This paper is a preprint (IEEE 'accepted' status).

DOI. Not yet assigned.

RansomWall: A Layered Defense System against Cryptographic Ransomware Attacks using Machine Learning

Saiyed Kashif Shaukat, Vinay J. Ribeiro
Department of Computer Science and Engineering,
Indian Institute of Technology Delhi, Hauz Khas, New Delhi, India
saiyedkashif@gmail.com, vinay@cse.iitd.ernet.in

***Abstract*—Recent worldwide cybersecurity attacks caused by Cryptographic Ransomware infected systems across countries and organizations with millions of dollars lost in paying extortion amounts. This form of malicious software takes user files hostage by encrypting them and demands a large ransom payment for providing the decryption key. Signature-based methods employed by Antivirus Software are insufficient to evade Ransomware attacks due to code obfuscation techniques and creation of new polymorphic variants everyday. Generic Malware Attack vectors are also not robust enough for detection as they do not completely track the specific behavioral patterns shown by Cryptographic Ransomware families.**

This work based on analysis of an extensive dataset of Ransomware families presents RansomWall, a layered defense system for protection against Cryptographic Ransomware. It follows a Hybrid approach of combined Static and Dynamic analysis to generate a novel compact set of features that characterizes the Ransomware behavior. Presence of a Strong Trap Layer helps in early detection. It uses Machine Learning for unearthing zero-day intrusions. When initial layers of RansomWall tag a process for suspicious Ransomware behavior, files modified by the process are backed up for preserving user data until it is classified as Ransomware or Benign. We implemented RansomWall for Microsoft Windows operating system (the most attacked OS by Cryptographic Ransomware) and evaluated it against 574 samples from 12 Cryptographic Ransomware families in real-world user environments. The testing of RansomWall with various Machine Learning algorithms evaluated to 98.25% detection rate and near-zero false positives with Gradient Tree Boosting Algorithm. It also successfully detected 30 zero-day intrusion samples (having less than 10% detection rate with 60 Security Engines linked to VirusTotal).

I. Introduction

In today's digitally connected world, organizations across the globe are witnessing a massive growth in cybercrime. The increased dependency on digital technologies has helped economies transform the world of business but also lead to escalation in the number of cyberattacks. Individual users and corporates keep their important documents, photos, reports and organizational data in digital form. Recently, massive-scale attacks were carried out using a kind of malware known as *Ransomware* [1] that denies access to user data files and demands a ransom for restoring it. In a very short period of time, Ransomware has grown exponentially to become the most dangerous and aggressive malware of recent times. The attacks have been carried out on various sectors [2] including finance, insurance, banking, real estate, medical, public administration to name a few.

Scareware [3] is an early form of Ransomware which leverages false fear in the victim that his system is infected with a large number of viruses, spyware and security issues. The user is tricked to buy a fake antivirus product and hence pay a ransom for removing infections. User awareness and improved security software have drastically reduced threat posed by this kind of malware. *Locker Ransomware* (e.g. Reveton [4]) denies access to computing resources by locking system's user interface. It employs social engineering methods for threatening the user to pay ransom. Effective tools and techniques are provided by various security vendors which are able to restore the blocked user interface for most variants.

Cryptographic Ransomware [5] targets user data files with specific extensions that varies with each family. Access to user data is blocked by encrypting files with advanced encryption algorithms. A Ransom-note is displayed to the user containing threatening message to delete hostage files permanently in case of non-payment. Ransom is requested through Bitcoin cryptocurrency. System files are not encrypted to keep the operating system working. Even after payment it is not guaranteed that the user receives the decryption key to restore encrypted files.

Modern-day Cryptographic Ransomware variants use a combination of *Symmetric (AES, Triple DES) and Asymmetric (RSA, ECC) Key Cryptographic algorithms* for encryption. User files are encrypted using Symmetric Key generated in the victim's system. The Symmetric Key is encrypted using Asymmetric Public Key provided by the attacker whereas corresponding Asymmetric Private Key is kept secret at Command & Control server [6]. Cyberattackers create a new *Bitcoin wallet* for each infection and send its identifier to the victim for ransom payment. Anonymity is provided by passing Bitcoins through multiple mixers which shuffles them among different users. *Tor networks* are used for hidden communication with Command & Control server.

Cryptographic Ransomware is the major variant of Ransomware families that has caused devastation worldwide as compared to the other two variants - Scareware and Locker Ransomware. *Microsoft Windows* operating system has become the most attacked OS by Cryptographic Ransomware

in recent times with huge cyberattacks primarily targeting its vulnerabilities for entering into the victim's system [7]. Widespread use of this operating system in various platforms across globe is the main reason for emerging as the prime target of these attacks.

Due to huge extortion amounts involved, new Cryptographic Ransomware variants are created everyday. Most of the existing Ransomware detection techniques are effective against known and already analyzed samples but very weak against polymorphic, obfuscated and zero-day attacks which are extensively used by modern-day Cryptographic Ransomware. Indicators used for tracking are large in number and similar to generic malware, but they do not completely capture the specific behaviors shown by Ransomware families.

