Automated Ransomware Behavior Analysis: Pattern Extraction and Early Detection

Qian Chen[1(✉)], Sheikh Rabiul Islam[2], Henry Haswell[1], and Robert A. Bridges[3]

[1] Electrical and Computer Engineering Department, University of Texas at San Antonio, San Antonio, TX, USA
guenevereqian.chen@utsa.edu, henry.haswell@my.utsa.edu
[2] Computer Science Department, Tennessee Technological University, Cookeville, TN, USA
sislam42@students.tntech.edu
[3] Computational Sciences and Engineering Division, Oak Ridge National Laboratory, Oak Ridge, TN, USA
bridgesra@ornl.gov

Abstract. Security operation centers (SOCs) typically use a variety of tools to collect large volumes of host logs for detection and forensic of intrusions. Our experience, supported by recent user studies on SOC operators, indicates that operators spend ample time (e.g., hundreds of man hours) on investigations into logs seeking adversarial actions. Similarly, reconfiguration of tools to adapt detectors for future similar attacks is commonplace upon gaining novel insights (e.g., through internal investigation or shared indicators). This paper presents an automated malware pattern-extraction and early detection tool, testing three machine learning approaches: *TF-IDF* (term frequency–inverse document frequency), *Fisher's LDA* (linear discriminant analysis) and *ET* (extra trees/extremely randomized trees) that can (1) analyze freshly discovered malware samples in sandboxes and generate dynamic analysis reports (host logs); (2) automatically extract the sequence of events induced by malware given a large volume of ambient (un-attacked) host logs, and the relatively few logs from hosts that are infected with potentially polymorphic malware; (3) rank the most discriminating features (unique patterns) of malware and from the behavior learned detect malicious activity, and (4) allows operators to visualize the discriminating features and their correlations to facilitate malware forensic efforts.

F. Liu et al. (Eds.): SciSec 2019, LNCS 11933, pp. 199–214, 2019.
https://doi.org/10.1007/978-3-030-34637-9_15

To validate the accuracy and efficiency of our tool, we design three experiments and test seven ransomware attacks (i.e., WannaCry, DBGer, Cerber, Defray, GandCrab, Locky, and nRansom). The experimental results show that *TF-IDF* is the best of the three methods to identify discriminating features, and *ET* is the most time-efficient and robust approach.

1 Introduction

Ransomware, a class of self-propagating malware, uses encryption to hold victim's data and has experienced a 750% increase in frequency in 2018 [1]. Recently, the majority of these ransomware attacks target local governments and small business [2]. For example, the 2018 SamSam ransomware hit the city of Atlanta, encrypted at least one third of users' applications, disrupted the city's vital services [3], and resulted in $17M of remediation to rebuild its computer network [4]. Unlike large multinational businesses, small cities and businesses usually face stricter financial constraints than larger enterprises and struggle to establish or keep pace with cyber defensive technology and adversary/malware advancements. Consequently, they are less capable to defend against cyber threats. More generally, SOC's resource constraints and the shortage of cybersecurity talent [5–7] motivate us to develop an automated tools for SOCs.

Currently, manual investigation of logs is commonplace in SOCs and extremely tedious. E.g., our interaction with SOC operators revealed a 160 man-hour forensic effort to manually analyze a few CryptoWall 3.0 infected hosts' logs [8] with the goal of (a) identifying the adversary/malware actions from user actions in their logs and (b) leveraging learned information to reconfigure tools for timely detection. This motivates our target use case—from SOC-collected logs from an attacked host (esp. a ransomware infection) and non-attack host logs, we seek to automated the (currently manual) process of identifying the attack's actions. In the ransomware case, this should be used to provides a pre-encryption ransomware detector. For testing in a controlled environment, we use "artificial logs", that is, logs obtained by running malware and ambient (emulated user) activities in a sandbox.

Note that this mirrors classical dynamic analysis—(a) performing dynamic malware analysis to (b) extract indicators or signatures—and, hence, dynamic analysis is a second use case. Malware analysis takes considerable time and requires an individual or a team with extensive domain knowledge or reverse engineering expertise. Therefore, malware analysts usually collaborate across industry, university and government to analyze the ransomware attacks that caused disruptive global attacks (e.g., WannaCry). However, the security community has insufficient resources to manually analyze less destructive attacks such as Defray, nRansom and certain versions of Gandcrab. Therefore, manual analysis reports of such malware do not provide enough information for early detection [9–16]. Our approach, regardless of the malware's real-world impacts and potential damages, efficiently help to automate tedious manual analysis by accurately extracting the most discriminating features from large amount of host logs and identifying malicious behavior induced by malware.

While our approach holds promise for more general malware and other attacks, we focus on ransomware. Note that upon the first infection identified in an enterprise, the logs from the affected host can be automatically turned into a detector via our tool. The tool applies three machine learning algorithms, (1) Term Frequency-Inverse Document Frequency (*TF-IDF*), (2) Fisher's Linear Discriminant Analysis (*Fisher's LDA*) and (3) Extra Trees/Extremely Randomized Trees (*ET*) to (a) automatically identify discriminating features of an attack from system logs (generated by an automatic analysis system, namely, Cuckoo Sandbox [17]), and (b) detect future attacks from the same log streams. Using Cuckoo and set scripts for running ransomware and emulated user activity provides source data for experimentation with ground truth. We test the tool using infected system logs of seven disruptive ransomware attacks (i.e., WannaCry, DBGer, Cerber, Defray, GandCrab, Locky, and nRansom) and non-attack logs from emulated user activities, and present experiments varying log quality and quantity to test robustness. These system logs include files, folders, memory, network traffic, processes and API call activities.

Contributions of the pattern-extraction and early detection tool are

1. analyzing ransomware (esp. initial infection) using Cuckoo Sandbox logs (more generally, ambient collected host logs) and generating features from the host behavior reports.
2. extracting the sequence of events (features) induced by ransomware given logs from (a few) hosts that are infected and (a potentially large amount of) ambient logs from presumably uninfected hosts;
3. ranking the most discriminating features (unique patterns) of malware and identifying malicious activity before data is encrypted by the ransomware.
4. creating graph visualizations of ET models to facilitate malware forensic efforts, and allowing operators to visualize discriminating features and their correlations.

