Understanding the Behavior of Ransomware: An I/O Request Packet (IRP) Driven Study on Ransomware Detection against Execution Time

Md. Ahsan Ayub and Ambareen Siraj
Department of Computer Science, Tennessee Tech University, Cookeville, USA
Emails: mayub42@tntech.edu, asiraj@tntech.edu

***Abstract*—Industries of diverse sizes, ranging from retail to critical infrastructure, are experiencing a worldwide upswing in ransomware attacks. On a daily basis, ransomware researchers encounter fresh samples and uncover novel ransomware families in the wild. This research investigates ransomware's I/O Request Packet (IRP), a low-level file system I/O log, to understand their behavior. We analyze IRP logs of 383 ransomware samples belonging to 21 families to execute these tasks. To evaluate our schemes' capabilities on detection against execution time, we report our empirical findings between 15 and 40 minutes of IRP logs, whereas each sample covers 90 minutes of logs on average. By utilizing one-class classification algorithms, e.g., One-Class SVM, Isolation Forests, and Local Outlier Factor (LOF), we demonstrate the identified sequences successfully discover new ransomware upon which the classifiers were not trained. We achieve exceptional experimental results in identifying ransomware families by applying Decision Trees, Random Forests, Extra Trees, and Bagging classifiers. To highlight, we at best obtain an accuracy of 93.94%, precision score of 93.27%, recall score of 91.28%, and F1 score of 91.90%.**

***Index Terms*—Dynamic Analysis, Ransomware, Machine Learning**

I. Introduction

Amidst an increasing frequency of severe cyber attacks targeting business organizations, ranging from small to large enterprises, ransomware has emerged as a prominent and highly consequential threat. These attacks can potentially paralyze entire organizations, leading to irreversible damage. As a result, ransomware has become a focal point of extensive research in the domain of malware analysis. Security researchers have been diligently working since 2015 to not only prevent ransomware from infiltrating victim machines but also to propose innovative frameworks for early detection.

Remarkably, little attention has been given to understanding ransomware behavior solely through analyzing I/O Request Packet (IRP) logs, collected during dynamic analysis throughout execution. To address this knowledge gap, we conduct rigorous empirical research on low-level I/O data obtained from diverse ransomware families, specifically focusing on partitioning the datasets into 5-minute time frames. Through a meticulous examination of file-based I/O operations in 21 ransomware families, our findings reveal a noteworthy trend: the majority of encryption tasks are executed within the initial 40 minutes of the ransomware's life cycle. Fig. 1 graphically illustrates the number of file objects that ransomware processes during I/O operations on the file system (e.g., read, write, modify) based on data extracted from 383 ransomware samples.

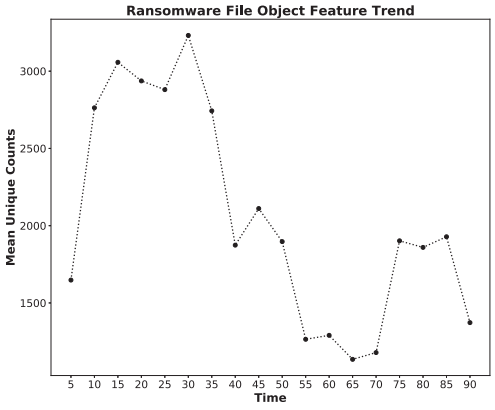

Fig. 1. A time series trend analysis graph on mean unique counts of 21 ransomware families' File Object feature from the I/O Request Packet logs

These insights bolster our confidence in developing ransomware detection schemes that can effectively learn and identify ransomware behavior within the first 40 minutes of its encryption pattern, as evident from the IRP logs. Detecting its presence early on accentuates the effectiveness of our proposed detection approaches and provides defenders with valuable time to respond promptly and effectively.

Research Questions (RQ). Our study asks the following two important research questions to evaluate its effectiveness:

RQ1. Is there any distinguishable pattern(s) present during ransomware encryption?

RQ2. How effectively is it possible to identify the families of ransomware *early enough* during its infection through continuous monitoring of IRP logs?

Major Contributions. The major contributions are as follows:

- We perform Time Series Analysis on 383 ransomware samples, belonging to 21 ransomware families, to learn the encryption behavior throughout their execution.
- We extract five notable sequences that significantly set ransomware apart from the benign processes.

979-8-3503-3912-3/23/$31.00 ©2023 IEEE
DOI 10.1109/CIC58953.2023.00018

- We utilize the One-Class Classification to discover new ransomware based on the counts of such extracted sequences from different time chunks of the IRP log.
- We employ several machine learning classification algorithms to empirically investigate their performances and report our findings on multiclass classification tasks.

