IJCSI International Journal of Computer Science Issues, Volume 16, Issue 3, May 2019
ISSN (Print): 1694-0814 | ISSN (Online): 1694-0784
www.IJCSI.org https://doi.org/10.5281/zenodo.3252963

Static Malware Analysis to Identify Ransomware Properties

Deepti Vidyarthi[1], CRS Kumar[2], Subrata Rakshit[3,] and Shailesh Chansarkar[4]

**[1,2] Department of Computer Science & Engineering, Defence Institute of Advanced Technology
Pune, Maharashtra, India**

**[3, 4] Center of Artificial Intelligence & Robotics, DRDO
Bangalore, Karnataka, India**

Abstract

The study in this paper presents the results of ransomware analysis to identify the characteristic properties that distinguish ransomware executable from other malware and benign executables. The executables are analyzed both statically and dynamically to observe the typical structure and behavior of the ransomware. Static analysis is used to extract ransomware specific properties from the executable file. The inclusion of these properties with the set of generic malware properties has shown improved classification for malware detection specifically ransomware using well-proven software verification and validation techniques. Runtime analysis identifies the most common behavior of the ransomware. The paper is focused upon the ransomware attacks which have risen exponentially over the past decade with increasing severity. This has the conventional anti-malware techniques compelled to include advanced detection mechanisms for ransomware and this paper demonstrates a method to identify the properties related to such software environments.

Keywords: *Ransomware, malware detection, static analysis, dynamic analysis.*

1. Introduction

A ransomware attack has a huge potential to damage and it is a fast-growing malware, the major reason being it is designed to earn money from the victims. New types of ransomware are being constantly added to the family. McAfee Mobile Threat Report 2019 [1] notifies the increase in "mobile ransomware". As predicted by Barkly Endpoint Security, ransomware continued to experience record growth in 2017-18. A new target organization is facing a ransomware attack at every 40 seconds [2]. According to Sophos labs forecast 2018, ransomware is now being used and provided as a service in the form of Ransomware-as-a-service (Raas) [3].

According to the Ransomware Statistics in 2019 (Infographic), ransomware attacker gains access to victims system through: phishing via email or social media, drive-by download, click on a compromised website, infection from computer part of the botnet [4]. The most prevalent modus operandi is as follows. Ransomware reaches the target computer as an email, using social engineering techniques and activates after the victim downloads the email attachments or clicks on insecure links from the email. As it spreads over the network easily, the number of victims exposed to this threat is very high. It can encrypt, delete or modify any data on the affected machine. It also can lock the whole system. The ransomware encrypts the most commonly used files in the target computer, thereby forcing the individual users or institutions to pay a ransom for decryption of their own data. The attackers prefer Bitcoin as a means for payment, as it is an anonymous payment mechanism and conceals their identity and location. These peculiarities encourage attackers to choose ransomware in their malicious attacks out of the other malware. With the above-mentioned threat of ransomware and its damaging potential in sight, it is mandatory to analyze the ransomware beyond the general analysis conducted on malware or just signature-based detection. There is a need to evolve an advanced ransomware detection approach to complement positively to contemporary anti-malware techniques. Detection and prevention techniques for ransomware are being proposed through the literature, however, there is a scope for them to become sound and well established. For any detection mechanism, the core component is to identify the characteristics that would serve as a unique reference.

This paper focuses on the identification of such characteristics of the Portable Executable (PE) file that can be used effectively in ransomware detection. Further, this paper brings out the improvement in general malware classification brought about due to the incorporation of the identified characteristics, serving as an empirical proof that such an approach complements positively to the contemporary anti-malware techniques.

2. Ransomware

Malware analysis is the study of malware by dissecting its different components and studying its behavior and effects on the computer system.

2.1 Ransomware Working

The general mechanism of ransomware operations [5] can be described in the following five stages:

Stage 1: Infecting the computer by the execution of the malicious ransomware file is the first step. It can be through a phishing email or an exploit kit. E.g. Angler exploit kit, (CryptoLocker) exploits the bugs in Adobe Flash and IE.

Stage 2: Delivery of the actual ransomware executable through the exploit is the next step. The executable is run on the system and made as a persistent process.

Stage 3: Destroying Back-up is a unique feature of ransomware. The back-up files and folders on the system are targeted and removed to prevent system recovery on its own. E.g. vssadmintool to remove the volume shadow copies from the system.