This paper presents *RansomWall*, a layered defense system for protection against Cryptographic Ransomware. Each RansomWall layer is based on a specific functionality. The layers are organized in computation order of the features that are generated during the sample's execution. It is implemented for Microsoft Windows operating system. Contributions of this work are as follows:

- **Identify a novel compact set of features that characterizes the Cryptographic Ransomware behavior:** Based on analysis of an extensive dataset of Ransomware families, a novel compact feature set is identified which capture patterns common across different Cryptographic Ransomware variants. The Layered Architecture of RansomWall follows a Hybrid approach of combined Static and Dynamic analysis to compute values of the selected feature set.
- **Create a Strong Trap Layer that helps in early detection:** This layer tracks malicious activities performed by Cryptographic Ransomware to break defenses of the victim's system and monitors file operations performed extensively for encrypting user data files. These activities are essential components of a Cryptographic Ransomware attack.
- **Use Machine Learning for unearthing zero-day intrusions:** Cryptographic Ransomware extensively use polymorphic, metamorphic and obfuscation techniques to evade signature-based detection mechanisms employed by Antivirus Software. Most successful intrusions are zero-day attacks. Machine Learning is used to develop a generalized model for the compact feature set that is able to detect zero-day samples. Performance of following supervised learning algorithms is evaluated: Logistic Regression, Support Vector Machines (Gaussian-Kernel), Artificial Neural Networks, Random Forests and Gradient Tree Boosting.
- **Develop a Backup mechanism for preserving user data during detection process:** The computation and analysis of compact feature set values by RansomWall layers and classification decision by Machine Learning Engine takes some time, while Ransomware is already encrypting the user data files. Therefore, there is a need to tag a process as suspicious based on initial features of Static, Dynamic and Trap layers. The files modified by the suspicious process are backed up in a separate folder to preserve user data until the process is classified as Ransomware or Benign by Machine Learning Layer.
- **Evaluate RansomWall against 574 samples from 12 Cryptographic Ransomware families and 442 samples of Benign Software in real-world user environments:** The Performance Metrices of Machine Learning Layer show best results with Gradient Tree Boosting Algorithm. With this learning model, RansomWall attains a detection rate (TPR) of 98.25% with near-zero false positives.
- **Compare RansomWall's capability to detect zero-day intrusion samples with 60 Security Engines linked to VirusTotal:** 30 zero-day intrusion samples having less than 10% detection rate by 60 Security Engines linked to VirusTotal [8] are collected. RansomWall successfully detected all the samples in real-time.

Rest of the paper is organized as follows. Section II describes literature review and related work. Section III presents the system design and layered architecture of RansomWall. It also describes the selected compact feature set based on experimental analysis of Ransomware families. The implementation details and development environment are presented in Section IV. In Section V results obtained with various Supervised Machine Learning algorithms along with Evaluation Metrices are described. Finally, Section VI concludes the paper.

II. Related Work

The related works can be broadly classified into two categories: a) Approaches that treat Ransomware as a subset of the overall malware community and apply generic malware indicators for their detection. b) Methods designed specifically for Ransomware based on their characteristic properties.

Antivirus Signature Based Detection Techniques [9] are effective against known threats whose signatures are already present in their databases but weak against polymorphic and zero-day attacks. *Group Policies and Application Whitelisting* [10] are mostly used within restricted corporate networks but are not suitable for individuals and public organizations. There is a dependency on correct maintenance of whitelisted applications list. Moreover, it is still possible for malware to exploit vulnerabilities in whitelisted software. *Static Analysis Detection Techniques* based on Control Flow Graphs [11], Data Flow Graphs [12] and System API Calls [13] are prone to code obfuscation, polymorphic and metamorphic techniques.

Real-Time Virtual Environment Analysis based on tracking information flow [14] has the limitation that many Ransomware variants wait for a long time based on timers, count of system restarts etc. before starting malicious activity. Hence, it is not always feasible to isolate the sample and run it in a virtual environment during real-time execution. *Network-Based Intrusion Detection Systems* [15] find anomalies in network patterns. Ransomware exchange limited number of encrypted messages with Command & Control server which are difficult to differentiate from normal traffic.

Kharraz et al. [16] performed evolution-based study of Locker and Cryptographic Ransomware. They suggested techniques based on monitoring of Master File Table and file-system activities but these methods were not evaluated by the authors. Andronio et al. [17] developed a technique for detecting Android mobile Ransomware using analysis of threatening ransom messages. Detection of ransom-note on Windows platform once it is displayed to the victim is not useful in real-time analysis as user data is already encrypted at this stage.

Kharraz et al. [18] implemented techniques incorporating structural similarity of screenshots before and after sample execution for detecting Locker Ransomware and monitoring of file-system activities for Cryptographic variants. These methods do not track malicious operations, executed by Ransomware to break system defenses, for early detection nor provide any backup mechanism for preserving user data files during analysis process. Mercaldo et al. [19] utilized formal methods for detecting Android Ransomware using Bytecode representations.

III. System Design

This section presents system design and layered architecture of RansomWall.