We compare outputs with ransomware intelligence reports, and validate that our tool is robust to variations of input data. *TF-IDF* is the best method to identify discriminating features, and *ET* is the most time-efficient approach that achieves an average of 98% accuracy rate to detect the seven ransomware. This work builds on preliminary results of our workshop paper [8], which only considered feature extraction, only used TF-IDF, and only tested with one ransomware.

2 Background and Related Work

Ransomware. In contrast to the 2017 ransomware WannaCry that infected 300K machines across the globe, the majority of ransomware attacks in 2018 and 2019 have been targeting small businesses. These crypto-ransomware attacks usually use Windows API function calls to read, encrypt and delete files. Ransom messages are displayed on the screen after the ransomware infecting the host. This paper selects and analyzes seven recently disruptive ransomware attacks.

1. **WannaCry** *(2017)*, a ransomware with historic world-wide effect, was launched on May 12, 2017 [18]. The WannaCry dropper is a self-contained program consists of three components, an application encrypting and decrypting data; an encryption key file; and a copy of Tor. WannaCry exploits vulnerabilities in Windows Server Message Block (SMB) and propagates malicious code to infect other vulnerable machines on connected networks.
2. **DBGer** *(2018)*, a new variant of the *Satan* ransomware [19], scans the victim local network for vulnerable computers with outdated SMB services. DBGer incorporates a new open-source password-dumping utility, *Mimikatz*, to store credential of vulnearble computers [20]. The dropped Satan file is then executed to encrypt files of the infected computers with AES encryption algorithm. A text file `_How_to_decrypt_files.txt` containing a note of demands from the attackers is displayed on victim's screen.
3. **Defray** *(2017)*, a ransomware attack targets healthcare, education, manufacturing and technology industries [16]. Defray propagates via phishing emails with an attached *Word* document embedding an *OLE* package object. Once the victim executes the OLE file, the Defray payload is dropped in the `%TMP%` folder and disguises itself as an legitimate executable (e.g., `taskmgr.exe` or `explorer.exe`). Defray encrypts the file system but does not change file names or extensions. Finally, it deletes volume shadow copies of the encrypted files [15]. Defray developers encourage victims to contact them and negotiate the payment to get the encrypted files back [14].
4. **Locky** *(2016, 2017)* has more than 15 variants. It first appeared in February 2016 to infect Hollywood Presbyterian Medical Center in Los Angeles, California. The ransomware attackers send millions of phishing emails containing attachments of malicious code that can be activated via *Microsoft Word Macros* [11]. Locky encrypts data using RSA-2048 and AES-128 cipher that only the developers can decrypt data. In this research, we analyze the malicious behavior of a new variant of Locky ransomware called *Asasin*, which encrypts and renames the files with a `.asasin` extension.
5. **Cerber** *(2016–2018)* infected 150K Windows computers in July 2016 alone. Several Cerber variants appeared in the following two years have gained widespread distribution globally. Once the Cerber ransonware is deployed in the victim computer, it drops and runs an executable copy with a random name from the hidden folder created in `%APPDATA%`. The ransomware also creates a link to the malware, changes two Windows Registry keys, and encrypts files and databases offline with `.cerber` extensions [21,22].
6. **GandCrab** *(2018, 2019)*, a Ransomware-as-a-Service (RaaS) attack has rapidly spread across the globe since January, 2018. GandCrab RaaS online portal was finally shut down in June, 2019. During these 15 months, GandCrab creators regularly updated its code and sold the malicious code, facilitating attackers without the knowledge to write their own ransomware [23]. Attackers then distribute GandCrab ransomware through compromised websites that are built with *WordPress*. The newer versions of GandCrab use *Salsa20* stream cipher to encrypt files offline instead of applying RSA-2048 encryption technique connecting to the C2 server [24]. GandCrab scans

logical drives from `A:` to `Z:`, and encrypts files by appending a random Salsa20 key and a random initialization vector (IV) (8 bytes) to the contents of the file. The private key is encrypted in the registry using another Salsa20 key and the IV is encrypted with an RSA public key embedded in the malware. This new encryption method makes GandCrab a very strong ransomware, and the encrypted files can be decrypted by GandCrab creators only [25].

7. **nRansom** *(2017)* blocks the access to the infected computer rather than encrypting victim's data [13]. It demands ten nude photos of the victim instead of digital currency to unlock the computer. As recovery from nRansom is relatively easy, it is not a sophisticated malware but a "test" or a "joke".

Ransomware Pattern Extraction and Detection Works. Homayoun et al. [26] apply sequential pattern mining to find maximal frequency patterns (MSP) of malicious activities of four ransomware attacks. Unlike generating behavioral features directly from host logs, their approach summarizes activity using types of MSPs. Using four machine learning classifiers, the team found that atomic Registry MSPs are the most important sequence of events to detect ransomware attacks with 99% accuracy.

Verma et al. [27] embed host logs into a semantically meaningful metric space. The representation is used to build behavioral signatures of ransomware from host logs exhibiting pre-encryption detection, among other interesting use cases.

Morato et al. introduces REDFISH [28], a ransomware detection algorithm that identifies ransomware actions when it tries to encrypt shared files. REDFISH is based on the analysis of passively monitored SMB traffic, and uses three parameters of traffic statistics to detect malicious activity. The authors use 19 different ransomware families to test REDFISH, which can detect malicious activity in less than 20 seconds. REDFISH achieves a high detection rate but cannot detect ransomware before it starts to encrypt data. Our approach, discovering ransomware's pre-encryption footprint, promises a more accurate and in-time detection.

The Related Work section our preliminary work [8] includes works published previously to those above. As the more general topic of dynamic analysis is large and diverse, a comprehensive survey is out of scope, but many exist, e.g. [29].

