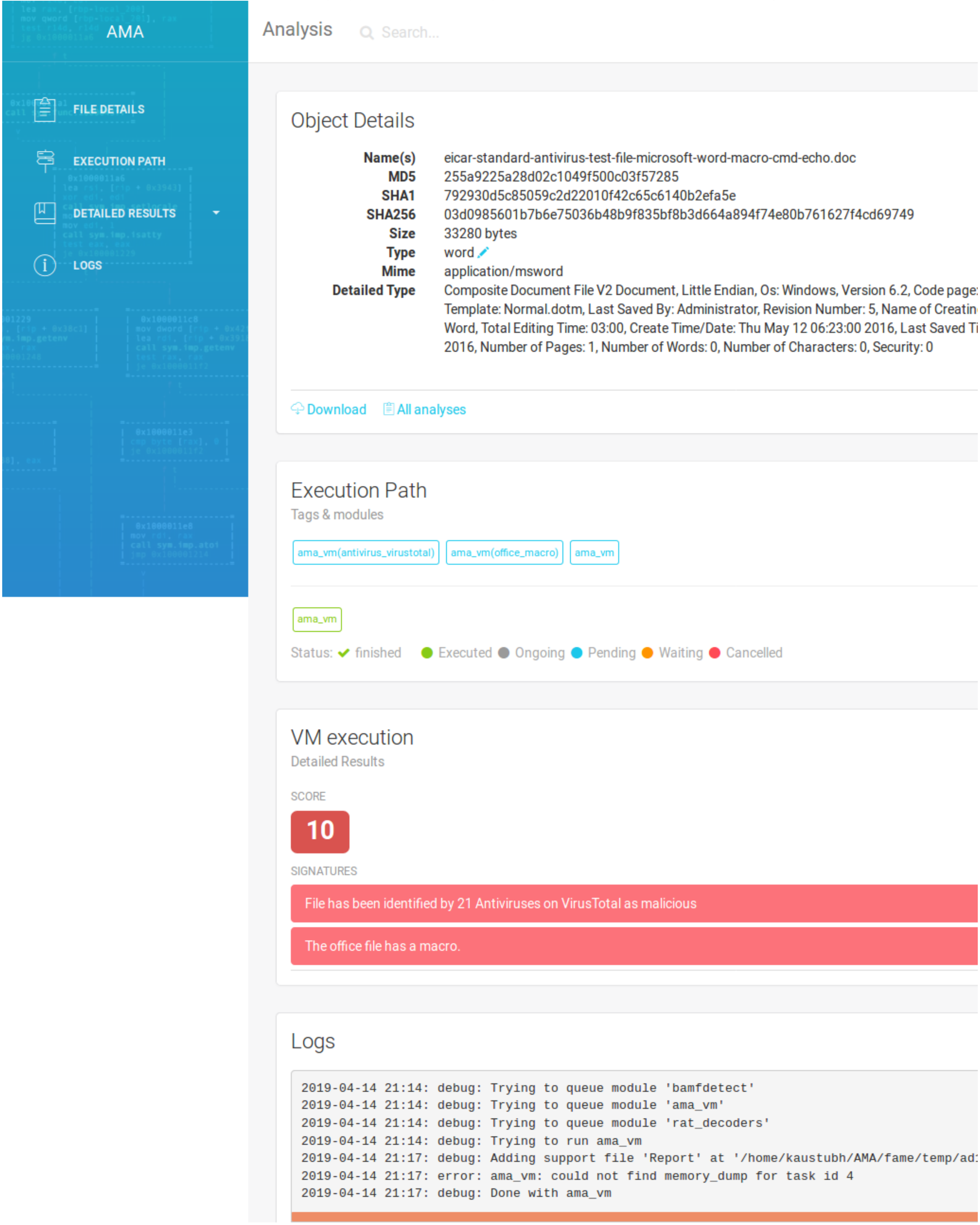
Attempts to modify proxy settings

Logs

```
2019-04-14 17:41: debug: Trying to queue module 'bamfdetect'
2019-04-14 17:41: debug: Trying to queue module 'ama_vm'
2019-04-14 17:41: debug: Trying to queue module 'rat_decoders'
2019-04-14 17:41: debug: Trying to run ama_vm
2019-04-14 17:50: debug: Adding support file 'Report' at '/home/kaustubh/AMA/fame/temp/7c
2019-04-14 17:50: error: ama_vm: could not find memory_dump for task id 2
2019-04-14 17:50: debug: Done with ama_vm
```

Fig. 5: Report generated on submitting a standard eicar PDF file, with javascript alerts

Figure [6] represents the report generated by the system on a standard eicar DOC file, with macros. The system gave a score of 10 out of 10, treated as highly malicious, listing out the potential threats on running the file.