A. Analysis of Cryptographic Ransomware Attack stages

- *Stage 1 - Propagation and Infection:* Cyberattackers use wide range of mechanisms - spam emails, drive-by-downloads, malvertisement, botnets, self-propagation, malicious applications, removable drives, Ransomware-as-a-Service, compromised logins on servers etc. Hence, there is a lot of variation in attack vectors.
- *Stage 2 - Communication with Command & Control server:* A few small messages are exchanged between Ransomware client on the victim's system and Command & Control server, which are extremely difficult to differentiate from normal traffic. In advanced Ransomware the messages are encrypted and Tor networks are used making sniffing and tracing ineffective. Also, using this communication as a detection method is not feasible as many variants encrypt user data before contacting Command & Control server resulting in late detection.
- *Stage 3 - Encryption of User Data:* Ransomware performs malicious activities to break defenses of the victim's system and encrypt user data. These activities are characteristic behavior of Cryptographic Ransomware and common across all variants. The attack and infection vectors in first stage are large in number with new software vulnerabilities exploited every day, whereas encryption and malicious activities to break defenses are vectors that cannot be avoided by Cryptographic Ransomware.
- *Stage 4 - Extortion:* The final stage of displaying Ransom-note to the victim and ransom payment via Bitcoins is too late for detection as user data is already encrypted. Moreover, cybercriminals use Bitcoin cryptocurrency and Tor networks for anonymity and untraceability.

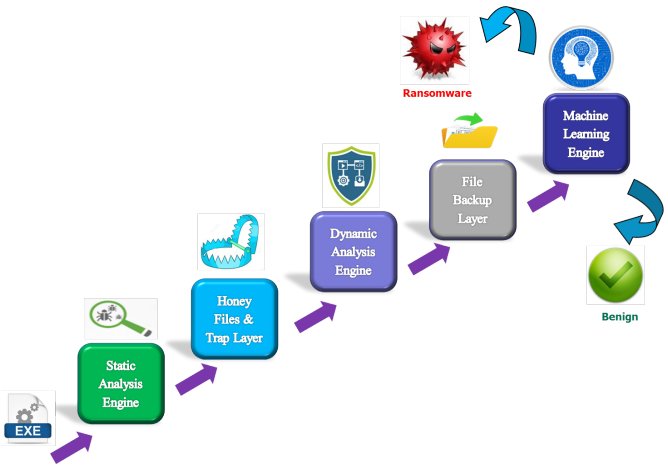

Fig. 1. RansomWall Layered Defense Architecture

Thus, detecting Cryptographic Ransomware based on its characteristic behavior of malicious activities for breaking system's defense mechanism and massive encryption, in early phase of Stage 3 is a suitable solution.

B. RansomWall Layered Defense Architecture

Each RansomWall layer is based on a specific functionality. The layers are organized in computation order of the features that are generated during the sample's execution as shown in Fig. 1. Successful tracking of suspicious Ransomware behavior by an early defense layer results in faster detection. The various layers present in RansomWall Architecture are described below.

1) Static Analysis Engine: It provides useful information from binary code of the sample. This is the first layer of RansomWall Architecture as features required for analysis can be obtained before executing the sample. Static features considered during the experimental analysis for finding their effectiveness against Ransomware families are: PE (Portable Executable) header details, embedded resources, detection of packers/cryptors, sample's Entropy, PE Digital Signature, embedded strings and fuzzy hashes.

2) Honey Files & Trap Layer: The behavioral patterns characteristic of Cryptographic Ransomware involve performing malicious activities to break defenses of the victim's system and encrypting user data files. This layer sets trap by tracking occurrence of these malicious activities. Cryptographic Ransomware perform encryption of user data files with specific extensions. Honey Files (with user data file extensions mostly attacked by Ransomware) and Honey Directories are deployed in critical user data folders. These are trap files/directories which are not expected to be modified by the user during normal operation. Modification of these files/directories by a process provides an indication of suspicious behavior.

3) Dynamic Analysis Engine: Static features alone are not sufficient due to code obfuscation, packing and encryption techniques used by Ransomware. Dynamic analysis monitors

behavior of the sample during actual execution. Cryptographic Ransomware performs extensive encryption of user data files. This layer monitors file system operations and entropy modifications for tracking massive encryption activities.

4) File Backup Layer: The computation and analysis of features collected during the sample's execution and classification decision by Machine Learning layer takes some time, while Ransomware is already encrypting user data files. Therefore, there is a need to tag a process as suspicious based on initial features of Static, Dynamic and Trap layers. Files modified by the suspicious process are backed up in a separate folder to preserve user data until the process is classified as Ransomware or Benign by Machine Learning layer. RansomWall maintains list of files that are backed up along with their original locations and Process ID of the suspicious process. If Machine Learning layer classifies as Ransomware, then the process is killed and files modified by it are restored to their original locations. If it is classified as Benign then these files are deleted from the backup folder.

5) Machine Learning Engine: This layer builds a generalized model which is effective against zero-day Ransomware attacks. It takes feature values collected by Static, Dynamic and Trap layers as input and classifies the executable as Ransomware or Benign. The Machine Learning Engine is trained offline using Supervised algorithms. Training data consists of feature values with Ransomware and Benign labels. Trained Machine Learning Engine use the learned model to classify executables in real-time based on input feature values. Performance of following Supervised Machine Learning algorithms is evaluated: Logistic Regression, Support Vector Machines (Gaussian-Kernel), Artificial Neural Networks, Random Forests and Gradient Tree Boosting.