Paper Organization. The remainder of the paper is organized as follows. Section 2 covers the details of the I/O request packet. Section 3 discusses the dataset used for our study, including the description of data processing tasks, and describes the empirical study of our work to highlight the strategies we incorporated to address the important research questions. We attempt to answer the research questions in Section 4. Then, we present a case study to show a comparison between ransomware and benign users' behavior in Section 5. The discussion of this research, along with its limitations and future work to further improve upon this conducted research, is included in Section 6. Section 7 summarizes the paper and its contributions.

II. I/O Request Packet (IRP)

We base our study of ransomware detection on I/O Request Packet (IRP), a low-level file system log. The structure of IRP can be discussed in the following four categories:

A. Types of IRP Operations

Three types of I/O operations can be triggered from user space: IRP; FIO (Fast I/O) – designed to transfer data between user buffer and system cache directly; and FSF (File System Filter) – designed to support IRP operations on file system[1]. We notice another type of IRP operation, *ERR*, in the dataset. It indicates that the I/O manager returns failure I/O operation status; hence, we ignore such logs in this case study.

B. Process based Features

Each IRP log resembles the I/O operation of a process initiated from the user space and recorded in the kernel space. Several pieces of information regarding a process in the IRP include Process ID, Process Name, Thread ID, and Parent ID. It is worth mentioning that such IDs are only valid as long as the process is active. For example, the Operating System (OS) allocates a certain set of IDs to a process upon its starting execution. When the process completes its actions, such IDs are freed, and the OS can reallocate the same ID(s) to another process. All this to say is that the process ID (or any given ID) is not unique in any captured ransomware sample's IRP dataset. Each IRP log presents two additional pieces of information regarding the process: pre-operation time – the timestamp of a process that starts its operation for a given IRP request; and post-operation time – the timestamp of the process competes for its IRP request with either a failure or a success status.

[1]IRPs Are Different From Fast I/O: https://docs.microsoft.com/en-us/windows-hardware/drivers/ifs/irps-are-different-from-fast-i-o

C. Flag based Features

The IRP structure provides several flag-based features, or categorical variables in other words, to cover additional information about the type of task a process carries out. It includes IRP Flag, IRP Major Operation Type, IRP Minor Operation Type, Status, Inform, Transaction, and Argument 1-6. IRP Flag feature comprises four special flags: No Cache, Paging I/O, Synchronous API, and Synchronous Paging I/O, along with a 32-bit Hexadecimal value. IRP Major Operation Type feature indicates the type of the IRP operation executed by the process, *e.g.*, read, write, close, etc. IRP Minor Operation Type feature also shows a process's IRP operation type, *e.g.*, query directory, start/remove an I/O device, etc.; however, these features depend on each other. To further explain, IRP Minor Operation Type may or may not exist when a categorical value is available at IRP Major Operation Type. Still, we do not find any example where a categorical value of IRP Minor Operation is present while the IRP Major Operation Type feature's value is null. Status is another feature with a 32-bit Hexadecimal value that is designed to map the IRP operation request to a human-readable format, *e.g.*, success, abandoned, alerted, timeout, etc. Microsoft provides most of the possible flag values' definitions in its documentation[2].

D. File System based Features

The IRP structure has a set of features representing intrinsic pieces of information, such as File Object, Device Object, File Name, Buffer Length, and Entropy, as a process interacts with the machine's file system. While a process accesses one of the files in the file system, the IRP records the objects' locations created by the process with the file's name as string datatype. Buffer length (a float datatype) and Entropy (a float datatype between 0 and 1) features indicate the portion of the file is written on memory and modified from its previous state, respectively, from each IRP request.

III. Experimental Methodology

This section describes the experimental methodology to answer the research questions.

A. Dataset Construction

We acquire the dataset from Continella *et al.* [6]. Their research proposed an intuitive ransomware detection framework, ShieldFS, in 2016 that successfully identified the signs of ransomware. Additionally, it detected the ransomware-like behavior of a process by monitoring its IRP operations. To facilitate the file recovery feature in its framework, the authors built a protection layer, and its job was to create a copy of the file when a triggered process would want to interact. Each running process had to go through an investigation phase before being given the privilege to read/write access to the file system. If the process was flagged as malicious, SheildFS

[2]https://github.com/microsoft/Windows-driver-samples/blob/8fb512ac674df5ba129a69906d450f2a1361136d/filesys/miniFilter/minispy/user/mspyLog.h

would enable the Operating System to kill it and revert the file system to its original state as it kept the copy of the file(s).

