Journal of Computer Science

Original Research Paper

Rule-Based Approach to Detect IoT Malicious Files

[1]Faisal Alsattam, [1,2*]Mousa Al-Akhras, [1]Marwah M. Almasri and [1]Mohammed Alawairdhi

[1]College of Computing and Informatics, Saudi Electronic University, Riyadh, Saudi Arabia
[2]King Abdullah II School of Information Technology, The University of Jordan, Amman, Jordan

Article history
Received: 07-08-2020
Revised: 07-09-2020
Accepted: 10-09-2020

Corresponding Author:
Mousa Al-Akhras
College of Computing and Informatics, Saudi Electronic University, Riyadh, Saudi Arabia
and
King Abdullah II School of Information Technology, The University of Jordan, Amman, Jordan
Email: m.akhras@seu.edu.sa
mousa.akhras@ju.edu.jo

Abstract: The current immersive increase of cyber-attacks requires constant evolution of the used security solutions. Current malware detection solutions are only able to identify known malwares that were previously detected. They also lack the ability to deeply investigate every file in the system. Therefore, new detection techniques are needed to fill this gab. In this study, a flexible and an effective rule-based approach is proposed to detect malicious files by searching for specific types of strings that should not exist in normal legitimate files. The proposed detection technique relies on the use of LOKI as a scanning agent that uses customized YARA rules with different complexities to search for the needed strings. The proposed methodology has been tested and it detected all malwares successfully.

Keywords: Digital Forensics, IoT Forensics, LOKI, YARA Rules, IoT Malware

Introduction

The Internet usage has increased drastically during the past few years. Recently, the term "Internet of Things" (IoT) has become popular, where different devices are connected to the Internet to provide users with requests or services without being around which saves their time and makes their lives much easier. Due to the increased number of businesses/individuals who use IoT technologies, especially in critical domains such as health and military sectors, a lot of sensitive data are being sent/received. Therefore, security has become a crucial aspect in protecting these sensitive data.

Securing IoT devices and its data is not an easy task due to the numerous types of cyber-attacks. Current techniques to detect IoT malicious files are not mature enough due to their lack of accuracy, intensive processing power, complexity, inefficiency and time consumption.

Therefore, a more accurate and efficient IoT scanner that detects malicious files in a large volume of stored data is needed. According to (Demeter *et al*., 2019), the nature of IoT attacks is becoming more and more sophisticated and it is the same for malwares. More than 70% of the top IoT threats are originating from "NyaDrop", "Mirai" and "Gafgyt" malwares which have been used by attackers before 2016. Unfortunately, these malwares cannot be detected by normal Antiviruses. The reason is that the code is versatile enough and can be easily compiled in any level of complexity.

Different malwares have different strings, patterns and characteristics than user legitimate files. Hence, in this study, we use this information to find a fast, accurate and efficient identification technique that deeply investigates every stored file and then differentiate between legitimate and illegitimate files.

The proposed technique can be a solution to resolve two main security issues that have been faced by many organizations, which are:

- The difficulty of detecting new malware variants that has not been seen before
- The difficulty of searching for a specific Indicator Of Compromise (IOC) or a malicious string related to malicious activities

The rest of the paper is organized as follows: Literature review section discusses recent researches that provide different techniques and solutions to increase IoT security. The proposed methodology section explains our suggested method in detail and differentiates between legitimate and malicious files. The experimental setup and design section discusses the hardware requirements, data collection, network environment and the scanning process. The implementation section demonstrates how the proposed

methodology was carried out. Results and discussion section evaluates the efficiency of the proposed detection model. Limitations are identified and recommendations are pointed out in the limitations and recommendations section. Finally, the conclusions section summarizes the outcomes based on the findings.

Literature Review

Several researches discussed various techniques and methods to detect IoT malwares and malicious files. Abawajy *et al*. (2018) discussed different malware techniques and characteristics that can be used to create a detection module. It combines static and dynamic analysis capabilities in order to detect android-based mobile malwares.