Stage 4: Strong encryption algorithms like AES 256 are used for encrypting the targeted files/directories, etc. Secure key exchange is performed with the C & C server. The scope of encryption is different for different variants. E.g. CryptoWallv3 doesn't encrypt filename, CryptoWallv4 randomizes filename.

Stage 5: Ransom Demand is the final stage, the instructions for extortion and payment are saved onto the hard drive. The victim is given a few days to pay and after that time the ransom increases. Malware removes its traces

2.2 Ransomeware Properties

Ransomware is a particular type of malware with a specific mechanism and set of properties. It locks the system or encrypts the data facilitating the attacker towards demanding a desired amount of money as 'ransom', in return for the decryption key. Generally, ransomware are of two types, one of which and most popular these days is 'Crypto ransomware' that can encrypt the whole data on the victim's machine, making it inaccessible to the victim. The other kind of ransomware is 'Locker ransomware' that can lock the whole system so that the victim is unable to use it. The commonality between both types of ransomware is the ransom demand for releasing the assets of the victim. The general operational architecture of ransomware is as follows. It mainly consists of a 'dropper' that contains the encrypter [6]. The encrypter component contains a decryption engine and a password protected compressed component (zip), containing a copy of Tor [7] and several individual files with other information and the encryption key. First, the dropper tries to connect to a remote website say "xyz.etc" and exit after connection. If the connection fails in the first step, it then creates a service for example "mssecsvc2" with another name [7]. After creating a service the dropper extracts the encrypter binary from its resource and then executes it. Encrypter also checks for the presence of the component called mutex. The encrypter creates the "mutex" if not present, before execution. In another approach, the ransomware shares its encryption key using command and control server (C & C Server) [8]. In the case where the C & C server is not present, it uses hard-coded keys and uses the same key for all encryption [8].

3. Related Work

3.1 Malware Analysis Techniques

Multiple malware analysis techniques have been proposed for malware detection and classification in general by various researchers over the years. Broadly, malware analysis methods can be classified into static analysis and dynamic analysis. The detection techniques can be characterized in the terms of executable's features used for detection and their classification approach [9]. In our previous work [10], Random Forest, Decision Trees (J-48), Naive Bayes and Support Vector Machine (SVM) classifiers were trained towards malware classification based on the features extracted through static and dynamic analysis of the malware executable in general.

3.2 Ransomware Analysis

The majority of literature and work related to ransomware focuses on the encryption and deletion mechanisms and communication techniques. Analysis of the file system activities of ransomware samples suggested that it is possible to detect ransomware by looking at I/O request and Master File table in NFTS file system [11]. Another approach for detection of ransomware using C & C server DNS logs is discussed [8]. Here, the ransomware uses DGA algorithm to generate random fake domain name by detecting the encryption key while communication with command and control server, so that the detection of the valid or invalid domain name is difficult. Cryptolocker, WannaCry, TeslaCrypt use DGA for generating domain name. Kramer and Bradfield [12] suggest a continuous monitoring approach for ransomware detection that includes - maintaining the ransomware signature and Indicators of Compromise (IOC), looking for file execution from %APPDATA% folder and %TEMP% folder, monitoring back-up files, checking file extensions, observing the anomalous network behaviour during key exchange and looking at I/O requests and Master File Table (MFT) in NTFS file Detection. Brewer [5] proposes an automated approach to track the changes to the system's desktop that indicate ransomware-like behavior. The above-mentioned work is observed to be focused on analyzing the typical working mechanism of ransomware. K. P. Subedi et. al. [13], performed static and dynamic analysis and concentrated on developing signatures by reverse engineering. Their signature database is specific to crypto ransomware. A study proposed by Zimba et.al. [14] discusses the attack model of ransomware and its techniques are extracted through the static analysis. The study of ransomware variants, BitPaymer and KeyPass, details the behavior of these ransomware and identifies the presence through code analysis [15]. Zavarsky et.al. [16] present the experimental analysis of ransomware on windows and

android platform and recognize the main behavioral properties. Through the study of existing approaches, we design methods to analyze the ransomware samples for identification of its typical properties.

4. Proposed Work: Identification of Ransomware Specific Properties

In this paper, we focus on identifying approaches to analyze ransomware. The proposed method is based on the observations that the static and dynamic analysis of ransomware specifically would result in the features unique to ransomware. These newly extracted features that are appended to the features used for the general malware classification as performed on file size, entropy, timestamp, size of initialized and uninitialized data, entrypoint address, etc [6], demonstrate the improvement in the performance of classification.