To obtain knowledge of benign user behavior, the authors in this paper performed a large-scale IRP data collection generated by benign applications with the help of an IRPLogger. The researchers built this data-collection agent to capture the day-to-day tasks of 11 voluntary machines used by home, office, and developer users for several weeks. On the other side, leveraging the same tool installed on a Windows 7 (64-bit) machine, the authors collected 383 active ransomware samples' IRP logs during its run-time execution. Each ransomware sample's captured IRP dataset covers some of the common utility applications' IRP logs, such as Adobe Reader, Microsoft Office, Web Browsers, and Media Player. In other words, not every process in the ransomware IRP dataset is malicious. The authors in [6] claimed that only the ransomware processes in each dataset interacted with the file system. Following this claim, we isolate malicious processes.

B. Data Processing

After we acquire the IRP dataset and understand its structure, we begin necessary data processing tasks for every feature, such as trimming extra spaces in string-based variables, one hot encode categorical variables [26], etc. The prime functions are to extract the ransomware process(es) from each ransomware sample's IRP dataset and discover to which family it belongs. The detailed discussion on how we perform such tasks is as follows:

Ransomware Process Extraction. We introduce a couple of new features in the dataset: *Operation Time Elapsed* – the difference between Post-Operation Time and Pre-Operation Time of each IRP request in seconds; and *Total Unique Files Accessed* – a count-based feature derived from the activity life span of each process with the help of File Name feature. To further describe, we group the dataset based on Process Name and Process ID features from which we can compute the active period and the total count of unique files accessed to isolate process-wise activities. With this aggregated version of each sample's dataset, we identify the active processes for the longest time and access a significantly higher number of files for every case. Thus, we separate malicious processes from benign ones. Then, we take a step back and label each record in the IRP dataset with either malicious or benign in *Class* feature. During the experimentation on ransomware analysis and collection of IRP logs in [6], no user action(s) was observed. We do not find any benign process accessing regular files (*e.g.*, doc, pdf, etc.) stored in the file system. However, we notice benign processes generated by Mozilla Firefox (a web browser), for instance, perform IRP operation on Windows system drive folders. Having said that, we consider the extracted benign processes from the logs as the processes that remain active in an idle state of the machine where a user does not interact with their device.

Ransomware Family Labeling. We proceed with identifying the family name of all 383 ransomware samples. Each sample is represented uniquely by its SHA256 hashes. We utilize VirusTotal API Engine [30] to scan the samples' hashes and receive a complete scan report containing results aggregated from many Anti-Virus (AV) tools, such as Kaspersky, Symantec, etc. If the scanned sample was detected malicious, the AV engines labeled it with a malware family name, *i.e.*, Kaspersky generated the label *Yakes* for the ransomware samples having "00ce22ce923e246990e43289b8b5b8191cbfc28dbee6d30b66226df0aa14b7bd" SHA-256 hash. We observed that the labels provided by different AV engines were not the same for the same signature hashes. Therefore, we explore a generalized approach to assign one family label to samples from the same family as scanned by different AV engines. Hence, we make use of AVClass - a malware labeling tool [27]. We feed the scan report for all the samples in a JSON file to the AVClass tool. Then, the tool assigns one (most probable) family name to a set of similar samples as diagnosed by VirusTotal. We thereby obtain 21 ransomware families for all the samples used in our research.

C. Strategy to Answer RQ1: Sequence Mining and One-Class Classification

We select the IRP Major Operation Type feature to investigate the unique pattern(s) in ransomware IRP logs that are common among all the samples and distinctly distinguishable from the benign IRP logs. The intuition behind this selection is that each flag-based value is mapped with a certain IRP request executed by the process (*e.g.*, read, write, close, etc.). Due to this fact, we can extract the sequence of IRP operations performed by all the ransomware processes in our study based on the majority number of times a certain pattern is observed. Therefore, we can capture a meaningful representation of a portion or a complete flow of actions and their frequency during execution. We record frequency every 5 minutes of the execution cycle.

To find the sequence(s), we randomly choose one ransomware sample's IRP logs dataset from each ransomware family. After partitioning each dataset in a 5-minute time frame, we drop all other features but IRP Major Operation Type. Then, we assign each occurred flag value into a unique letter, *i.e.*, we change *IRP_MJ_Write* (a flag value) to *W*. Finally, we convert all these characters (which appeared as records in the dataset) into a long string to compare them with all other samples. Our approach is an application of the longest string matching problem. After discovering the longest matched string, we break it down into 4 additional set of sequences as the frequencies of each derived set are also significantly high. We present all our extracted sequences in Fig. 2.