In addition, others have used blockchain technology to solve digital forensics challenges such as: Evidence alteration/deletion or data integrity. Pourvahab and Ekbatanifard (2019), have used blockchain attack detection technique through "Chain of Custody (CoC)". Moreover, (Quick and Choo, 2018) discussed the solutions to address two main issues: The growing volume of data of IoT devices and the different data formats/structures of IoT devices. They used bulk digital forensic data analysis to extract the needed features to differentiate between different IoT devices and activities in a timely manner.

Furthermore, (Al-Sadi *et al*., 2018) described the different phases of Digital Forensic Investigation (DFI) process. There are three layers of IoT forensic framework as follows: (1) Top layer: IoT application server. (2) Middle/Second layer: Network layer which provides communication between the end user and the top layer. (3) Third layer: IoT device layer which contains a collection of IoT devices.

Moreover, (Alasmary *et al*., 2019) were able to differentiate between IoT and modern Android malwares through a graph-based analysis detection model. Another study was conducted by (Visu *et al*., 2019) which analyzed and detected IoT malwares by exe image visualization. This technique compared malicious and non-malicious files using random forest and decision tree methods.

Additionally, (MacDermott *et al*., 2018) have explained the roles of computers in cybercrimes and the different challenges that digital investigators may face in the scenes of Internet of Anything (IoA) crimes. These challenges include: Object size, possible connections to other local/non-local devices, the relevancies between collected devices, unclear network boundaries and legal issues. Furthermore, (Namanya *et al*., 2020) provided an accurate hash-based scoring approach that can be used to detect malicious Windows Portable Executable (PE) files.

In addition, using YARA rules to differentiate and find the similarities between different malware variants has been tested and proved by many authors. Hou *et al*. (2019) have utilized YARA rules to detect specific types of Malware (WannaCry Ransomware) on the scanned system and provided great results on catching all malicious files that had the same functionality/characteristics. Similarly, (Naik *et al*., 2019) have used different techniques (such as: Fuzzy hashing, import hashing and YARA rules) to test four pertinent ransomware categories: WannaCry, Locky, Cerber and CryptoWall. Based on the findings, YARA has provided the second most accurate results. On the other hand, the authors did not mention which YARA rules were used and what is the most important factor in YARA that could significantly change the results. Therefore, more advanced and accurate YARA rules could lead to better results.

Proposed Methodology

In this section, the proposed methodology will be represented and discussed in detail. The proposed methodology utilizes LOKI scanner as a scanning agent in order to scan every stored file. It looks for any malicious or suspicious string in these files by utilizing defined YARA rules. It detects any malicious file (malware) that has been used by attacker/s for any malicious activity on the victim's machine. Once a malware file is detected, all strings can be extracted and then added to the signatures' database. Later, it will be triggered whenever a file that contains similar strings is scanned.

Even though numerous papers have discussed different malwares detection techniques, still according to our best knowledge there is no one found solution to combine accuracy with fast scanning capabilities. Malware detection in IoT devices demands more requirements as they can be used in any operating system.

Scanning Agent

Scanning agents can be used to detect malicious files by searching for strings that are known to exist in malwares. There are four scanning agents developed by "Nextron Systems": THOR, THOT-Lite, SPARK/SPARK-CORE and LOKI:

a. **THOR:** is an enterprise product that supports all the major platforms (Linux, Windows and macOS). It was written in GO language, THOR is a powerful but a heavy tool that has a size of 16 MB, however THOR is inflexible since the default rules are encrypted and cannot be viewed or modified
b. **THOR Lite:** A free (Registration required) version of THOR that has the same size (16 MB) but with less features and therefore less efficiency
c. **SPARK/SPARK-CORE:** A lighter tool (9 MB) that also supports (Linux, Windows and macOS). Since 2019, it has been fully integrated into THOR

d. **LOKI:** A powerful python-based open source and free tool that has many extra features and comes in a lighter size (8 MB) and it is much more flexible and can be customized based on the user's needs

LOKI scanner was chosen as a scanning agent due to three main reasons:

- It is the only open source solution
- It is lighter than the other alternatives
- It is much simpler to use

LOKI can scan certain characteristics in all files. It provides the capability to scan systems through four different methods:

1. File path and name, such as: C:\Inetpub\wwwroot\shell[.]aspx
2. Hashes check, using SHA1, SHA256 or MD5 hashes
3. Endpoints process connections to C2
4. YARA Rules

Rule-Based Approach

YARA rule defines certain malware patterns/strings for malware detection. In this study, YARA rules along with LOKI scanner are used to detect malicious and suspicious files.