C. Experimental Setup

We perform an extensive experimental analysis of samples collected from various Cryptographic Ransomware families to derive a compact set of relevant features for Static, Dynamic and Trap Layers of RansomWall. The experiments are executed in following stages:

- *Stage 1 - Sample Collection:* Cryptographic Ransomware samples across various families are collected from VirusShare [20].
- *Stage 2 - Static Reverse Engineering:* The collected samples are reverse engineered to perform initial analysis from binary code. IDA (Interactive Disassembler) tool [21] is used.
- *Stage 3 - Execution in Controlled Sandbox Environment:* We use Cuckoo Sandbox [22] for generating the experimental setup. The Guest Virtual Machines consist of Windows 7 and Windows 8.1 as the operating systems. User data environment of around 500 directories and 8000 files with varied data extensions and file sizes is created. The setup includes common utility software along with other processes running to create a realistic user environment. We execute multiple anti-emulation scripts to simulate mouse movements and clicking, keyboard strokes, process executions, system restarts; file access, creation and deletion. IP address range and MAC addresses of the VMs are changed to avoid virtual setup detection. To create a secure testing environment network traffic is controlled by using firewall, only allowing limited HTTP and DNS traffic to access Command & Control server and restricting network bandwidth. We use the best practices for creating malware execution environments described in the paper [23]. Wireshark tool [24] is used for monitoring messages exchanged between Ransomware client on the victim's system and Command & Control server. We allow each sample to execute for 60 minutes or until it gets terminated automatically. After every sample execution, state of the test environment is reset by restoring the original snapshot.

D. Compact Feature Set for Static, Dynamic and Trap Layers

Based on the experimental analysis of behavioral patterns demonstrated by various Cryptographic Ransomware families, following set of features are selected for RansomWall layers.

1) Static Analysis Engine:

a) PE Digital Signature Verification: Ascertains that certificate which is used to sign the sample, chains up to a root certificate located in the Windows Trusted Root Certificate Store. This implies that identity of the publisher is verified by a Certification Authority. Microsoft Windows WinVerifyTrust API [25] is used for the verification purpose.

b) Presence of Packers/Cryptors: Windows PEiD tool [26] is used to detect the presence of Packers/Cryptors. Detection may not happen if customized versions of Packers/Cryptors are used. But even in this case the sample has a high Entropy which is verified.

c) Suspicious Embedded Strings: Presence of strings related to: Ransom, Bitcoin, Encrypt and Crypto. An open source tool FLOSS [27] is used for extracting embedded strings from obfuscated Ransomware binary.

2) Honey Files & Trap Layer:

a) Honey Files/Directories modification: They are placed such that Ransomware attacks them earlier than the critical files. From Ransomware analysis it is observed that many variants use Depth First Search while enlisting files for encryption. Write, Rename and Delete operations on Honey Files/Directories are tracked for malicious activities.

b) Suspicious Windows Cryptographic API usage: From analysis of Ransomware families it is observed that most variants use standard Windows Cryptographic APIs for encryption. Massive use of these APIs can be considered suspicious. Windows Crypto APIs tracked are stored in: rsaenh.dll, cryptsp.dll, cryptbase.dll, bcrypt.dll, crypt32.dll, cryptdll.dll, cryptsvc.dll and dssenh.dll.

c) Disabling safe-mode boot options: Ransomware disables safe-mode boot options to prevent recovery of the victim's system by booting in safe mode and taking corrective action.

It is a common behavior across Ransomware variants and bcdedit.exe is used for this purpose.

d) Deletion of Volume Shadow copies: Ransomware deletes Windows Volume Shadow copies so that the user is not able to recover data from the backup. Integrity of these copies is tracked. vssadmin.exe and WMIC.exe are used for issuing commands to delete them.

e) Suspicious Registry modifications: Analysis of Cryptographic Ransomware families have shown that they execute Registry modifications to perform malicious activities at boot time in order to maintain persistence. Some Ransomware variants store list of encrypted files in Registries. These Registry modifications to maintain persistence across reboots are tracked.

3) Dynamic Analysis Engine:

a) Directory Info Queries: To perform encryption, Ransomware first constructs list of user data files having extensions targeted by its family. To form the list it generates a large number of Directory Listing Queries to get info regarding contents of each directory. Large number of Directory Listing operations by a process is tracked as a suspicious behavior.

b) File Read Operations: Contents of user data files are read before encrypting them. Massive encryption generates extensive read operations on user data files with target extensions that are tracked.

c) File Write Operations: Encrypted user data is written back to the file generating huge write operations which are monitored.

d) Data to Non-Data File Rename Operations: Most Ransomware variants rename files to an extension (non-data) which is characteristic of their family after encrypting them. This results in massive file rename operations from data to non-data file extensions.

e) File Delete Operations: Some Ransomware families delete original files after creating new encrypted files. Extensive deletion operations performed on user data files with target extensions are monitored.

f) File Fingerprinting: Most Ransomware variants encrypt entire file contents including file signature in header which uniquely identifies its extension. Modification of file signature in header of a user data file to a new signature which does not match its extension in a write operation indicates suspicious behavior. In normal operation a file rename should result in file signature modification instead of a write operation. Moreover, in normal write operation file signature should match its extension.

g) Shannon's Entropy of File Writes: Entropy of data buffer in memory modified during file write operation to a value around 8 indicates encryption possibility. Massive write operations on user data files with high Entropy suggest suspicious behavior to be tracked.

IV. Implementation

This section describes implementation of RansomWall.