3 Methodology

The proposed approach requires a set of normal (presumably uninfected) system logs and at least one log stream containing ransomware behavior. In this study, the seven ransomware executables introduced in Sect. 2 are deployed inside a realistic but isolated environment with a sandbox tool, Cuckoo [17], for harvesting reproducible and shareable host logs. The Cuckoo host logs are dynamic analysis reports outlining behavior (i.e., API calls, files, registry keys, mutexes), network traffic and dropped files

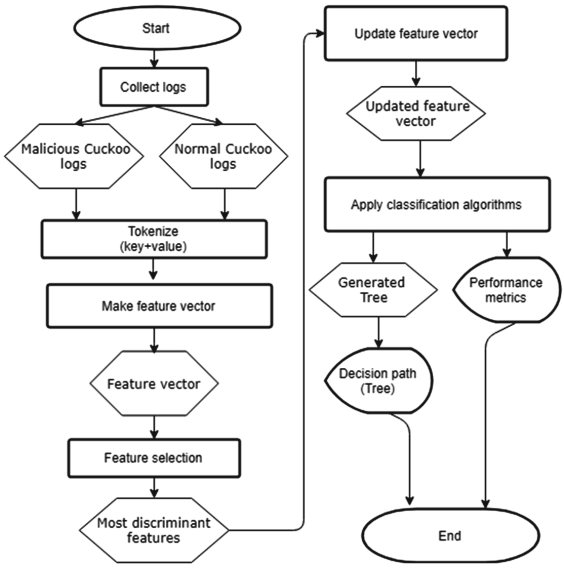

Fig. 1. Flowchart of research methodology

Meanwhile, Cuckoo also captures logs from scripted, emulated normal user activity such as reading and writing of executables, deleting files, opening websites, watching YouTube videos, sending and receiving emails, searching flight tickets, and posting and deleting tweets on Twitter (see [8]). The normal user and the ransomware events/behavior in the raw host logs produced by Cuckoo are then converted to features, and the three machine learning techniques are used to automatically obtain the most discriminating features from normal and ransomware-including logs. Afterwards, we discard the features that have little or no influence, and update the feature vector to reduce the search space of ET decision tree models. The decision tree graphs are created to present the most discriminating features of ransomware attacks. See flowchart in Fig. 1.

3.1 Feature Generation

To build features we only use the *enhanced* category and part of the *behavior* category of Cuckoo-captured logging output. The details of the feature building can be found in our previous work [8]. As malware often uses random names to create files, modules and folders, in this study, we augment paths of specific files to emphasize their names only. For example, `C:\\Windows\\system32\rsaenh.dll` is converted to a string "c:..rsaenh.dll". Here, ".." is used as a wild-card to avoid generating duplicated features that represent similar host behavior.

3.2 Discriminating Feature Extraction with Machine Learning

TF-IDF, Fisher's LDA and ET are algorithms used in this research to automatically extract the most discriminating features of ransomware from host logs.

TF-IDF, was defined to identify the relative importance of a word in a particular document out of a collection of documents [30]. Our TF-IDF application follow our previous work for accurate comparison. Given two sets of documents let $f(t, d)$ denote the frequency of term t in document d, and N the size of the corpus. The TF-IDF weight is the product of the Term Frequency, $\mathrm{tf}(t, d) = f_{t,d} / \sum_{t' \in d} f_{t',d}$ (giving the likelihood of t in d) and the Inverse Document Frequency, $\mathrm{idf}(t, D) = \log[N/(1 + |\{d \in D : t \in d\}|)]$ (giving the Shannon's information of the document containing t). Intuitively, given a document, those terms that are uncommonly high frequency in that document are the only terms receive high scores. We use log streams from infected hosts as one set of documents and a set of normal log streams as the other to apply TF-IDF; hence, highly ranked features occur often in (and are guaranteed to occur at least once in) the "infected" document, but infrequently anywhere else [8].

Fisher's LDA is a supervised learning classification algorithm that operates by projecting the input feature vectors to a line that (roughly speaking) maximizes the separation between the two classes [31]. For our application we consider a binary classification where one class (C_1) is comprised of the feature vectors $\{x_i\}_i \subseteq \mathbb{R}^m$ representing host logs that included ransomware, and the second class (C_2) are those vectors of ambient logs. We use this classifier for identifying the discriminating features between the classes. Consider the set $\{v^t x_i : x_i \in C_1 \cup C_2\} \subset \mathbb{R}$, which is the projection of all feature vectors to a line in $\mathbb{R}^m$ defined by unit vector v. Fisher's LDA identifies the unit vector v that maximizes $S(v) := [v^t(\mu_1 - \mu_2)]^2 / [v^t(\Sigma_1 + \Sigma_2)v]$ with μ_j, Σ_j the mean and covariance of $C_j, j = 1, 2$, respectively. $S(v)$ is the squared difference of the projected classes' means divided by the sum of the projected classes' variances. It is an exercise in linear algebra to see the optimal $v \propto (\Sigma_1 + \Sigma_2)^{-1}(\mu_1 - \mu_2)$. Geometrically, v can be thought of as a unit vector pointing from C_1 to C_2; hence, ranking the components of v by absolute values sorts the features that most discriminate the ransomware and normal activity.

Extremely Randomized Trees (ET) is a tree-based ensemble algorithm for supervised classification and regression. "It consists of randomizing strongly both attribute and cut point choice while splitting the tree node" [32]. In the extreme case, the algorithm provides "totally randomized trees whose structures are independent of the output values of the learning sample" [32,33]. The randomization introduces increased bias and variance of individual trees. However, the effect on variance can be ignored when the results are averaged over a large ensemble of trees. This approach is tolerant with respect to over-smoothed (biased) class probability estimates [32]. See the cited works for details.

4 Experimental Results

Experiment One: Extracting Discriminating Features from Host Logs. This experiment applies the machine learning approaches to extract the most discriminating features/behavior of each ransomware attack. In addition to obtaining a Cuckoo analysis report (raw behavior log) for each ransomware sample, Python scripts immitating various users' normal activities (such as reading, writing and deleting files, opening websites, etc.) are submitted to the Cuckoo sandbox to generate a large volume of normal reports.

Table 1 illustrates the most discriminating features of the seven ransomware attacks. The first column of the table (#) lists the name of seven ransomware. The second column (*Pattern*) presents the pre-encryption patterns (activities) of each ransomware attack obtained from the detailed ransomware technical (static) analysis produced by cybersecurity companies (e.g., FireEye [34]), security help websites (e.g., Bleeping Computer [35,36]) and malware research teams (e.g., The Cylance Threat Research [16]). The third column (*Feature*) presents the features extracted from the host logs using the proposed approaches that match the unique patterns of rasomware attacks. The last column (*Rank*) lists the TF-IDF, Fisher's LDA and ET rankings of the features that represent the unique patterns of the seven ransomware attacks. The features that have the largest TF-IDF and Fisher's LDA scores, or the non-leaf nodes (features) of the Extremely Randomized Trees that have smallest levels, are top-ranked discriminating features. For the ET algorithm, the features that are at the top of the tree contribute more to correctly classifying a larger portion of input logs. E.g., a feature with rank = 1 is one of the most indicative feature of the malware according to that algorithm. Ties are possible as the scores may be the same between multiple features. We use the rankings of these features to evaluate the efficiency of the proposed three machine learning methods. The methods that provide higher rankings of the selected features are more efficient than the approaches that yield a lower rank of the same feature.