YARA rule approach was chosen due to its many advantages as follows:

- Deep investigation for every stored file
- Not limited to only EXE files
- Fast scanning capabilities
- Results can be saved for further investigations and review
- Easy to add, modify or delete YARA rules
- Can be used to scan both online systems and offline disk images
- Very flexible and customizable to use

A YARA rule has four mandatory fields that are written as one part. A simple YARA rule is shown in Table 1.

Table 1: Simple YARA rule and its mandatory fields

```
1  rule Rule_name: {
2  meta: description = "information about the rule"
3  strings:
       $c1 = "the first string to search for"
       $c2 = "the second string to search for"
4  condition:
       $c1 or $c2 }
```

These fields are as follows:

- **Rule**: Preferably a meaningful name that represents the rule name. It is composed of letters, numbers and special characters
- **Meta**: It can be used to add optional information such as: The author name, date-created, date-modified and rule's description
- **Strings**: It includes one or more suspicious indicators that could lead to malicious activities detection. Different types of indictors and strings can be added such as: Command and Control (C&C), domain/IPs, hashes or malicious functions that known to be used by attackers
- **Condition**: It uses Boolean condition in order to specify and classify the file as a malicious one

Even though, LOKI scanner comes with a database of thousands of default YARA rules that can detect common malicious files that have been used by attackers, it has some disadvantages as follows:

- It checks every stored file and compares it with a variety of different malwares' hashes/signatures, which might result in false positive triggers
- It cannot detect new malwares and malicious files that have not been detected before by known security solutions. As a result, it cannot detect malwares that were created and developed to target specific victims

Therefore, in order to increase the efficiency of our detection technique, new YARA rules were created for any new detected malware and were added to the default rules database. This step can be achieved with the help of "YARAGen", which is an open source tool that can be used to easily identify and scan malicious files (malware samples) and extract the needed strings to include it in YARA rule format. This process can add much value to the overall process since large organizations, that are more targeted by attackers and advanced persistent threats "APTs", usually find new (unreported) malwares in their environments.

The complete cycle of the scanning process is shown in Fig. 1, which includes the following:

- Connect the scanning machine to the same network where suspected machine resides
- Run 'command prompt' as administrator (or 'sudo' in Linux)
- Move to directory where LOKI.exe resides
- Run LOKI scan and specify the IP of the targeted machine to be scanned (it could be used to scan only specific directory on the scanned machine)
- Analyze the scan output