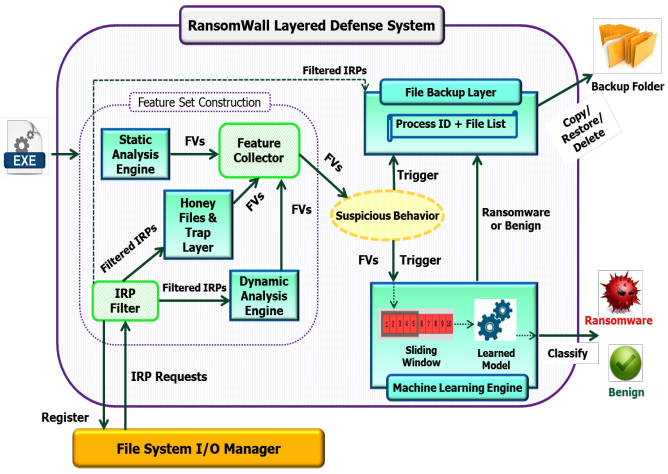

Fig. 2. RansomWall Logical Workflow

A. RansomWall Logical Workflow

The RansomWall Logical Workflow is depicted in Fig. 2. The Compact Feature Set extraction is performed by Static, Dynamic and Trap Layers. Feature values are forwarded to Feature Collector which accumulates them for each process. File System activities are monitored by analyzing IRPs (I/O Request Packets) which are generated for each file operation. IRP Filter registers with File System I/O Manager during RansomWall initialization for receiving IRP messages. During file operations, I/O Manager forwards generated IRP messages to the registered IRP Filter. The IRP Filter forwards IRP messages, which are created for file operations on only user data files, to Dynamic and Trap Layers for feature computation.

If the Feature Collector observes presence of 6 or more feature indicators in the Compact Feature Set, then the process is tagged as suspicious. For suspicious processes File Backup and Machine Learning Layers are triggered. Filtered IRPs are forwarded to File Backup Layer. If it observes a file modification request by the suspicious process, then the file is backed up in a backup folder. For the suspicious process, Feature Collector sends feature values to Machine Learning Engine on regular intervals. The Machine Learning Engine performs moving average sliding window on the received feature values to get smoothed average numbers. These are fed to the Learned Model (from Offline Training) to classify the sample as Ransomware or Benign. The classification output is sent to File Backup layer. If the suspicious process is classified as Ransomware then it is killed and File Backup layer search for the files backed up in its list by Process ID value and restore the files back to their original locations. Otherwise, if it is marked as Benign then files backed up due to the suspicious process are deleted.

B. RansomWall File System Filter Driver

RansomWall implements a File System Filter Driver as shown in Fig. 3. It is a kernel level driver that filters I/O operations performed on one or more file systems. For Microsoft Windows operating system modern File System Filter Drivers

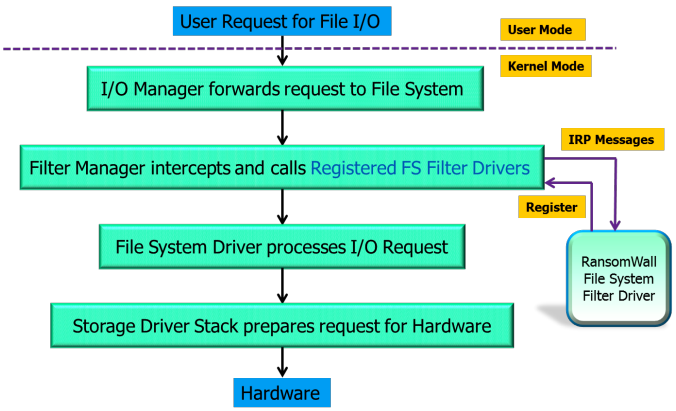

Fig. 3. RansomWall File System Filter Driver

are known as minifilter drivers [28]. Filter Manager forwards I/O Request Packets generated by file system operations to the registered filter drivers. Pre-operation, post-operation or both callback routines can be registered for desired IRPs depending on whether monitoring is required before or after the file operation.

C. Machine Learning Layer Implementation

To distinguish complex benign activities involving large number of file system operations from behavioral patterns demonstrated by Cryptographic Ransomware, feature values within the Compact Feature Set are required to build decision boundaries for classification. This functionality is provided by the Machine Learning Layer.

The Machine Learning Layer is based on Sequential Supervised Learning with Moving Average Sliding Window. As the output required is either Ransomware or Benign, hence binary classification is used. Feature value data with Ransomware and Benign labels are provided for offline training, therefore Supervised Machine Learning Algorithms are used. Feature values of Compact Feature Set for each process are continuously computed by Static, Dynamic and Trap Layers. Feature Collector fetch the values at regular time-intervals known as Buckets. In the implementation, Bucket Size is selected as 1 second. If a process is tagged as suspicious, then Feature Collector sends feature values of the process to Machine Learning Layer. These feature values are treated as sequential data. For avoiding false detections due to small glitches in process behavior, the feature values are smoothed by taking average over 3 time-intervals (1 Current and 2 Previous). Hence, input represents a moving average sliding window scenario. The Machine Learning Layer outputs a suspicious process as Ransomware or Benign if the classification result is same for 3 contiguous time-intervals, to reduce false detections.

The Machine Learning Layer is implemented using APIs provided by scikit-learn [29]. Following Supervised Machine Learning Algorithms are evaluated for RansomWall: Logistic Regression, Support Vector Machines (Gaussian-Kernel), Artificial Neural Networks, Random Forests and Gradient Tree Boosting.