We set a large `class_weight` parameter for the target class in *ExtraTreesClassifier* of Python's *Scikit-Learn* library to make the ET classifier biased to learn the pattern of malicious logs more meticulously. Therefore, some features representing the ransomware patterns are not selected as the nodes to compose the tree. In this scenario, we use "NA" to present the rankings of the feature that are not nodes in the tree. Details are elaborated by ransomware:

1. **WannaCry:** The six patterns of WannaCry before the attack encrypting data are presented in Table 1. All of these patterns can find WannaCry-generated features from the host logs. A total of 1,207 unique features have been extracted from host logs containing both normal and abnormal behavior, while only a small portion are resulting from WannaCry actions. The experimental results indicate that TF-IDF is better than the other two methods for identifying WannaCry's behaviors. The rankings generated by the ET classifier are slightly lower than the TF-IDF's. However, ET is more time efficient for extracting the most discriminating features from large volume of host logs,

Table 1. The most discriminating features of the seven ransomware attacks

#	Pre-Encryption Pattern	Feature	Rank		
			TF-IDF	LDA	ET
1. WannaCry	1. Import CryptoAPI from advpi32.dll	data_file+'advapi32.dll'+event+'load'+object+'library'	3	294	6
	2. Unzips itself to .wrny files	*.wnry	1	176	1
	3. Creates a registry, `HKEY_LOCAL_MACHINE\Software\WanaCrypt0r\wd`	api+'regcreatekeyexw'+arguments_1_value+'33554432' +category+'registry' *(subkey="Software\\WanaCrypt0r")*	6	177	NA
	4. Run 'attrib +h', to set the current directory as a hidden folder	data_file+'attrib +h .'+event+'execute'+object+'file'	6	298	11
	5. Run 'icacls . /grant Everyone:F /T /C /Q' to grant user permissions to the current directory	data_file+'icacls . ..q'+event+'execute'+object+'file'	6	298	11
	6. Import public and private RSA AES keys (000.pky, 000.eky) from t.wrny	data_file+'c:..00000000.pky'+event+'write'+object+'file'	6	298	11
2. DBCer	1. Drop ExternalBlue files at 'C:\Users\All Users\'	data_file+'c:..users'+event+'create'+object+'dir', data_file+'c:..allusers'+event+'create'+object+'dir', data_file+'c:..blue.exe'+event+'write'+object+'file', **... 22 various dropped file features...** data_file+'c:.. satan.exe' +event+'write'+object+'file', data_file+'c:..mmkt.exe'+event+'write'+object+'file'	9	125	11, 12
	2. Drop satan.exe on C drive and execute the file for encryption	data_file+'c:..satan.exe'+event+'write'+object+'file', data_file+'c:..satan.exe'+event+'execute'+object+'file'	9	125	11, 12
	3. Drop "KSession" file at %Temp%	data_file+'c:..ksession'+event+'write'+object+'file'	9	125	11
3. Defray	1. Import/Load Microsoft OLE from "ole32.dll"	data_file+'ole32.dll'+event+'load'+object+'library'	9	10	9
	2. Drop and execute "explorer.exe"	data_file+'explorer.exe'+event+'load'+object+'library'	17	93	NA
	3. Call ShellExecute to run as more privileged user to disable startup recovery and delete volume shadow copies	data_file+'c:..-hibernate-timeout-dc0'+event+'execute' +object+'file'	17	121	NA
4.Locky	1. Read and write 'PIPE\\wkssvc' and 'PIPE\lsarpc'	data_file+'pipe..wkssvc'+event+'write'('read')+object+'file', data_file+'pipe..lsarpc'+event+'write' ('read')+object+'file'	2 7	72 408	2 7
	2. Read network provider name	data_regkey+'hkey_local_machine.. networkprovidername'+event+'read'+object+'registry'	3	186	3
	3. Read the path to the network provider .dll file	data_regkey+'hkey_local_machine.. systworkproviderproviderpath'+event+'read'+object+'registry'	4	171	4
	4. Load the network provider 'ntlanman.dll" file	data_file+'c:..ntlanman.dll'+event+'load'+object+'library'	4	130	4
	5. Obtain the name ty Identifier	data_regkey+'hkey_users..s-1-5-21-1966058-1343024091 -1003name'+event+'read'+object+'registry'	5	408	5
5.Cerber	1. Create two .tmp files under a random folder	- - ent+'write'+object+'file' c. data_file+'c:..5572.tmp'+event+'write'+object+'file'	10	a.105 a.10 b.230 b.10 c.230 c.11	
	2. Find users profiles and read the	- - - rofilesdirectory'+event+'read'+object+'registry' b.data_regkey+'hkey_local_machine.. softlelistdefaultuserprofile'+even - - - hine.. softs-1-5-18profileimagepath' +event+'read'+ob - - - fileimagepath +event+'read'+object+'registry'	a.5 b.5 c.7 d.7	a.111 a.5 b.111 b.5 c.150 c.7 d.150 d.7	
	3. Read and load "rsaenh.dll"	a. data_regkey+'hkey_local_machine.. softaphic provid t+'read'+object+'registry' b. data_file+'c:..rsaenh.dll'+event+'read'+object+'file' c. data_file+'c:..rsaenh.dll'+event+'load'+object+'file'	a.3 b.1 c.6	a.79 b.15 c.119	a.3 b.1 c.6
	4. Obtain M	data_regkey+'hkey_local_machine.. cryptographymachineguid'+event+'read'+ob			
	nformation a. computer name b. session manager name c. domain name d. processor typ	- - - name'+event+'read'+object+'registry' b.data_regkey+'hkey_local_machine..sessionmanagername' +event+'read'+object+'registry' c. data_regkey+'hkey_local_machine..parametersdomain' +event+'read'+object+'registry' d.1 data_regkey+'hkey_local_ t+'read'+object+'registry' d.2 data_regkey+'hkey_local_machine..0identifier' +event+'read'+object+'registry' d.3 data_regkey+'hkey_local_machine.. systgersafeprocesssearc d'+object+'registry'	a.1 b.6 c.7 d.7	a.276 a.1 b.430 b.8 c.431 c.10 d.431 d.9	
	2. Copy the ransomware .exe file to %APPDATA%/Microsoft and ad unOnce key	a. data_file+'c:..lrcjty.exe'+event+'write'+object+'file' b. data_content+'..x00'+data_object+'n - - - +event+'write'+object+'registry'	7	431	a. 9 b.10
7.nRansomware	1. Create temprary directory in \%TEMP%\1.tmp\ tools\	data_file+'c:..tools'+event+**'create'+object+'dir'**	5	32	5
	2. Download and write following files: a ransom.exe) b. a media control file (i.e.,interop.wmplib. le (i.e., your-mom-gay.mp3)	a.data_file+'c:..**nransom.exe'+event+*'write'***+object+'file' b.data_file+'c:..**interop.wmplib.dll'+event+*'write'***+object+'file' c. -			a.4 b.4 c.4
	3. Execute the executable (i.e., nransom.exe) using comma	- ***'execute'***+object+'file' b. data_file+'c:..**cmd'+event+*'execute'***+object+'file'	a.6 b.6	a.60 b.60	a.7 b.6
	4. Play the looped song using the downloaded audio file (i.e., your-mom-g	-			
	orary folders with the downloaded files	data_file+'c:..**1.tmp'+event+*'delete'***+object+'dir' +object+'file'	6	60	6