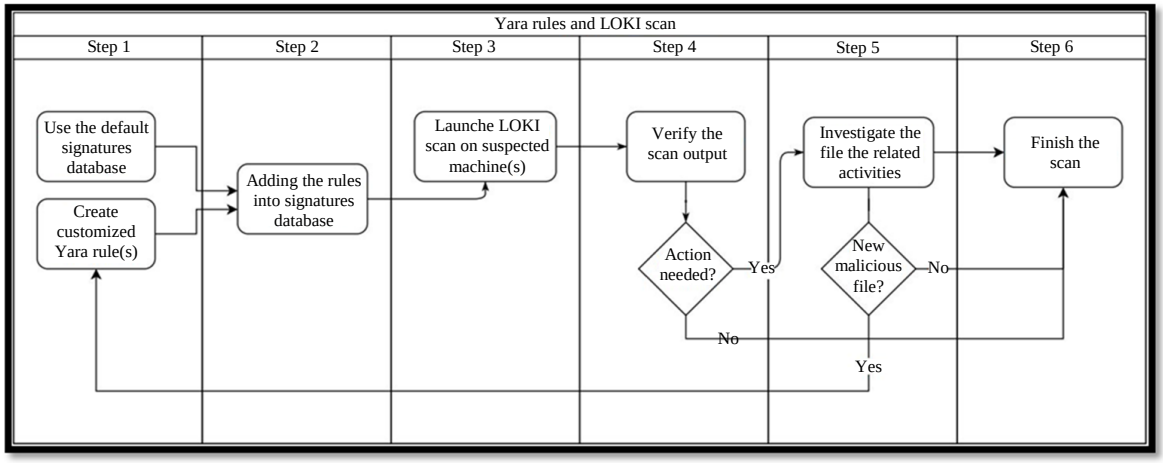

Fig. 1: Procedures to run LOKI scan

Main Methodology

Step1: Start
Step2: Build the environment
Step3: Simplify the environment
Step4: Is the environment simple enough?
{If Yes, then go to step5
If No, then go to step3}
Step5: Search through the Internet for an appropriate dataset
Step6: Check, test and analyze the dataset
Step7: Is it the needed dataset?
{If Yes, then go to step8
If No, then go to step5}
Step8: Combine the dataset with self-generated data
Step9: Install LOKI and test its capabilities with the default signatures/rules
Step10: Determine LOKI's limitations
Step11: Understand the syntax of YARA rules
Step12: Analyze all possible outputs/results
Step13: Create customized YARA rules
Step14: Test the depth of LOKI scanner through the created YARA rules
Step15: Stop

Fig. 2: Main methodology for the detection of malicious files

Figure 2 represents the main methodology used for the detection of malicious files using the customized rule-based approach.

Experimental Setup and Design

Hardware Requirements

Both LOKI and YARA rules do not need high requirements to run the scans. LOKI scanner was tested on different OSs using a large database of signatures and YARA rules with big files' size as shown in Table 2.

Data Collection

The dataset used in this study has been acquired from "M57 Corpus" (Horsman, 2019). It contains forensic images of different types of machines/devices and it has been combined with self-generated data from fresh installations of Windows and Linux OSs. This dataset was provided by "Digital Corpora" organization, which delivers different types and formats of data that can be used for testing and education proposes. In addition, it uses real devices which simulates realistic data.

Table 2: Details of the used scanning system

CLASS	Description
LOKI Version	LOKI 0.30.5
Operating System	Windows 10
RAM	16 GB
Processor	Intel i7 (7th generation)
Used command	Loki.exe -p --norpocscan
Number of scanned files	417,741 files
Number of scanned folders	72,052
Total size of scanned files	514 GB
Number of used C2 indicators	33578 elements
Number of used malicious MD5 Hashes	19011 hashes
Number of Malicious SHA1 Hashes	7100 hashes
Number of Malicious SHA256 Hashes	22732 hashes
Number of Used YARA rules	451 moderate/complex rules