Motivation: The ransomware behavior discussed through the literature is studied and the properties of ransomware executable which are responsible for the typical behavior are observed by static and run-time analysis of ransomware. Such distinguishing characteristics of ransomware termed as the ransomware specific properties are proposed to improve the general malware classification by training the classifier based on the combination of ransomware specific and other generic malware properties. The study is motivated to experiment with the classification using three algorithms, Random Forest, Decision Trees (J-48), and Naïve Bayes. These algorithms are selected based upon their suitability and better performance for general malware classification.

Aim: The proposed approach aims to identify the characteristic properties of ransomware for classification of the executables. The activities performed by malware are realized through different sections of the PE file. Our approach is to learn and identify the typical characteristics of ransomware through static and run-time analysis of the ransomware PE file and increase the identification parameters.

4.1 Method to identify properties through static analysis of PE

The static analysis consists of examining the executable file without viewing the actual instructions. The PE file is dumped and disassembled to learn the structure and components of the file [9,10]. The method of extracting the general malware and ransomware specific properties from a PE through static analysis is given in 6 steps below:

Step 1: PE file is dumped and the overall structure is checked through the header information.
Step 2: The number of sections, names, etc. are scanned using PE readers like PE explorer, PEid, etc. to check if the file is packed or not [20].

Step 3: If the file is packed then identify the packer and run the appropriate unpacker to extract the original file.

Step 4: Read the original file to extract the metadata from the header fields. All the DOS, File and optional headers are read to extract field values. These values are used for differentiating the general malware from benign executable files.

Step 5: The ransomware working is studied as discussed in section 2 to observe the specific ransomware characteristics.

Step 6: Based on the observed characteristics, various fields from PE file sections are identified. The related field values with reference to the observed properties are used to differentiate ransomware from the benign files.

As derived from the steps above, a malicious file has distinctive values for various header fields. For example, the number of sections is not as in regular PE file, the import section might have certain typical dynamic-link libraries (DLLs) or application programming interface (API), etc. The general malware properties are extracted in step four of the flow and the last two steps extract the ransomware specific properties.

4.2 The Static Parameters Identified

For identifying the static properties for general malware detection, header analysis is carried out. It includes the analysis of three kinds of headers, DOS header, file header and optional headers, which contain metadata at different levels. We propose to analyze all the header sections of an executable and extract the features as described in the next section. Structural analysis is carried out at various sections of the executable image. It gives information about packer's identification, and entropy of sections like - entropy for file, data and text section. The import section of an executable file contains names of the imported DLLs and API functions. Import directory of ransomware PE is analyzed to note the particular DLLs used by ransomware that are uncommon in other software. The string section of PE contains strings used in executable – commands, dialogues, function names, etc. Strings are analyzed to identify suspicious commands and function used by ransomware to perform various malicious activities. The static properties used in this work to detect the general malware are generated using the static analysis of PE file headers. Totally 60 static properties are identified. The major properties for general malware detection and their significance are given in Table 1.

Table 1: Properties read from PE headers

Field	Header	Significance
e_cblp	DOS header	Bytes on last page of file
e_crlc	DOS header	Pages in file
e_cparhdr	DOS header	Relocations
e_lfanew	DOS header	4-byte offset into the file where the PE file header is located
NumberOfSections	File header	Number of section headers and section bodies in the file
NumberOfSymbols	File header	Symbols in the header
SizeOfOptionalHeader	File header	Size of optional header (optional header contains initial stack size, program entry point location, preferred base address, operating system version, etc.)
Characteristics	File header	Specific characteristics about the file
SizeOfCode	Optional header	Size of code section, in bytes, or sum of all such sections if multiple code sections.
SizeOfInitializedData	Optional header	Size of the initialized data section, in bytes, or the sum of all such sections if multiple initialized data sections
SizeOfUninitializedData	Optional header	Size of the uninitialized data section, in bytes, or the sum of all such sections if multiple uninitialized data sections.
AddressOfEntryPoint	Optional header	Location of the entry point for the application

Table 2: Ransomware specific properties from PE sections

Identified Property	Significance
Ispacker	Presence of packer
Packer type	Identity of type of packer
Open mutex Create mutex	Checking and creating mutex for isolation
Entropy (e_file, e_text, e_data)	Entropy of file, text and data sections
Strings	Presence of common strings extracted through ransomware string analysis; E.g. crypt/decrypt/%amount%)
Ws2_32.dll	DLL used for network connections and communication
Get_news	Command used to modify the registry
Add_entry	Store information from the client
Get_add	Get access to file from given path

The ransomware has several additional characteristics which may not be covered by these 60 static properties, thus extending the static properties aiming to cover all possible characteristics of ransomware becomes significant. Therefore, ransomware main characteristics such as packed code, packer identity, text section entropy, network connections/communication specific DLLs, mutex, and encryption related strings, etc. are considered to frame the sufficient additional properties. The identified additional 9 properties are given in Table 2. Static analysis is useful to extract the characteristics of PE file for ransomware detection as given above. However, in cases where complex obfuscation and packing techniques are used, simple static analysis may not be able to extract all the required features [17]. It is required to apply dynamic analysis technique as well to detect malicious functionalities of ransomware [18].