which requires only 215 features (nodes) to make decisions (i.e., WannaCry or Normal). Therefore, the results suggest using TF-IDF to analyze the few infected hosts' logs in an attempt to produce shareable threat intelligence reports and using the ET algorithm to obtain pre-encryption detection capabilities. This experiment also illustrates that the top-ranked features generated by Fisher's LDA are quite different from the other two techniques. Most of the top-ranked features are normal activities. Features representing WannaCry's patterns are listed as low as #200. Additionally, we notice that the loading and reading events of the *rsaenh.dll* module are ranked highly (i.e., #2 and #4 for TF-IDF and #3 and #8 for ET). The module implements the Microsoft enhanced cryptographic service provider for WannaCry to encrypt the victim's data with 128-bit RSA encryption. These two top ranked features are not listed in our table, as they are not discriminating features to identify WannaCry attacks from other crypto-ransomware attacks.

2. **DBGer:** The three unique patterns of DBGer ransomware reported by [37] are presented in Table 1. `dbger.exe`, the mother file of DBGer, first creates the `C:\Users\AllUsers` folder, drops *EternalBlue* and *Mimikatz* executables in the new folder, and then saves *satan.exe* into the *C* drive. A file named `KSession` is dropped to `C:\Windows\Temp\` for storing the host ID. TF-IDF and Fisher's LDA rank 1,104 features generated from normal and DBGer Cuckoo reports. The ET classifier builds the decision tree using 216 of the 1104 features. The three DBGer features are ranked highly. TF-IDF yields a highest ranking of the three features, which is better than the other two methods. ET is more time efficient. However, there are many features ranked higher than the ranking of the three features, but they are normal activity. E.g., dynamic link library (DLL) files `kernel32.dll` and `advapi.dll` are on the top of the three rankings, but are not discriminating features for DBGer.
3. **Defray:** The three unique patterns of Defray are loading the *ole32.dll* file, dropping and executing the ransomware executable file *explorer.exe*, and executing a shell command. The three machine algorithms rank the first feature "loading the ole32.dll file" #9 among the total 1,243 features. As Defray's executable file is disguised as a Windows Internet Explorer, all of the three methods struggle to distinguish it from the normal activities. The second feature therefore is not selected to build the ET model, and its TF-IDF and Fisher's LDA weights are much lower than the first feature's. The three machine learning approaches rank another three features (as shown in Table 2) highest among the 1243 features. These features represent unique malicious activities performed by Defray, thus, they are discriminating features to distinguish Defray from other ransomware. However, none of these three patterns are discussed in Defray manual analysis reports [14–16].
4. **Locky:** We execute *Asasin Locky*, a 2017 variant of Locky ransomware in the Cuckoo sandbox, collect and analyze its behavior using our tool. The static analysis reports [9,11] indicate that after being deployed, Locky's executable file disappears. Its dropped copy `svchost.exe` is executed from the `%TEMP%` folder. However, our tool generates features from the behavior logs and presents that Asasin Locky does not drop the executable file.

Instead, the attack modifies the workstation services `\PIPE\wkssvc` launched by the `svchost.exe` process. As a member of the Cryptowall family, Asasin Locky also modifies `PIPE\lsarpc`, a file communicates with the Local Security Authority subsystem [38]. The attack then reads network provider name and the path to the Network Provider DLL file from registry by loading the network provider `ntlanman.dll`. Registry is retrieved by Asasin Locky to obtain the name of the Security Identifier. TF-IDF and ET provides the same and higher rankings for these five features from a total 1,047 normal and ransomware features. These two methods both rank `rsaenh.dll` as the top feature; however, this feature is not a unique pattern for Asasin Locky.