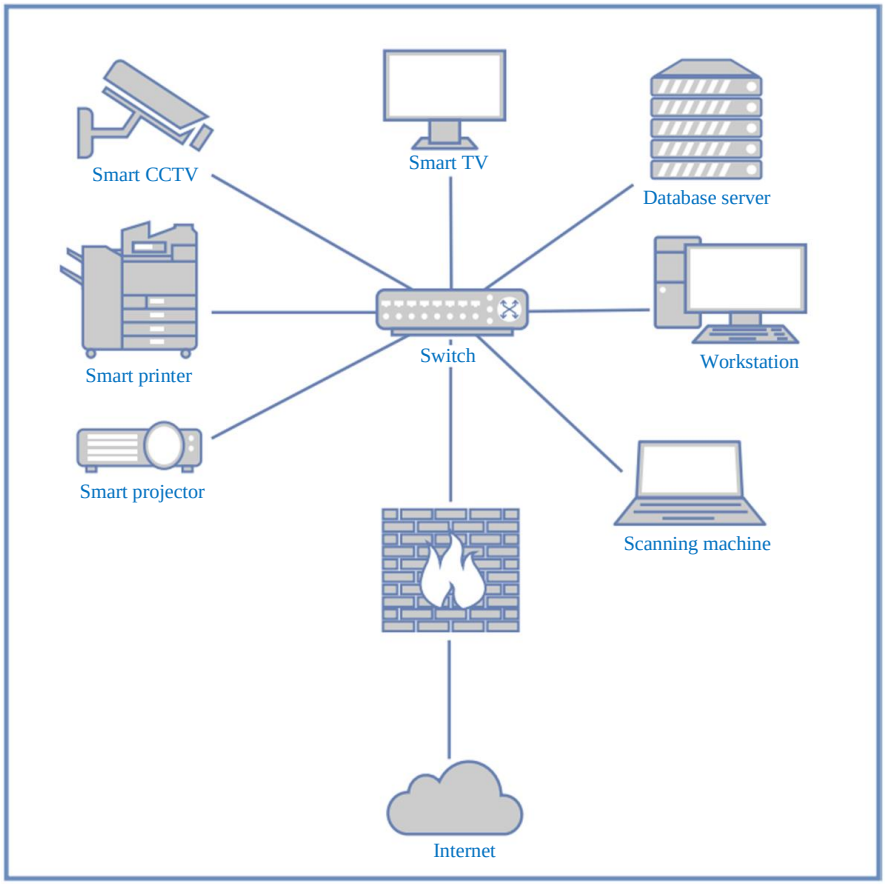

Fig. 3: Network environment used for LOKI scan

Network Environment and Scanning Process

In our environment, a typical network with small number of different devices was used. It included CCTV, smart printers, servers and Windows and Linux machines to simplify the analysis process. We believe that the obtained results will be the same for any type of environment. Figure 3 represents the network environment used for testing our detection technique.

Implementation

In the implementation phase, the proposed methodology was tested in three different complexity levels as follows:

a) Simple YARA Rule

The YARA rule has one or more strings. The condition should be easy to read by including “and”, “or” or any

simple Boolean expression as illustrated in Fig. 4. The illustrated YARA rule called "simple_rule" will simply search for any of the two strings string1 (known malicious IP "185.244.217[.]126") or string2 (MD5 hash value of Mirai malware) in every scanned file.

b) Moderate YARA Rule

In this complexity level, the rules are more complex, where it may include more specifications to "meta", "strings" and "conditions" fields, in order to make it more useful and accurate:

- **Meta**: A score value can be added to the rule which ranges between 40 to 100, where 100 indicates that the rule has the highest level of accuracy and 40 is the lowest which is used only in generic rules
- **Strings**: In addition to the text and hexadecimal representations, we can be more specific to search for the needed strings by using regular expressions to catch the targeted strings
- **Condition**: YARA made writing/reading of conditions much easier by allowing the use of "all" and "any" keywords:
    - "Any of them": To raise a trigger when any defined string is found
    - "All of them": To raise a trigger when all defined strings are found

Figure 5 has included hexadecimal string and regular expression string that can be used to search in all system's files for any 7Z file and MD5 values, respectively. Therefore, we have scored the rule with (50) since it is very generic and can provide false positive results. In the condition section, "any of them" was used to raise a trigger in case any of the mentioned strings were found.

c) Complex YARA Rule

There are many more features that can be used to make the rules more accurate and have deeper investigations capabilities as illustrated in Fig. 6.

The rule in Fig. 6 can be used to detect any PDF, MZ (DOS executable) or PNG file that contains a known malicious function "ActiveXObject". Furthermore, the magic number (file signature) was used to specify the file type. In the condition field, files larger than 200 KB were targeted to reduce the false positive rate) which must contain "ActiveXObject" text in one of the following file types PDF, MZ executables or PNG.