TABLE I
SAMPLE-SET DISTRIBUTION ACROSS RANSOMWARE FAMILIES

S.No.	Cryptographic Ransomware Family	# Samples
1	CryptoWall	68
2	TeslaCrypt	56
3	Cerber	112
4	CTB-Locker	26
5	Jigsaw	42
6	TorrentLocker	51
7	Locky	64
8	CryptoLocker	32
9	CryptoDefense	36
10	Hidden Tear	32
11	CryptoFortress	28
12	CrypVault	27
	Total	**574**

D. Development Environment and Debugging Tools

Development Environment for implementing RansomWall:
- Microsoft Visual Studio 2015.
- Microsoft Windows SDK 10.
- Microsoft Windows Driver Kit 10.
- C Programming Language.

Debugging Tools used during RansomWall implementation:
- Windows WinDbg: Offline debugging of Kernel crashes.
- Windows Debug-View: Displays Kernel Mode Traces.
- Windows Sysinternals Process Explorer: Active processes.

V. EVALUATION AND RESULTS

This section describes the evaluation process and results obtained with RansomWall testing.

A. Evaluation Sample-Set

Cryptographic Ransomware used for testing RansomWall are collected from VirusShare [20]. The Sample-Set consists of 574 samples from 12 Cryptographic Ransomware families as described in Table I. It also contains 442 samples of Benign Software.

B. Evaluation Setup

The setup used for testing RansomWall consists of:

a) Execution in Controlled Sandbox Environment as used in the experimental analysis of Ransomware variants.

b) Selective samples from each Ransomware family are evaluated on Real Testbed with Windows 8.1 as operating system, all common utility software installed and user data environment similar to the experimental setup.

C. Feature Vector Plots (Ransomware and Benign Samples)

To evaluate effectiveness of the Compact Feature Set in distinguishing behavioral patterns between Cryptographic Ransomware and Benign Software, Feature Vectors obtained from execution of the samples are collected through debug traces and plotted. Feature Vector Plots shown in Fig. 4 (Ransomware) and Fig. 5 (Benign Software) demonstrate presence of common behavioral characteristics across Ransomware families which are different from Benign Software.

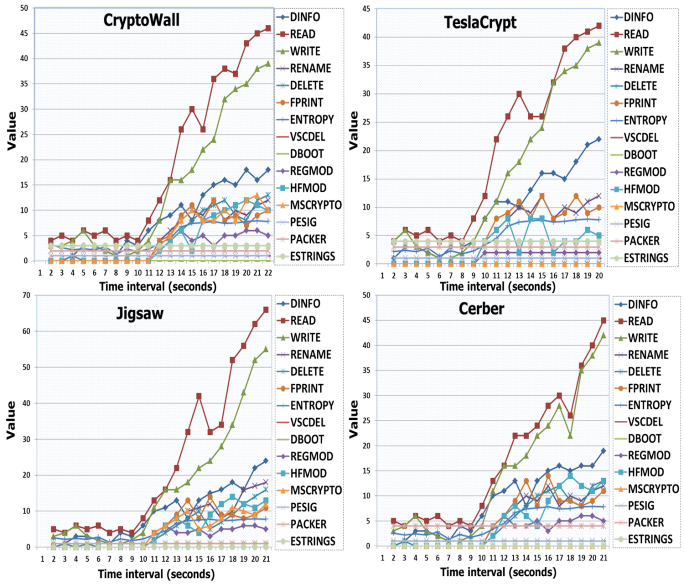

Fig. 4. Feature Vector Plots (Cryptographic Ransomware)

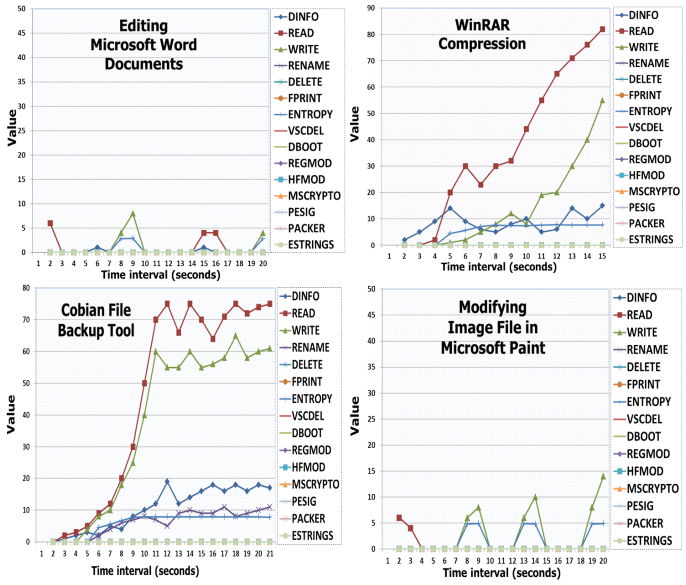

Fig. 5. Feature Vector Plots (Benign Software)

D. Evaluation Process

Cryptographic Ransomware and Benign Software samples are executed in the Evaluation setup. RansomWall computes feature values at regular time-intervals and these are collected through debug traces. The collected feature values of Compact Feature Set are labeled as Ransomware/Benign and fed to the Machine Learning Layer for offline training.

12-Fold Cross Validation is performed on the Ransomware Sample-Set. In each test run, Machine Learning Layer is trained on all samples from 11 out of 12 Cryptographic Ransomware families and 221 out of 442 samples from Benign Software. The learned model is tested against all samples from the last Ransomware family and remaining Benign samples on the Evaluation Setup. This process of evaluation is selected since most of the successful Ransomware Attacks are zero-day intrusions with samples from an entirely new Ransomware family or its upgraded variant. The generalization capabilities of various Supervised Machine Learning Algorithms are tested for RansomWall.