5. **Cerber:** This ransomware copies itself as `cerber.exe` to the hidden `%APPDATA%` folder, creates a directory with a random name, and drops two `.tmp` files [10]. Cerber also escalates its privilege to admin level and reads profiles from the users' profile image paths. Afterwards, Cerber finds the image path of `rsaenh.dll`, reads and loads the DLL file to encrypt data. Cerber obtains the Machine GUID (globally unique identifier) and uses its fourth part as the encrypted files' extension. The Cerber sample tested has an extension of `93ff`. The three methods rank the total 1,137 features. ET selects 145 features to composes the decision tree. TF-IDF and ET provides similar and higher rankings of the discriminating features than Fisher's LDA's.
6. **GandCrab:** This experiment uses Gandcrab V2.3.1, a variant that scans the victim machine and collects information of user name, domain name, computer name, session manager name and processor type [12]. The execution is terminated if the ransomware finds the system language is Russian or the victim machine installed specific anti-virus (AV) software. Otherwise, it copies the executable file into `%APPDATA%/Microsoft` and adds an entry of the copied executable file path to the `RunOnce` key as a one-time persistence mechanism. GandCrab then decrypts the ransom notes and generate RSA keys for encryption. After encrypting data, the malware uses Windows' NSLOOKUP tool to (1) find IP address of the GandCrab's C2 (command and control) server; and (2) communicate with the C2 server (i.e., sending information collected from the victim's machines to the C2 server and/or receiving commands from the C2 server). Table 1 presents two unique pre-encryption patterns of GandCrab V2.3.1. TF-IDF and ET rank them highly among 1,017 features. The rankings of these features are much lower by Fisher's LDA.
7. **nRansom:** This attack first creates a subfolder in `%TEMP%` with a random name ended with `.tmp`. In our experiment, the subfolder is named `1.tmp`. nRansom drops an executable file (i.e., `nransom.exe`) and two Windows Media Player control library files (i.e., `Interop.WMPLib.dll` and `AxInterop.WMPLib.dll`) in `1.tmp`. An audio file `your-mom-gay.mp3` is dropped in `1.tmp  Tools`. Then `nransom.exe` is executed through the command prompt `cmd.exe`. After locking the victim's computer screen, nRansom plays a looped song from the dropped mp3 file, and deletes the subfolders and dropped files. TF-IDF and ET both rank the five discriminating features of nRansom highly among 1046 features. 55 features are used for composing ET.

Table 2. Static analysis missed unique patterns and their behavioral features

#	Pattern	Feature
Defray	1. Read *CliEgAliases.mof* and *Cli.mof* from `C:\WINDOWS\System32\Wbem`	data_file+'c:..cliegaliases.mof' +event+'read'+object+'file' data_file+'c:..cliegaliases.mft'+event+'read'+object+'file'
	2. Read/write temporary files (tmp1-tmp3.tmp) in %TMP% folder	data_file+'c:..tmp2.tmp'+event+'write'+object+'file' data_file+'c:..tmp2.tmp'+event+'read'+object+'file'
	3. Read registries to obtain the infected host's host name, domain and active computer name	data_regkey+'hkey_local_machine..parametershostname'+event+'read'+object+'registry' data_regkey+'hkey_local_machine..parametersdomain'+event+'read'+object+'registry' data_regkey+'activecomputernamecomputername'+event+'read'+object+'registry'
Locky	Embedding object created in one application into another application (e.g., embedding an excel file inside a word	data_file+'ole32.dll'+event+'load'+object+'library'
Cerber	Embedding object created in one application into another application (e.g., embedding an excel file inside a word file).	data_file+'ole32.dll'+event+'load'+object+'library'
Gandcrab	1. Import CryptoAPI from advpi32.dll	data_file+'advapi32.dll'+event+'load'+object+'library'
	2. Control certain API functions of windows shell	data_file+'shell32.dll'+event+'load'+object+'library'
	3. Implements certificate and cryptographic messaging functions	data_file+'crypt32.dll'+event+'load'+object+'library'
nRansom	1. Read the batch file that executes commands with windows command prompt (i.e., cmd.exe)	data_file+'c:..2.bat'+event+'read'+object+'file'
	2. Control Windows Media Player	data_file+'c:..axinterop.wmplib.dll'+event+'write'+object+'file'
	3. Load the "uxtheme.dll" library	data_file+'uxtheme.dll'+event+'load'+object+'library'

Ransomware Unique Patterns Missed from Manual Analysis. As discussed above, besides the patterns obtained from Defray's threat intelligence reports, the three features shown in Table 2 are also unique behavior to distinguish Defray attacks. From the dynamic analysis provided by our methodology, we also found that many ransomware attacks have similar patterns. For example, Defray, Locky and Cerber all conduct an event to load the `ole32.dll` file. However, neither Locky nor Cerber's static analysis have mentioned this pattern. Similarly, manual analysis of GandCrab does not discuss the malware sample has imported CryptoAPI from `advapi32.dll`, which is also a discriminating feature of WannaCry attacks. Thus, our tool provides automated—more efficient and without reliance on security experts—and better quality malware behavior analysis.

Table 3. WannaCry discriminating feature ranking with varying normal data

TF-IDF				LDA				ET			
Feature	Rank			Feature	Rank			Feature	Rank		
	C1	C2	C3		C1	C2	C3		C1	C2	C3
***.wnry**	1	1	1	api+'ntreadfile'+arguments_1_value+'0x0000018c'+category+'filesystem'	1	1	1	***.wnry**	1	1	1
data_file+'c:..**rsaenh.dll**'+event+'read'+object+'file'	2	2	2	api+'regclosekey'+category+'registry'	2	5	5	api+'ntwritefile'+arguments_1_value+'0x00000080'+category+..	2	2	2
data_file+'**advapi32.dll**'+event+'load'+object+'library'	3	3	3	api+'findfirstfileexw'+category+'filesystem'	3	2	2	data_file+'c:..**rsaenh.dll**'+event+'read'+object+'file'	3	3	3
data_file+'**kernel32.dll**'+event+'load'+object+'library'	4	4	4	api+'ntqueryinformationfile'+arguments_1_value+'..x00'+..	4	6	6	api+'ntreadfile'+arguments_1_value+'0x00000090'+category+..	4	4	4
data_regkey+'hkey_users..+event+'read'+object+'registry'	5	5	5	api+'ntreadfile'+arguments_1_value+'0x0000009c'+category+'filesystem'	5	7	7	api+'ntreadfile'+arguments_1..+category+'filesystem'	5	5	5
data_regkey+'hkey_local_machine..softer (prototype)image path'+event+'read'+object+'registry	6	6	6	api+'ntreadfile'+arguments_1_value+'0x00000188'+category+'filesystem'	6	16	15	api+'ntdeviceiocontrolfile'+arguments_1_value+'0x00000090'+category+'filesystem'	6	6	6
data_file+'**rsaenh.dll**'+event+'load'+object+'library'	7	7	7	api+'ntreadfile'+arguments_1_value+'0x00000178'+category+'filesystem'	7	15	16	data_regkey+'hkey_users..+event+'read'+object+'registry'	7	7	7
data_regkey+'activecomputernamecomputername'+event+'read'+object+'registry'	8	12	12	api+'ntreadfile'+arguments_1_value+'0x0000017c'+category+'filesystem'	8	18	18	data_regkey+'hkey_local_machine..softer (prototype)image path'+event+'read'+object+'registry'	8	8	8
data_file+'c:...taskdl.exe'+event+'write'+object+'file'	9	9	9	api+'regqueryvalueexw'+arguments_1_value+'0'+category+'registry'	9	14	14	data_file+'**kernel32.dll**'+event+'load'+object+'library'	9	9	9
data_file+'c:...taskse.exe'+event+'write'+object+'file'	10	10	10	api+'regopenkeyexa'+arguments1_value+'0x000000a4'+category+'registry	10	19	19	data_file+'**advapi32.dll**'+event+'load'+object+'library'	10	10	10