4.3 Method to identify properties through dynamic analysis of PE

Dynamic analysis techniques are used to observe the run-time behavior of the executable file and its effect on the system [9,19]. The flow of monitoring the ransomware execution is as given in Figure 2. It is proposed to monitor the run-time behavior using user-level and kernel-level debugging. While execution, the system may get locked (in case of locker ransomware) or the execution may proceed (if the executable has sensed debugger and used evasive technique). In the case, when the system is locked, the monitoring of process on the same system is not possible. In such case, the kernel-level debugging using another system machine is required.

Step 1: PE file is dumped and overall structure is checked through the header information.

Step 2: The number of sections, names, etc. are scanned using PE readers like PE explorer, PEid, etc. to check if the file is packed or not[20].

Step 3: If the file is packed then identify the packer and run the appropriate unpacker to extract the original file.

Step 4: An isolated runtime environment is prepared to execute the ransomware PE file. It can be either emulator, debugger or Virtual machine.

Step 5: Start the execution through the user-level debugger.
Monitor the activities through sys-internal tools.

Step 6: Study the mechanisms for different ransomware families to identify the commonly present activities, strings, and dlls, etc. during the debug-time.

Step7: Extract the debug-time properties based on the ransomware specific activities.

Step 8: If the system is locked, restart process debug and monitoring through the kernel-level debugger.

Ransomware of different types i.e. Crypto-ransomware, Locker-ransomware, a combination of crypto-locker, etc. are executed to observe their behavior in form of suspicious commands, registry changes, suspicious strings, file encryption, etc.

4.4 The Dynamic Behavior Identified

The dynamic analysis is performed over different types of ransomware downloaded from VXHeaven and GitHub [21,22]. The observations obtained from run-time / debug-time analysis for Crypto-Locker, Locky, and WannaCry ransomware samples are discussed in this section.

Observations: Initially, user-level debugging using OllyDbg [23] is done where the process level break-points are set to analyze the code and different modules of the code independently. The 'Locker-ransomware' is identified as running by changing its name to rundll32.exe. As the system might get locked during the course of execution, kernel-level debugger, WinDbg [24] is used for debugging ransomware affected system. The ransomware 'Crypto-locker' locks the system during the run-time analysis. Its running monitoring shows the presence of suspicious DLLs names like BCRYPT and related functions. The 'Locky' ransomware does not lock the system immediately, but shows the evasive behavior and uses another name for execution. The debugger could read multiple suspicious strings entries. The ransomware 'WannaCry' encrypts the files on the system and changes the registry values. The RSA1 encryption and suspicious strings are identified during the execution of 'WannaCry'.

The observations from multiple run/debug of all types of ransomware lead to the most commonly observed ransomware behaviors as listed below:

1. Presence of suspicious DLLs in the import section at run-time. Most common functions are to write /edit system files etc.
2. Change in windows registry.
3. Deletion and modification of windows directories.
4. Hiding the traces by using 'DestroyWindow' command and changing the process name.
5. Catching the encryption key used for encryption of the file during the ransomware attack.
6. Encryption techniques, such as RSA1.
7. Presence of the suspicious strings found at run-time.

As seen during the runtime analysis of locker type ransomware, it is not possible to monitor the true behavior of the process. This is the case with general evasive malware as well. Thus more accuracy is achieved using the static analysis method discussed in the previous section.

5. Classification Based on Static Properties for Ransomware Identification

The effect of combining ransomware specific properties with general static properties for malware classification is validated through the experiment performed over the data from 2488 benign samples and 2722 malware samples (a combination of ransomware and other malware). The sample set is used for training and classification of malware. The classification is done using three algorithms, Random Forest, Decision Trees (J-48), and Naive Bayes. These algorithms are selected based upon their suitability and better performance for general malware classification. Table 3 defines the evaluation metrics used to estimate the classification performance. Here, TP is True Positive i.e. malware correctly classified as malware; FP is False Positive i.e. benign incorrectly classified as malware; TN is True Negative i.e. benign correctly classified as benign; FN is False Negative i.e. malware incorrectly classified as benign.