For the purpose of validating and checking the efficiency of our methodology, three files were added into the scanned system that should be detected as follows:

- "Web Shell" file, is a type of malicious script that is known and used by attackers to provide an authorized access to the targeted system/machine and create persistent backdoor
- "AnyDesk" executable file, is a powerful and very popular remote desktop tool that enables its users to do almost anything in the system without the need to be around. AnyDesk requires "Administrator privileges". With its versatile features, it became commonly used by attackers as well
- Text file which contains a hash value of sample of the "Monero Cryptocurrency Mining" malware

```
rule simple_rule {
 meta:
 description = "known malicious IP and MD5 hash of IoT malware"
 strings:
 $string1 = "185.244.217.126"
 $string2 = "9110C043A7A6526D527B675B4C50319C3C5F5C60F98CE8426C66A0A103867E4E"
 condition:
 $string1 or $string2}
```

Fig. 4: Simple YARA Rule

```
rule moderate_rule {
meta:
 description = "EXE files or MD5 values"
 score = 50
strings:
 $string1 = {37 7A BC AF 27 1C}
 //hexadecimal value of 7Z (7-zip) compressed files
 $string2 = /[0-9a-fA-F]{32}/
 //regular expression for any MD5 values
condition:
 any of them}
```

Fig. 5: Moderate YARA rule

```
rule complex_rule {
 meta:
 description = "hidden ActiveX Object"
 score = 100
strings:
 $string1 = "ActiveXObject" nocase
 $string2 = {25 50 44 46 2d}
 //File Signature of PDF file type
 $string3 = {4D 5A}
 //File Signature of MZ file type
 $string4 = {89 50 4E 47 0D 0A 1A 0A}
 //File Signature of PNG file type
condition:
 Filesize > 200KB and $string1 and 1 of ($string2, $string3, $string4) }
```

Fig. 6: Complex YARA rule

```
rule Monero_miner_rule {
meta:
 description = "Monero Cryptocurrency Malware Hash"
 score=60
strings:
 $s1                                                                =
"0E1F82AC5ACCA3F826A2E5D9B5A3BA43431990AA0D
0165C88AC5E0C7C84232ED"
condition:
 $s1 }

rule Anydesk_rule {
meta:
 description = "AnyDesk executable"
 score = 80
strings:
 $s1 = "Anydesk" nocase
 $s2 = "This program cannot be run in DOS mode" nocase
 $s3 = "philandro Software GmbH" nocase
condition:
 all of them }

rule Webshell_rule {
meta:
 description = "Malicious PHP Webshell"
 score=100
strings:
 $s1 = "post" nocase
 $s2 = "get" nocase
 $s3 = "cmd" nocase
 $s4 = "file" nocase
 $s5 = "execute"
condition:
 all of them}
```

Fig. 7: Used YARA rules to detect three different files

Consequently, three YARA rules were created and added to the rules' database before launching the scan (".yar" into YARA directory "loki\signature-base\YARA\"). The newly added YARA rules are shown in Fig. 7.

Results and Discussion

Table 3 shows the start and end time of the scan and the time taken to scan each file and each MB, however the time taken for the scanning process was not long.

In addition, the output of the scan shows a number of detected files that have suspicious strings that matched YARA rule database. The output has different classifications for triggers highlighted with different colors.

Based on how YARA rules were written, each event/trigger of the scan results can be in one of four possible types and each type will be displayed in a unique color with different meaning, as shown in Fig. 8. The most important triggers that need to be analyzed are "ALERT" and "WARNING", respectively, since others show only the used configurations and events with lowest level of risks.

By analyzing the three triggers in Fig. 8, it can be concluded that "AnyDesk" is a known remote desktop tool that is used by both organizations and attackers. However, the file was not downloaded by the system users. As a result, it was used by the attacker since it is located in the "temp" folder which is a common place to find malwares. the "webshell.php" file which has the highest score value is clearly a malicious file since it contains many commands/functions that enable the attacker to perform many malicious activities and it is also located in the "temp" folder. The last file is a text file that contains a hash value of a known malware "Monero Cryptocurrency Mining" and is located in the "Recycle Bin" which is also a common place that attackers use to hide their malicious files.