Object Details

Name(s)	eicar-standard-antivirus-test-file-microsoft-word-macro-cmd-echo.doc
MD5	255a9225a28d02c1049f500c03f57285
SHA1	792930d5c85059c2d22010f42c65c6140b2efa5e
SHA256	03d0985601b7b6e75036b48b9f835bf8b3d664a894f74e80b761627f4cd69749
Size	33280 bytes
Type	word
Mime	application/msword
Detailed Type	Composite Document File V2 Document, Little Endian, Os: Windows, Version 6.2, Code page: Template: Normal.dotm, Last Saved By: Administrator, Revision Number: 5, Name of Creatin Word, Total Editing Time: 03:00, Create Time/Date: Thu May 12 06:23:00 2016, Last Saved Ti 2016, Number of Pages: 1, Number of Words: 0, Number of Characters: 0, Security: 0

Execution Path

Tags & modules

Status: ✓ finished ● Executed ● Ongoing ● Pending ● Waiting ● Cancelled

VM execution

Detailed Results

SCORE

10

SIGNATURES

File has been identified by 21 Antiviruses on VirusTotal as malicious

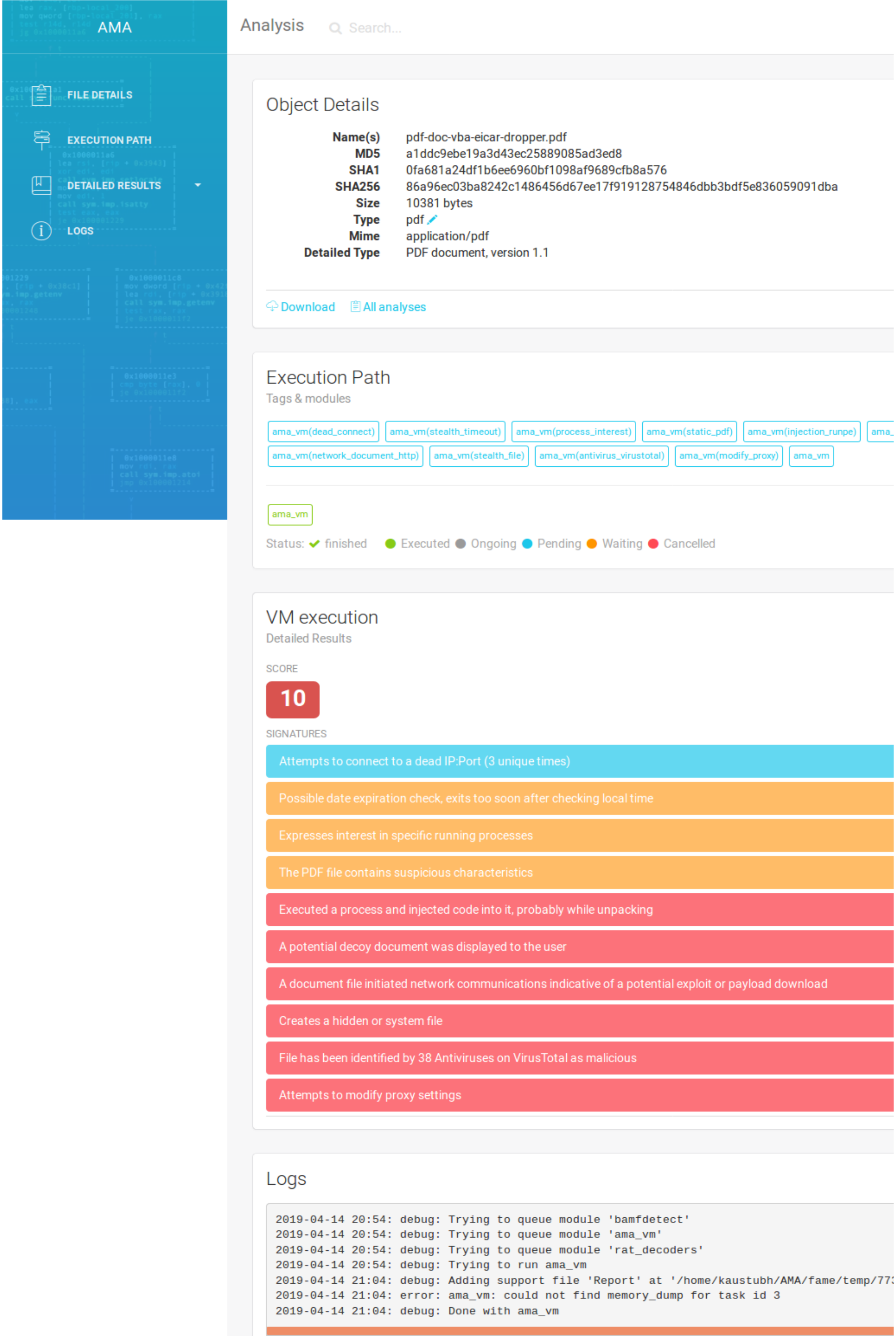
The office file has a macro.