Table 3: Evaluation Metrics to Measure Classification Performance

S. N	Measure	Definition	Desired effect
1	True Positive Rate (TPR) or Recall	$\frac{TP}{TP + FN}$	High
2	False Positive Rate (FPR) or Fall-out	$\frac{FP}{FP + TN}$	Low
3	Precision	$\frac{TP}{TP + FN}$	High
4	F-measure	$\frac{2TP}{2TP + FP + FN}$	High
5	Accuracy	$\frac{TP + TN}{TP + TN + FP + FN}$	High

As given in the table, for an acceptable classification, high values of TPR, Precision, F-measure, and Accuracy are desired; whereas, a low FPR is desired.

As the first part of the experiment, the static properties for general malware are used for malware classification. The result of classification using three algorithms; Random Forest, Decision Trees (J-48), and Naive Bayes are given in Table 4. The performance of the three algorithms is compared as shown in Fig 1. Based on these parameters, the random forest algorithm has shown the best performance with the accuracy of 98.34%.

Table 4: General Static Properties based Classification

Classification Algorithm	TPR (%)	FPR (%)	Precision (%)	Accuracy (%)
Naïve Bayes	62.0	34.9	74.4	62.789
J48	97.3	2.8	97.3	97.28
Random Forest	98.3	1.7	98.3	98.34

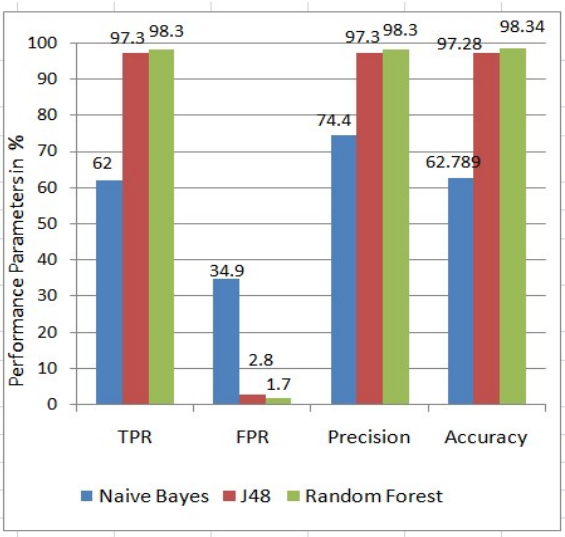

Fig. 1 Performance Comparison for three Classifiers based on General Static Properties

To check the significance of identified ransomware specific properties in the classification, the specific properties are added to the set of general properties for classifying the same set of malware including ransomware. The performance of three classifiers is as given in Table 5. It is observed that on the integrated data set, all the three classifiers have shown better results than on general feature data set. Also, as shown in Fig 2, considering the parameters, TPR, FPR, Precision, and Accuracy, the random forest has shown the best results with the accuracy of 99.25% among all three classifiers.

Table 5: Classification based on ransomware specific properties

Classification Algorithm	TPR (%)	FPR (%)	Precision (%)	Accuracy (%)
Naïve Bayes	65.2	31.9	78.3	65.20
J48	97.4	2.2	97.8	97.81
Random Forest	99.3	0.8	99.3	99.25

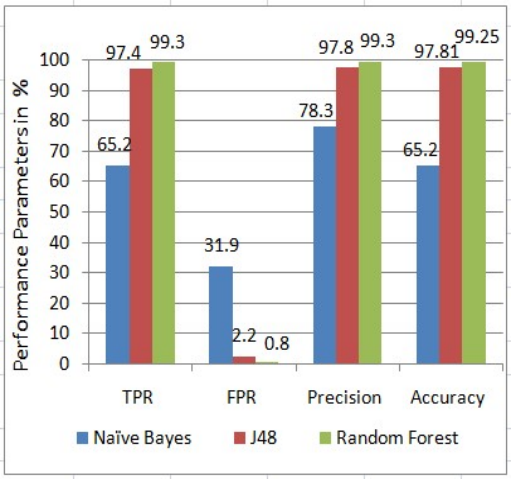

Fig. 2 Performance Comparison for three Classifiers based on Ransomware Specific Properties

As the Random Forest classifier has shown the best performance, in this case, the details of the classification using Random Forest are considered for observing the performance improvement by the proposed method. Table 6 gives the confusion matrix for classification result using Random forest based on generic malware properties. The detailed Accuracy measured by class is as given in Table 7.