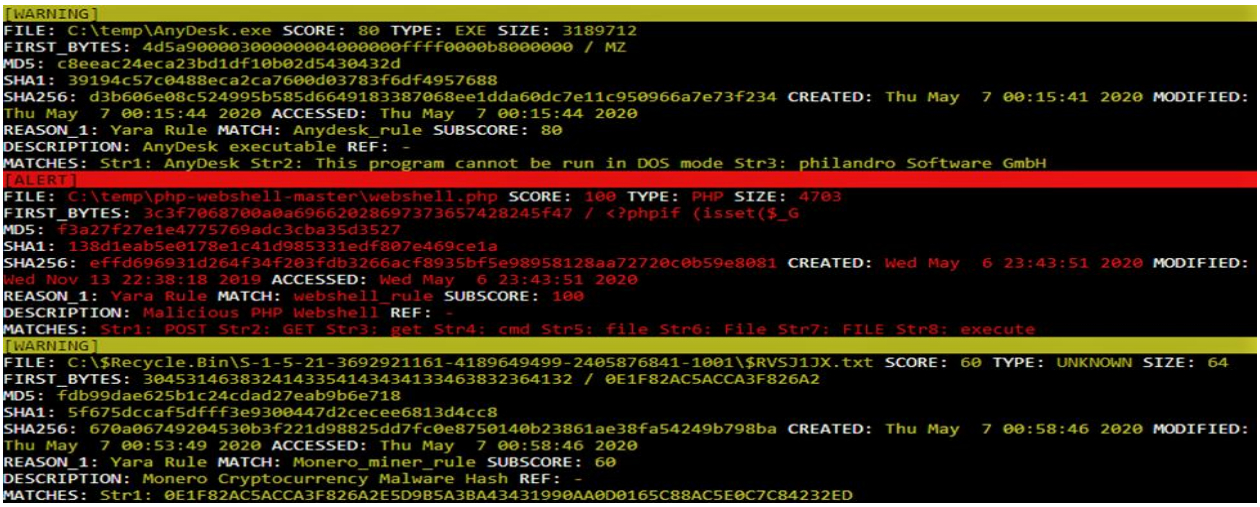

Fig. 8: LOKI scan's output

Table 3: Details of the scan results

Class	Description
Scan starting time	2020/04/24-23:47:18
Scan ending time	2020/04/25-04:25:28
Time taken to finish the scan	4 h, 38 min and 10 sec
Time taken to scan each file (approximately)	0.039 sec
Time taken to scan each MB (approximately)	0.032 Sec

Table 4: Comparison between typical Anti-Virus and the proposed methodology

	Antivirus	Proposed Methodology
Main Usage	Regulated scanning to detect common/ known malware's activities/files.	Flexible way to scan any type of malicious strings (hashes, commands, file types…etc.) or indicators of compromise.
Flexibility	Usually there is no option to add or delete signatures.	Easy to add, delete or modify YARA rules.
Hardware Requirements	Usually consumes lot of system resources.	Does not require high system performance
Deep Investigations	Besides processes, AV scan files only to look for certain scripts, headers or hashes.	Provides deep file analysis to search for any suspicious strings.
Trigger Classifications	Most AVs have only two options: Either the file is malicious or not	Has 5 levels of classifications
Supported File types	AV usually scans only certain types of files.	Has the ability to scan every type of files in the system.
Ability to search	Cannot be used to search for specific malicious strings or Indicator of Compromise (IoC)	Easy to search for any type of string in every file in the system.

Table 4 compares the proposed methodology with a normal Anti-virus. As indicated in Table 4, normal antiviruses have several limitations, such as: (1) Inflexiblity as the user can only use the software's (fixed) database of signatures that cannot be modified to include customized signatures, (2) Since the signatures' database was created to be used by all customers around the world, it is usually very genetic and contains only known malicious hashes, signatures and functions, (3) Antiviruses are known to be heavy applications and they require high system resources, which could interrupt and affect the business processes.

Based on the findings, all suspicious files have been detected and analyzed successfully using the proposed methodology with an increased efficiency.

Limitations and Recommendations

As we highlighted the urgent need in the cyber world for LOKI scanner which has many great capabilities that make it a proper and more powerful solution than other available techniques, there is one issue of using LOKI to scan for IoT malicious files which is the lack of ready-to-use databases of YARA rules that contain only IoT-related rules, therefore we recommend as a future research to create and collect a big database of YARA rules that contains all different types of IoT malware's strings to make the detection model much more effective and focused on IoT devices. Another recommendation is to use the great functionality of LOKI scanner to not only scan the stored files, but also to scan all running

processes on the scanned system to provide more in-depth analysis, however this step will increase the needed time to finish the scan.