Logs

```
2019-04-14 21:14: debug: Trying to queue module 'bamfdetect'
2019-04-14 21:14: debug: Trying to queue module 'ama_vm'
2019-04-14 21:14: debug: Trying to queue module 'rat_decoders'
2019-04-14 21:14: debug: Trying to run ama_vm
2019-04-14 21:17: debug: Adding support file 'Report' at '/home/kaustubh/AMA/fame/temp/ad
2019-04-14 21:17: error: ama_vm: could not find memory_dump for task id 4
2019-04-14 21:17: debug: Done with ama_vm
```

Fig. 6: Report generated on submitting a standard eicar DOC file, with macros

3) Figure [7] represents the report generated by the system on a standard eicar PDF file, with embedded DOC dropping. The system gave a score of 10 out of 10, treated as Highly malicious, listing out the potential threats on running the file.



Object Details

Name(s)	pdf-doc-vba-eicar-dropper.pdf
MD5	a1ddc9ebe19a3d43ec25889085ad3ed8
SHA1	0fa681a24df1b6ee6960bf1098af9689cfb8a576
SHA256	86a96ec03ba8242c1486456d67ee17f919128754846dbb3bdf5e836059091dba
Size	10381 bytes
Type	pdf
Mime	application/pdf
Detailed Type	PDF document, version 1.1

Execution Path

Tags & modules

ama_vm(dead_connect) ama_vm(stealth_timeout) ama_vm(process_interest) ama_vm(static_pdf) ama_vm(injection_runpe) ama_
ama_vm(network_document_http) ama_vm(stealth_file) ama_vm(antivirus_virustotal) ama_vm(modify_proxy) ama_vm

ama_vm

Status: ✓ finished ● Executed ● Ongoing ● Pending ● Waiting ● Cancelled

VM execution

Detailed Results

SCORE

10

SIGNATURES

Attempts to connect to a dead IP:Port (3 unique times)

Possible date expiration check, exits too soon after checking local time

Expresses interest in specific running processes

The PDF file contains suspicious characteristics

Executed a process and injected code into it, probably while unpacking

A potential decoy document was displayed to the user

A document file initiated network communications indicative of a potential exploit or payload download

Creates a hidden or system file

File has been identified by 38 Antiviruses on VirusTotal as malicious

Attempts to modify proxy settings

Logs

```
2019-04-14 20:54: debug: Trying to queue module 'bamfdetect'
2019-04-14 20:54: debug: Trying to queue module 'ama_vm'
2019-04-14 20:54: debug: Trying to queue module 'rat_decoders'
2019-04-14 20:54: debug: Trying to run ama_vm
2019-04-14 21:04: debug: Adding support file 'Report' at '/home/kaustubh/AMA/fame/temp/77
2019-04-14 21:04: error: ama_vm: could not find memory_dump for task id 3
2019-04-14 21:04: debug: Done with ama_vm
```

Fig. 7: Report generated on submitting a standard eicar PDF file, with embedded DOC dropping

VI. CONCLUSION

Growing concerns about the present scenario of security to protect the data and integrity of computing assets belonging to or connecting to an organization’s network across the world. Presently, the potential threat by a malware is primarily detected in testing laboratories due to legal liabilities to computing resources. Our solution provides an endpoint for general consumers to test their files ahead of schedule, with prototype analysis facilities emulating the process adopted by industrial research centers. This enables the user to make informed decisions about his system, and be independent of the database provided by his antivirus of choice.