Table 6: Confusion Matrix: Classification using Random forest based on generic malware properties

a	b	←Classified as
2443	58	**a = benign**
28	2655	**b = malware**

Table 7: Detailed Accuracy by Class: Classification using Random forest based on generic malware properties

TPR (%)	FPR (%)	Precision (%)	F-Measure (%)	Class
97.7	01.0	98.9	98.3	**Benign**
99.0	02.3	97.9	98.4	**Malware**
98.3	**01.7**	**98.3**	**98.3**	**Weighted Avg.**

Table 8 gives the confusion matrix for classification result using Random forest based on the proposed approach i.e. combining the ransomware specific properties with the generic properties. The detailed accuracy measured by class is as given in the Table 9.

Table 8: Confusion Matrix: Classification using Random forest based on ransomware specific properties

A	b	←Classified as
2458	30	**a = benign**
9	2713	**b = malware**

Table 9: Detailed Accuracy by Class: Classification using Random forest based on ransomware specific properties

TPR (%)	FPR (%)	Precision (%)	F-Measure (%)	Class
98.8	00.3	99.6	99.2	**Benign**
99.7	01.2	98.9	99.3	**Malware**
99.3	**00.8**	**99.3**	**99.3**	**Weighted Avg.**

The confusion matrix for both proposed and generic method show that the correctly classified instances are increased and the incorrectly classified instances are decreased. The effect is plotted in the Receiver Operating Characteristic (ROC) curves for classifications based on both set of properties as shown in Fig. 3. The Area under the ROC curve is computed as 0.999 for classification based on ransomware specific properties.

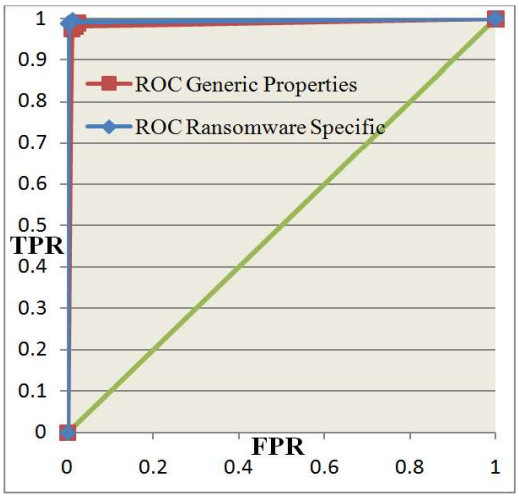

Fig. 3 Comparison: ROC Curves

The performance improvement is observed through the comparison of the classification based on the generic malware properties and based on additional ransomware specific properties using the detailed parameters derived in Table 7 and Table 9.

Fig. 4 and Fig. 5 show the class-wise comparison of TPR, FPR, Precision, and F-Measure for benign and malware classes respectively. The proposed approach has shown higher TPR, Precision, and F-Measure and lower value of FPR for both the classes.

Fig. 6 shows a similar comparison in terms of TPR, FPR, Precision, and F-Measure for the weighted average case. Overall, the increase in TPR, Precision, and F-Measure is observed. At the same time, the FPR is decreased.

The experimentation results illustrate that including the static properties identified in ransomware specific analysis with the general malware features can achieve higher accuracy in classification.