Experiment Two: Ransomware Feature Rankings with Varying Normal Activities. This experiment aims to validate that the rankings of the seven ransomware discriminating features are not influenced by varying the number of normal logs. To validate the hypothesis, we calculate the TF-IDF, Fisher's LDA and ET weights of the ransomware features in the following three scenarios.

- Case 1 (C1): Using Experiment One's normal logs as the baseline.
- Case 2 (C2): Adding 30% additional new normal host logs into training data.
- Case 3 (C3): Adding 60% more new normal host logs into training data.

Table 3 presents the top ten features of WannaCry that are calculated by the three machine learning methods when the ambient logging data are different. The experimental results present that the ET method is robust to provide the same rankings of the top ten features under the three tested scenarios. TF-IDF is less robust than ET, but Fisher's LDA provides completely different rankings of the top ten features in three different scenarios. Similar results were found when analyzing the top-ranked features of the other six ransomware attacks. Therefore, the ET algorithm is more robust to varying training data containing different quality and quantity of normal activity.

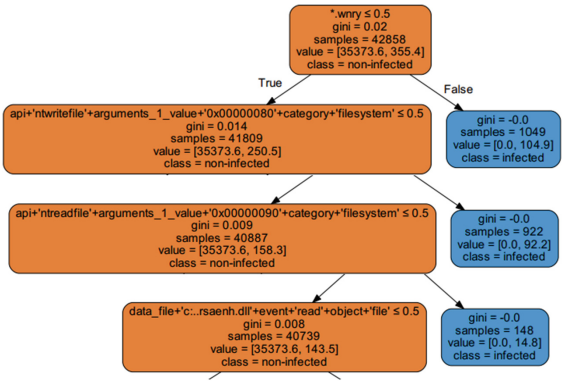

Fig. 2. Decision path based on the training logs showing how the most discriminating features are correlated in the decision making process.

Experiment Three: Ransomware Early Detection. The ET decision tree classifier is applied to detect the seven ransomware before encryption from a large majority of non-malicious activity. Table 4 presents the detection rate of the seven ransomware attacks. Note that while recall varies, meaning the method produces false negatives, precision is always perfect, meaning there are no false positives. In terms of overall performance metrics, the detection model Gandcrab performs the best and DBGer performs the worst. We also create graphs of each decision tree to better interpret and visualize the detection results. Using WannaCry attack as an example, Fig. 2 displays first three levels of the decision tree. The brown non-leaf nodes (rectangular boxes) represent the features of normal activity and the blue non-leaf nodes represent features induced by WannaCry.

Table 4. ET early detection results

Ransomware	Accuracy	Precision	Recall	F-Score
WannaCry	0.918	1	0.717	0.835
DBGer	0.987	1	0.308	0.471
Defray	0.994	1	0.992	0.996
Locky	0.997	1	0.806	0.893
Cerber	0.987	1	0.505	0.671
GandCrab	0.999	1	0.997	0.999
nRansom	0.994	1	0.382	0.553

By retrieving the blue nodes on the top of the decision tree, we can identify WannaCry's discriminating features. The correlation coefficients of these features are provided in non-leaf boxes. The graphs facilitate malware forensics analysis and allow operators to visualize disruptive activity and determine the damages induced by the malware for proposing an optimal protection and response plan.

5 Conclusion

We develop an automated ransomware pattern-extraction and early detection tool that extracts the sequence of events induced by seven ransomware attacks, identifies the most discriminating features using three machine learning methods, and creates graphs to facilitate forensic efforts by visualizing features and their correlations. The experimental results present that TF-IDF feature ranking yields the most accurate identification of the ransomware-discriminating features, while the ET method is the most time efficient and robust to the variation of inputs. Notable, discriminating features are automatically promoted by this method that malware analysis reports failed to identify.

As the target application is using this to analyze real host logs collected by SOCs, future research to test our tool using real-world host-based data captured in enterprise networks to determine conditions for success. Moreover, large enterprises generate large volumes of host data. The offline machine learning techniques used in this paper—creating features from host logs, determining malware discriminating features and detecting attacks—may not scale. Future research using online machine learning technique (e.g., incremental decision tree) and deep learning methods (e.g., LSTMs) can enhance the tool.

Acknowledgements. Special thanks to the reviewers that helped polish this document, including Michael Iannacone. Research sponsored by the Laboratory Directed Research and Development Program of Oak Ridge National Laboratory, managed by UT-Battelle, LLC, for the U. S. Department of Energy, and by the National Science Foundation under Grant No.1812599. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

1. Davis, J.: 71% of ransomware attacks targeted small businesses in 2018, March 2019. https://healthitsecurity.com/news/71-of-ransomware-attacks-targeted-small-businesses-in-2018
2. Dobran, B.: Definitive guide for preventing and detecting ransomware (2019). https://phoenixnap.com/blog/preventing-detecting-ransomware-attacks
3. Freed, B.: One year after atlanta's ransomware attack, the city says it's transforming its technology (2019). https://statescoop.com/one-year-after-atlantas-ransomware-attack-the-city-says-its-transforming-its-technology/
4. Olenick, D.: Atlanta ransomware recovery cost now at $17 million, reports say (2018). https://www.scmagazine.com/home/security-news/ransomware/atlanta-ransomware-recovery-cost-now-at-17-million-reports-say/