The experimental hypothesis promises high efficiency in detecting malware due to the merit of our approach which lies in combining various machine recognizable Indicators of Compromise derived from the commonly known malicious behaviour. To mitigate false negatives, it also integrates with an established 3rd party malware informatics provider. However, the sensitivity to various parameters is non-deterministic, and may produce uncertainty and fluctuations in the final test score between files with the same perceived severity. The score is dependent on the analysis time specified, as some malware may conduct activities of higher severity in the later stages of execution. This also results in high turnover times for the system, which can be reduced by the inclusion of high performance analysis machines but not completely eliminated as it is highly dependent on the execution path of the malware.

ACKNOWLEDGEMENT

We would like to express our deepest appreciation to all those who provided us the possibility to complete this report. A special gratitude we give to our final year project guide, Prof. Aparna Bannore, whose contribution in stimulating suggestions and encouragement, helped us in our research.

We would like to express deepest appreciation towards Dr. Vikram Patil, Principal, SIES Graduate School of Technology, Prof. Dr. Rizwana Shaikh, Head of Department of Computer Engineering and Prof.Masooda Modak,Project Coordinator whose invaluable guidance supported us in our research. We extend our gratitude to the industry professionals and academic researchers for building a solid foundation on which to further our ambitious efforts, and for the continuous technological innovations that keep our field of interest from obsolescence.

REFERENCES

[1] Sujyothi, Akshatha, and Shreenath Acharya. “Dynamic Malware Analysis and Detection in Virtual Environment.” *International Journal of Modern Education and Computer Science*, vol. 9, no. 3, 2017, pp. 48–55., doi:10.5815/ijmecs.2017.03.06.

[2] Branco, Rodrigo Rubira, and Udi Shamir. “Architecture for Automation of Malware Analysis.” *2010 5th International Conference on Malicious and Unwanted Software*, 2010, doi:10.1109/malware.2010.5665786.

[3] Vasilescu, Mihai, et al. “Practical Malware Analysis Based on Sandboxing.” *2014 RoEduNet Conference 13th Edition: Networking in Education and Research Joint Event RENAM 8th Conference*, 2014, doi:10.1109/roedunet-renam.2014.6955304.

[4] Shijo, P.v., and A. Salim. “Integrated Static and Dynamic Analysis for Malware Detection.” *Procedia Computer Science*, vol. 46, 2015, pp. 804–811., doi:10.1016/j.procs.2015.02.149.

[5] Guarnieri C. “Cuckoo Sandbox Book” Cuckoo Sandbox Book - Cuckoo Sandbox v2.0.6 Book, 2017, cuckoo.sh/docs/.

[6] Oracle. “VirtualBox Main API Documentation.” VirtualBox Main API: Main Page, 14 Aug. 2018, www.virtualbox.org/sdkref/.

[7] Kaspersky Lab. “Problem Caused by Malware Attacks.” Kaspersky.com, 11 Sept. 2015, www.kaspersky.com/about/press-releases/2015_4-in-5-malware-attacks-cause-problems-for-users-and-1-in-3-result-in-money-loss.

[8] “A Definition of Malware.” BullGuard, www.bullguard.com/bullguard-security-center/pc-security/computer- threats/malware-denition,-history-and-classification.aspx.

[9] Notenboom, Leo A. “What's the Difference between a Sandbox and a Virtual Machine?” Ask Leo! by Leo A. Notenboom, 14 Jan. 2012, ask-leo.com/whats_the_difference_between_a_sandbox_and_a_virtual_machine.html.

[10] Alvarez, Victor M. “Welcome to YARA's Documentation!” Welcome to YARA's Documentation! - Yara 3.7.0 Documentation, 2018, yara.readthedocs.io/en/v3.7.0/.

[11] “Python 3.7.1 Documentation.” 3.7.1 Documentation, docs.python.org/3/.

[12] Satran, Michael. “Reference.” Microsoft Docs, docs.microsoft.com/en-us/windows/desktop/com/reference.

[13] “VirusTotal.” *VirusTotal*, www.virustotal.com/.