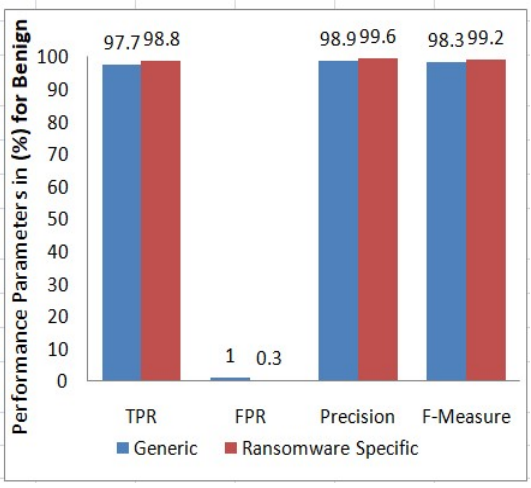

Fig. 4 Detailed Performance Comparison: Benign

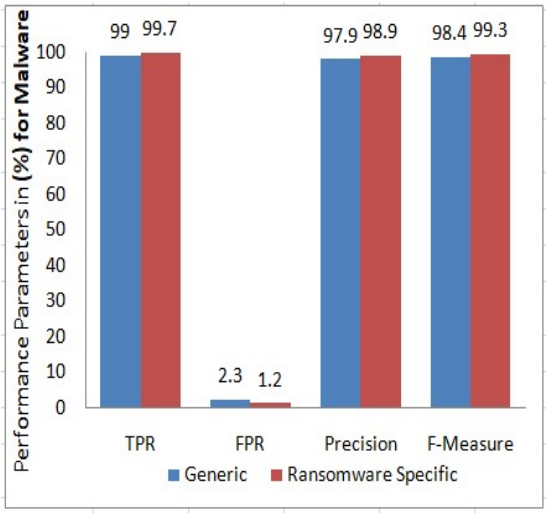

Fig. 5 Detailed Performance Comparison: Malware

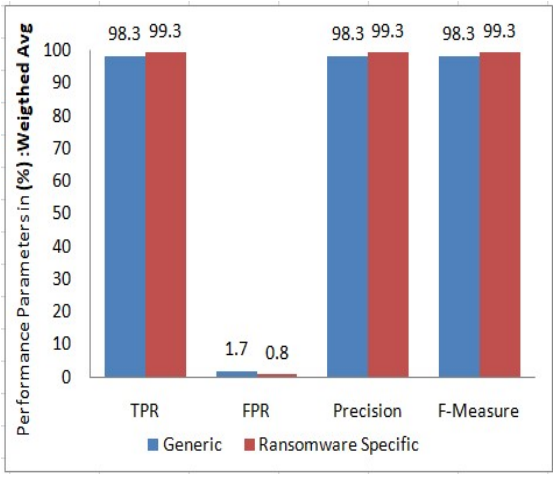

Fig. 6: Detailed Performance Comparison: Weighted Average

Along with the static properties specific for ransomware executables, seven runtime properties, typically observed during ransomware execution, are identified and listed in section 4.4.

6. Conclusions

The proposed approach is to learn and identify the typical characteristics of ransomware through static and run-time analysis of the ransomware, and increase the identification parameters. We have identified nine static properties of PE file that are specific to ransomware and seven common behavior during the execution of the ransomware. The inclusion of the specific properties in the existing generic static properties has shown improvement in the classification performance of malware. The further enhancement can be achieved by addressing the challenges such as identifying the evasive behavior and locking property of certain ransomware.

References

[1] McAfee Mobile Threat Report Q1, 2019 (2019). Retrieved from https://www.mcafee.com/enterprise/en-us/assets/reports/rp-mobile-threat-report-2019.pdf. Last Accessed: April 2019.

[2] Crowe J. (2017). Must-Know Ransomware Statistics. Stats & Trends. Retrieved from June 2017. Last Accessed: 30 Sep 2018.

[3] SophosLabs 2018 Malware Forecast (2018). Retrieved from https://www.sophos.com/de-de/medialibrary/PDFs/.../malware-forecast-2018.ashx. Last Accessed: 30 Sept 2018.

[4] Abraham S. (2019, April). Ransomware Statistics in 2019 (Infographic). https://www.malwarefox.com/ransomware-statistics-infographic/. Last Accessed: April 2019.

[5] Brewer, R. (2016). Ransomware attacks: detection, prevention and cure. Network Security, 2016(9), 5-9.

[6] Azmoodeh, A., Dehghantanha, A., Conti, M., & Choo, K. K. R. (2018). Detecting crypto-ransomware in IoT networks based on energy consumption footprint. Journal of Ambient Intelligence and Humanized Computing, 9(4), 1141-1152.

[7] Data Unit Blog. (2016). The Relationship between TOR and Ransomware. Extracted from blog.dataunit.be/the-relationship-between-tor-andransomware. Last Accessed: 10 Sep 2018.

[8] Singh S. (2018, January). Ransomware Command and Control Detection Using Machine. Aclavio. Retrieved from https://www.acalvio.com/ransomware-command-andcontrol-detection-using-machine-learning; Last Accessed: 10 Sep 2018.

[9] Damodaran, A., Di Troia, F., Visaggio, C. A., Austin, T. H., & Stamp, M. (2017). A comparison of static, dynamic, and hybrid analysis for malware detection. Journal of Computer Virology and Hacking Techniques, 13(1), 1-12.

[10] Vidyarthi, D., Choudhary, S. P., Rakshit, S., & Kumar, C. S. (2017). Malware Detection by Static Checking and Dynamic Analysis of Executables. International Journal of Information Security and Privacy (IJISP), 11(3), 29-41.