During evaluation, functionality of the File Backup Layer is also verified to check if the files are correctly backed up for suspicious processes and restored/deleted after receiving classification output from the Machine Learning Layer.

E. Evaluation Results

Performance Metrices of evaluated Supervised Machine Learning algorithms are presented in Table II. The computed Metrices show best results with Gradient Tree Boosting Algorithm. With this Learning Model, RansomWall attains a detection rate (True Positive Rate) of 98.25% with near-zero false positives. The advantages of Gradient Tree Boosting Algorithm [29] - efficient handling of heterogeneous data (numerical, categorical and features with different scale in the Compact Feature Set), high predictive power and robustness to outliers result in high performance.

Analysis of *False Negatives* show that two Ransomware samples abruptly terminated after encrypting only a few files. Since, resulting file system activity is limited hence samples are not detected. Another Ransomware sample encrypted only .vcf files. Number of such files on the user system is low hence limited file system activity is generated leading to false negative. Rest of the misclassification is due to decision boundary errors.

F. Comparison of RansomWall with 60 Security Engines linked to VirusTotal

30 zero-day Cryptographic Ransomware samples (collected from VirusTotal [8] which are not included in the Evaluation Sample-Set and having very low detection rate, less than 10%, with 60 Security Engines linked to VirusTotal) are also evaluated on Real Testbed with Windows 8.1 as operating system and all common utility software installed and running. RansomWall (with Gradient Tree Boosting Algorithm) successfully detected all the samples in real-time and backed up user files are restored to their original locations.

G. System Overhead

Feature values computation and collection for normal processes i.e. without triggering File Backup and Machine Learning Layers add less than 1% CPU Load. Ransomware/Suspicious processes i.e. with triggering of File Backup and Machine Learning Layers lead to around 2% CPU Usage.

VI. Conclusion

Recent worldwide cybersecurity attacks caused by Cryptographic Ransomware massively crippled organizations across the globe. Based on the analysis of an extensive Ransomware

TABLE II
TRUE POSITIVE RATE AND FALSE POSITIVE RATE :: LOGISTIC REGRESSION, SUPPORT VECTOR MACHINE, ARTIFICIAL NEURAL NETWORK, RANDOM FOREST AND GRADIENT TREE BOOSTING

S.No.	Ransomware Family	# Samples	LR		SVM		ANN		RF		GTB	
			TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR
1	CryptoWall	68	0.8529	0.0407	0.8823	0.0497	0.9411	0.0090	0.9558	0.0045	0.9852	0.0045
2	TeslaCrypt	56	0.8928	0.0497	0.9285	0.0316	0.9107	0.0090	0.9821	0.0045	0.9821	0.0045
3	Cerber	112	0.8750	0.0361	0.9017	0.0135	0.9107	0.0316	0.9642	0.0090	0.9910	0.0045
4	CTB-Locker	26	0.8461	0.0542	0.8076	0.0542	0.9230	0.0135	0.9230	0.0135	0.9615	0.0090
5	Jigsaw	42	0.9047	0.0497	0.9047	0.0090	0.9523	0.0226	0.9285	0.0045	0.9761	0.0045
6	TorrentLocker	51	0.8823	0.0407	0.9411	0.0135	0.9411	0.0180	0.9607	0.0135	1.0000	0.0090
7	Locky	64	0.8750	0.0316	0.8125	0.0226	0.9062	0.0316	0.9687	0.0090	1.0000	0.0045
8	CryptoLocker	32	0.8125	0.0271	0.9062	0.0316	0.9062	0.0135	0.9687	0.0090	0.9375	0.0045
9	CryptoDefense	36	0.8055	0.0497	0.8333	0.0226	0.9444	0.0045	0.9444	0.0045	0.9722	0.0045
10	Hidden Tear	32	0.8125	0.0542	0.8750	0.0452	0.8750	0.0226	0.9062	0.0135	1.0000	0.0090
11	CryptoFortress	28	0.7857	0.0316	0.8571	0.0542	0.8928	0.0407	0.8928	0.0045	0.9642	0.0045
12	CrypVault	27	0.8148	0.0407	0.8888	0.0407	0.8888	0.0045	0.9259	0.0045	0.9629	0.0045
	Weighted Average	**574**	**0.8571**	**0.0422**	**0.8832**	**0.0324**	**0.9180**	**0.0184**	**0.9511**	**0.0079**	**0.9825**	**0.0056**

dataset, this paper presents a layered defense mechanism with monitoring of a novel compact feature set that characterizes Ransomware behavior. Strong Trap layer (early detection), Machine Learning layer (zero-day intrusions) and File Backup layer (preserving user data) helps RansomWall to attain a detection rate of 98.25% with near-zero false positives using Gradient Tree Boosting Algorithm. We will be evaluating RansomWall on large-scale real setups as a future work.