Conclusion

There is an urgent need for a new detection technique that has the accuracy, customizability and efficiency to scan suspected systems in a less time and with the minimum usage of system resources. The use of LOKI scanner instead of typical scanning techniques provides different features and advantages that makes it a proper solution to search and detect stored malicious files. In addition, YARA rules, if properly written, have the flexibility to focus only on malwares that target IoT devices. The proposed methodology focuses on using LOKI as a scanning agent and customized YARA rules to increase the efficiency of the detection process.

Acknowledgment

The authors of this manuscript would like to express their appreciations and gratitude to their universities for supporting this research.

The authors would like to thank editors for their efforts in handling the manuscript and all reviewers for the constructive comments which improved the original submission.

Author's Contributions

Faisal Alsattam: Acquisition of data, investigation, software, original draft preparation, approved the version to be submitted and any revised version.

Mousa Al-Akhras: Conceptualization, design, investigation, analysis and interpretation of data, review and editing, approved the version to be submitted and any revised version.

Marwah M. Almasri: Conceptualization, investigation, methodology, analysis and interpretation of data, review & editing, approved the version to be submitted and any revised version.

Mohammed Alawairdhi: Conceptualization, methodology, design, original draft preparation, approved the version to be submitted and any revised version.

All authors have read and approved the final manuscript.

Ethics

This paper is original and innovative and contains unpublished material. There are no ethical issues involved and all authors have no conflicts of interest to release.

References

Abawajy, J., Huda, S., Sharmeen, S., Hassan, M. M., & Almogren, A. (2018). Identifying cyber threats to mobile-IoT applications in edge computing paradigm. Future Generation Computer Systems, 89, 525-538.

Al-Sadi, M. B., Chen, L., & Haddad, R. J. (2018, April). Internet of Things digital forensic investigation using open source gears. In SoutheastCon 2018 (pp. 1-5). IEEE.

Alasmary, H., Khormali, A., Anwar, A., Park, J., Choi, J., Abusnaina, A., ... & Mohaisen, A. (2019). Analyzing and detecting emerging internet of things malware: A graph-based approach. IEEE Internet of Things Journal, 6(5), 8977-8988.

Demeter, D., Preuss, M., & Shmelev, Y. (2019). IoT: a malware story. AO Kaspersky Lab. https://securelist.com/iot-a-malware-story/94451/

Horsman, G. (2019). Tool testing and reliability issues in the field of digital forensics. Digital Investigation, 28, 163-175.

Hou, J., Li, Y., Yu, J., & Shi, W. (2019). A Survey on Digital Forensics in Internet of Things. IEEE Internet of Things Journal, 7(1), 1-15.

MacDermott, A., Baker, T., & Shi, Q. (2018, February). Iot forensics: Challenges for the ioa era. In 2018 9th IFIP International Conference on New Technologies, Mobility and Security (NTMS) (pp. 1-5). IEEE.

Naik, N., Jenkins, P., Savage, N., & Yang, L. (2019, June). Cyberthreat hunting-part 1: triaging ransomware using fuzzy hashing, import hashing and YARA rules. In 2019 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE) (pp. 1-6). IEEE.

Namanya, A. P., Awan, I. U., Disso, J. P., & Younas, M. (2020). Similarity hash based scoring of portable executable files for efficient malware detection in IoT. Future Generation Computer Systems, 110, 824-832.

Pourvahab, M., & Ekbatanifard, G. (2019). An efficient forensics architecture in software-defined networking-IoT using blockchain technology. IEEE Access, 7, 99573-99588.

Quick, D., & Choo, K. K. R. (2018). IoT device forensics and data reduction. IEEE Access, 6, 47566-47574.

Visu, P., Lakshmanan, L., Murugananthan, V., & Cruz, M. V. (2019). Software-defined forensic framework for malware disaster management in internet of thing devices for extreme surveillance. Computer Communications, 147, 14-20.