[11] Kharaz, A., Arshad, S., Mulliner, C., Robertson, W., & Kirda, E. (2016). UNVEIL: A Large-Scale, Automated Approach to Detecting Ransomware. In 25th USENIX Security Symposium (USENIX Security 16) (pp. 757-772).

[12] Kramer, S., & Bradfield, J. C. (2010). A general definition of malware. Journal in computer virology, 6(2), 105-114.

[13] Subedi, K. P., Budhathoki, D. R., & Dasgupta, D. (2018, May). Forensic analysis of ransomware families using static and dynamic analysis. In 2018 IEEE Security and Privacy Workshops (SPW) (pp. 180-185). IEEE.

[14] Zimba, A., Wang, Z., & Chen, H. (2018). Multi-stage crypto ransomware attacks: A new emerging cyber threat to critical infrastructure and industrial control systems. ICT Express, 4(1), 14-18.

[15] Abrams, L. (2018). The Week in Ransomware - August 10th 2018 - BitPaymer & KeyPass. BleepingComputer. Extracted from https://www.bleepingcomputer.com/news/security/the-week-in-ransomware-august-10th-2018-bitpaymer-and-keypass/. August 11, 2018. Last Accessed: 10 Feb 2019.

[16] Zavarsky, P., & Lindskog, D. (2016). Experimental analysis of ransomware on windows and android platforms: Evolution and characterization. Procedia Computer Science, 94, 465-472.

[17] Chen, Q., & Bridges, R. A. (2017, December). Automated behavioral analysis of malware: A case study of wannacry ransomware. In 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA) (pp. 454-460). IEEE.

[18] Bhardwaj, A., Avasthi, V., Sastry, H., & Subrahmanyam, G. V. B. (2016). Ransomware digital extortion: a rising new age threat. Indian Journal of Science and Technology, 9(14), 1-5.

[19] Bayer, U., Moser, A., Kruegel, C., & Kirda, E. (2006). Dynamic analysis of malicious code. Journal in Computer Virology, 2(1), 67-77.

[20] Windows Sysinternals (2017). Retrieved and downloaded from https://docs.microsoft.com/en-us/sysinternals/downloads/. Last Accessed: April 2019.

[21] VXHeaven (2017). Malware Samples: Viruses don't harm, ignorance does!. Retrieved and downloaded from http://83.133.184.251/virensimulation.org/. Last Accessed: 10 Sep 2018.

[22] GitHub Ransomware Samples.(2018). Retrieved and downloaded from https://github.com/fabrimagic72/malware-samples/tree/master/Ransomware Last Accessed: 10 Feb 2019.

[23] OllyDbg (2019). Retrieved and downloaded from http://www.ollydbg.de/. Last Accessed: 10 Feb 2019.

[24] Debugging Tools for Windows 10 (WinDbg) (2019). Retrieved from https://docs.microsoft.com/en-us/windows-hardware/drivers/debugger/debugger-download-tools#small-classic-windbg-preview-logo-debugging-tools-for-windows-10-windbg. Last Accessed: 10 Feb 2019.

Deepti Vidyarthi completed B.E. Information Technology from the University of Pune, India in 2005, and M.Tech. Computer Science and Engineering from IIT Kanpur, India, in 2010. She is currently pursuing Ph.D. in Computer Science and Engineering from Defence Institute of Advanced Technology, Pune, India. She is Assistant Professor at the Department of Computer Science and Engineering, Defence Institute of Advanced Technology since 2011. Her current research interests include program analysis and verification, malware analysis and operating system security.

CRS Kumar, Professor, Department of Computer Science and Engineering, Defence Institute of Advanced Technology, Pune, India. His research interests include Network security, Cryptography, Wireless Network Security, Game Theory. He is a Fellow of IETE, Fellow of Institution of Engineers, Senior Member of IEEE.

Subrata Rakshit completed his B. Tech in Eng Phy from IIT Bombay in 1988, his M.S. and PhD from Caltech USA in 1989 and 1994 respectively. He has been with CAIR, DRDO since Dec 1994, working in Neural Networks, Image Processing, MSDF, Computer Vision and Secure Systems. He is currently Technical Mentor, Secure Systems Division.

Shailesh Chansarkar, Senior Scientist at Center of Artificial Intelligence & Robotics (CAIR), DRDO, Bangalore. His areas of interest include Computer Security, Intrusion Detection and Prevention, Database Security, Malware Analysis, and Secure Systems Division.