5. Bridges, R.A., Iannacone, M.D., Goodall, J.R., Beaver, J.M.: How do information security workers use host data? A summary of interviews with security analysts. arXiv preprint 1812.02867 (2018)
6. Goodall, J., Lutters, W., Komlodi, A.: The work of intrusion detection: rethinking the role of security analysts. In: AMCIS 2004 Proceedings, p. 179 (2004)
7. Werlinger, R., Muldner, K., Hawkey, K., Beznosov, K.: Preparation, detection, and analysis: the diagnostic work of it security incident response. Inf. Manag. Comput. Secur. **18**(1), 26–42 (2010)
8. Chen, Q., Bridges, R.A.: 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, December 2017
9. Malwarebytes LABS: Look into locky ransomware, July 2016. https://blog.malwarebytes.com/threat-analysis/2016/03/look-into-locky/
10. Gao, W.: Dissecting Cerber ransomware, July 2017. https://www.ixiacom.com/company/blog/dissecting-cerber-ransomware
11. Doevan, J.: Locky virus, how to remove (2018). https://www.2-spyware.com/remove-locky-virus.html
12. Cisco's Talos Intelligence Group Blog: Gandcrab Ransomware Walks its Way onto Compromised Sites (2018). https://blog.talosintelligence.com/2018/05/gandcrab-compromised-sites.html. Accessed 25 Aug 2018
13. This Ransomware Demands Nude instead of Bitcoin - Motherboard (2017). https://motherboard.vice.com/en_us/article/yw3w47/this-ransomware-demands-nudes-instead-of-bitcoin. Accessed 24 Aug 2018
14. Defray ransomware sets sights on healthcare and other industries, August 2017. https://www.trendmicro.com/vinfo/us/security/news/cyber-attacks/defray-ransomware-sets-sights-on-healthcare-and-other-industries
15. Crowe, J.: Alert: Defray ransomware launching extremely personalized attacks, August 2017. https://blog.barkly.com/defray-ransomware-highly-targeted-campaigns
16. Threat Spotlight: Defray Ransomeware Hits Healthcare and Education (2017). https://threatvector.cylance.com/en_us/home/threat-spotlight-defray-ransomware-hits-healthcare-and-education.html. Accessed 16 Aug 2018
17. Cuckoo Sandbox - Automated Malware Analysis. https://cuckoosandbox.org/. Accessed 26 Aug 2018
18. Perlroth, N.: Boeing possibly hit by 'WannaCry' malware attack, March 2018. https://www.nytimes.com/2018/03/28/technology/boeing-wannacry-malware.html
19. Lemos, R.: Satan ransomware adds more evil tricks, May 2019. www.darkreading.com/vulnerabilities---threats/satan-ransomware-adds-more-evil-tricks/d/d-id/1334779
20. Cimpanu, C.: DBGer ransomware uses EternalBlue and Mimikatz to spread across networks (2018). https://www.bleepingcomputer.com/news/security/dbger-ransomware-uses-eternalblue-and-mimikatz-to-spread-across-networks/
21. Barkly Research: Cerber ransomware: everything you need to know, March 2017. https://blog.barkly.com/cerber-ransomware-statistics-2017
22. Malwarebytes LABS: Cerber ransomware: new, but mature, June 2018. https://blog.malwarebytes.com/threat-analysis/2016/03/cerber-ransomware-new-but-mature/
23. Tiwari, R.: Evolution of GandCrab ransomware, April 2018. https://www.acronis.com/en-us/articles/gandcrab/

24. Salvio, J.: GandCrab V4.0 analysis: new shell, same old menace (2018). https://www.fortinet.com/blog/threat-research/gandcrab-v4-0-analysis-new-shell-same-old-menace.html
25. Mundo, A.: GandCrab ransomware puts the pinch on victims, July 2018. https://securingtomorrow.mcafee.com/mcafee-labs/gandcrab-ransomware-puts-the-pinch-on-victims/
26. Homayoun, S., Dehghantanha, A., Ahmadzadeh, M., Hashemi, S., Khayami, R.: Know abnormal, find evil: frequent pattern mining for ransomware threat hunting and intelligence. IEEE Trans. Emerg. Top. Comput. (2019)
27. Verma, M.E., Bridges, R.A.: Defining a metric space of host logs and operational use cases. In: 2018 IEEE International Conference on Big Data (Big Data), pp. 5068–5077, December 2018
28. Morato, D., Berrueta, E., Magaña, E., Izal, M.: Ransomware early detection by the analysis of file sharing traffic. J. Netw. Comput. Appl. **124**, 14–32 (2018)
29. Egele, M., et al.: A survey on automated dynamic malware-analysis techniques and tools. ACM Comput. Surv. (CSUR) **44**(2), 6 (2012)
30. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Inf. Process. Manag. **24**(5), 513–523 (1988)
31. Welling, M.: Fisher linear discriminant analysis. Department of Computer Science, University of Toronto, vol. 3, no. 1 (2005)
32. Geurts, P., Ernst, D., Wehenkel, L.: Extremely randomized trees. Mach. Learn. **63**(1), 3–42 (2006)
33. Islam, S.R., Eberle, W., Ghafoor, S.K.: Credit default mining using combined machine learning and heuristic approach. In: Proceedings of the 2018 International Conference on Data Science (ICDATA), pp. 16–22. ACSE (2018)
34. Wannacry Malware Profile - FireEye (2017). https://www.fireeye.com/blog/threat-research/2017/05/wannacry-malware-profile.html. Accessed 10 Aug 2018
35. DBGer Ransomware Uses EternalBlue and Mimikatz to Spread Across Networks (2017). https://www.bleepingcomputer.com/news/security/dbger-ransomware-uses-eternalblue-and-mimikatz-to-spread-across-networks/. Accessed 10 Aug 2018
36. Locky Ransomware Switches to the Asasin Extension via Broken Spam Campaign (2017). https://www.bleepingcomputer.com/news/security/locky-ransomware-switches-to-the-asasin-extension-via-broken-spam-campaigns/. Accessed 21 Aug 2018
37. Munde, S.: Satan ransomware raises its head again! June 2018. https://blogs.quickheal.com/satan-ransomware-raises-head/
38. Monika, Zavarsky, P., Lindskog, D.: Experimental analysis of ransomware on Windows and Android platforms: evolution and characterization. Procedia Comput. Sci. **94**, 465–472 (2016)