REFERENCES

[1] Barkly, "WannaCry Ransomware Statistics: The Numbers Behind the Outbreak," May 2017. [Online]. Available: https://blog.barkly.com/wannacry-ransomware-statistics-2017

[2] CNN Tech, "Ransomware attack: Who's been hit," May 2017. [Online]. Available: http://money.cnn.com/2017/05/15/technology/ransomware-whos-been-hit/index.html

[3] TechTarget, "Scareware," Aug 2010. [Online]. Available: http://whatis.techtarget.com/definition/scareware

[4] F-Secure, "Trojan: W32/Reveton: Threat description," 2017. [Online]. Available: https://www.f-secure.com/v-descs/trojan_w32_reveton.shtml

[5] Sophos, "The current state of ransomware: CTB-Locker," 2015. [Online]. Available: https://news.sophos.com/en-us/2015/12/31/the-current-state-of-ransomware-ctb-locker

[6] Panda Security, "CryptoLocker: What Is and How to Avoid it," 2015. [Online]. Available: http://www.pandasecurity.com/mediacenter/malware/cryptolocker

[7] SecureList, "WannaCry ransomware used in widespread attacks all over the world," May 2017. [Online]. Available: https://securelist.com/wannacry-ransomware-used-in-widespread-attacks-all-over-the-world/78351

[8] VirusTotal, "Free Online Virus, Malware and URL Scanner," 2017. [Online]. Available: https://www.virustotal.com

[9] Comodo, "How Antivirus Works," 2017. [Online]. Available: https://antivirus.comodo.com/how-antivirus-software-works.php

[10] SentinelOne, "The Truth About Whitelisting," Dec 2014. [Online]. Available: https://sentinelone.com/2014/12/07/the-truth-about-whitelisting

[11] P. Faruki, V. Laxmi, M. S. Gaur, and P. Vinod, "Mining control flow graph as api call-grams to detect portable executable malware," in *Proceedings of the Fifth International Conference on Security of Information and Networks*. ACM, 2012, pp. 130–137.

[12] T. Wüchner, M. Ochoa, and A. Pretschner, "Robust and effective malware detection through quantitative data flow graph metrics," in *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*. Springer, 2015, pp. 98–118.

[13] Y. Ye, D. Wang, T. Li, D. Ye, and Q. Jiang, "An intelligent pe-malware detection system based on association mining," *Journal in computer virology*, vol. 4, no. 4, pp. 323–334, 2008.

[14] H. Yin, D. Song, M. Egele, C. Kruegel, and E. Kirda, "Panorama: capturing system-wide information flow for malware detection and analysis," in *Proceedings of the 14th ACM conference on Computer and communications security*. ACM, 2007, pp. 116–127.

[15] N. Das and T. Sarkar, "Survey on host and network based intrusion detection system," *International Journal of Advanced Networking and Applications*, vol. 6, no. 2, p. 2266, 2014.

[16] A. Kharraz, W. Robertson, D. Balzarotti, L. Bilge, and E. Kirda, "Cutting the gordian knot: A look under the hood of ransomware attacks," in *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*. Springer, 2015, pp. 3–24.

[17] N. Andronio, S. Zanero, and F. Maggi, "Heldroid: Dissecting and detecting mobile ransomware," in *International Workshop on Recent Advances in Intrusion Detection*. Springer, 2015, pp. 382–404.

[18] A. Kharraz, S. Arshad, C. Mulliner, W. K. Robertson, and E. Kirda, "Unveil: A large-scale, automated approach to detecting ransomware." in *USENIX Security Symposium*, 2016, pp. 757–772.

[19] F. Mercaldo, V. Nardone, A. Santone, and C. A. Visaggio, "Ransomware steals your phone. formal methods rescue it," in *International Conference on Formal Techniques for Distributed Objects, Components, and Systems*. Springer, 2016, pp. 212–221.

[20] VirusShare, "Malware Repository," 2017. [Online]. Available: https://virusshare.com

[21] Hex-Rays, "IDA: About," 2015. [Online]. Available: https://www.hex-rays.com/products/ida

[22] C. Guarnieri, A. Tanasi, J. Bremer, and M. Schloesser, "Automated Malware Analysis - Cuckoo Sandbox," 2016. [Online]. Available: https://www.cuckoosandbox.org

[23] C. Rossow, C. J. Dietrich, C. Grier, C. Kreibich, V. Paxson, N. Pohlmann, H. Bos, and M. Van Steen, "Prudent practices for designing malware experiments: Status quo and outlook," in *Security and Privacy (SP), 2012 IEEE Symposium on*. IEEE, 2012, pp. 65–79.

[24] Wireshark, "About Wireshark," 2017. [Online]. Available: https://www.wireshark.org

[25] Microsoft, "WinVerifyTrust function," 2017. [Online]. Available: https://msdn.microsoft.com/en-us/library/windows/desktop/aa388208(v=vs.85).aspx

[26] Softpedia, "PEiD Tool," 2017. [Online]. Available: http://www.softpedia.com/get/Programming/Packers-Crypters-Protectors/PEiD-updated.shtml

[27] FireEye, "Automatically Extracting Obfuscated Strings from Malware using the FireEye Labs Obfuscated String Solver (FLOSS)," Jun 2016. [Online]. Available: https://www.fireeye.com/blog/threat-research/2016/06/automatically-extracting-obfuscated-strings.html

[28] Microsoft, "Introduction to File System Filter Drivers," 2017. [Online]. Available: https://docs.microsoft.com/en-us/windows-hardware/drivers/ifs/introduction-to-file-system-filter-drivers

[29] F. Pedregosa et al., "Scikit-learn - Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, no. Oct, pp. 2825–2830, 2011.