The description of the captured sequences is as follows:

- Sequence #1. This is the shortest pattern among all five availed sequences. It covers two IRP Major Operation Type flag values: "IRP MJ Acquire For Section Sync" labeled as *A* and "IRP MJ Release For Section Sync" labeled as *R*. Both flag values represent the File System Filter (FSF) callback operation. While *A* indicates that the file object is going to create a section that contains

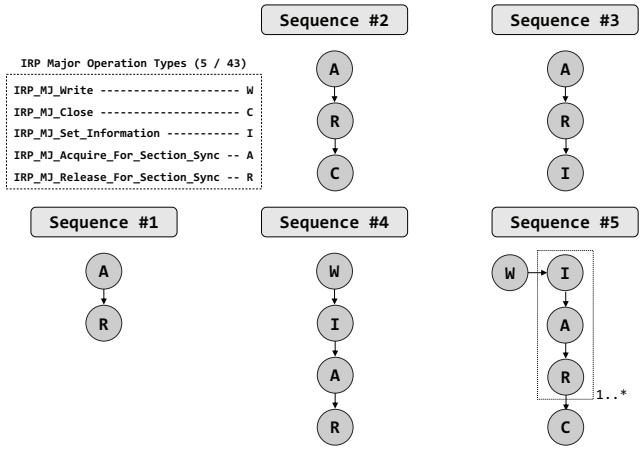

Fig. 2. Notable Sequences Observed from the IRP Major Operation Type Feature for all Ransomware Families' IRP Logs.

information about the structure size of the section along with a flag value, *R* indicates that the handle is associated with the file object for section has been closed or released. As shown in Table I, this pattern appears the highest over a 5-minute time frame among all 21 ransomware families' IRP logs.

- Sequence $\#2$. This sequence adds a flag value after Sequence $\#1$, that is "IRP MJ Close" labeled as *C*. This indicates that all outstanding I/O requests handled by a file object have been completed or canceled. In general, a driver should undo whatever actions it takes upon receipt of this request. We observe that the frequency of this ordered sequence, $A \rightarrow R \rightarrow C$, is the second largest over a 5-minute time frame with a median value of 554.5 (as shown in Table I).
- Sequence $\#3$. Similar to Sequence $\#2$, this sequence also adds another flag value after Sequence $\#1$. For this case, "IRP MJ Set Information" labeled as *I* is added after *A* and *R*. The operating system (OS) sends this IRP request, *I*, to set metadata about a file or file handle. Drivers are not required to handle this request, but when it changes any information about a file object, the OS sends this request. As per Table I, this sequence appears fewer times than the other two aforementioned sequences.
- Sequence $\#4$. This is another unique pattern, containing 4 IRP Major Operation Type flag values, that we observe among 21 ransomware families' IRP datasets. The sequence starts with "IRP MJ Write" labeled as *W*, triggered when a device driver transfers data from system memory to its device. The device driver sets the Information field of the I/O status block to the number of bytes transferred when it completes the IRP. As presented in Table I, this sequence's median and mean counts over a 5-minute time frame among all studied ransomware families are understandably lower than other discussed sequences because of its longer length.
- Sequence $\#5$. This sequence is the longest-matched pattern that covers all the flag values in other sequences. The pattern is similar to Sequence $\#4$ with a couple of exceptions: (1) the ordered flow $I \rightarrow A \rightarrow R$ has to appear one or multiple times after *W*, and (2) the pattern ends with *C*. We notice that this longest sequence obtains the median count of 3 over 5 minutes during all the studied ransomware families' 90 minutes execution (as pointed out in Table I).

TABLE I
SUMMARIZED STATISTICAL COUNTS OF NOTABLE SEQUENCES OBSERVED FROM THE IRP MAJOR OPERATION TYPE FEATURE FOR ALL RANSOMWARE SAMPLES' IRP LOGS OVER 5 MINUTES TIME FRAME. MEDIAN VALUES OF 90 MINUTES EXECUTION

Sequences	*Min*	*Q1*	*Median*	*Mean*	*Q3*	*Max*	*Max Outlier*
$\#1$	90.5	663.25	1,337.5	1,615.15	2,061.38	3,388.5	5,597.5
$\#2$	17.5	233.38	554.5	669.25	926.25	1,452	2,077
$\#3$	0	22.75	67.25	138.88	122.25	201.5	1,043
$\#4$	2	13.88	26.75	210.25	159.5	276.5	1,801.5
$\#5$	0	1.25	3	18.9	15.5	27	128.5

Motivation of Selecting the Sequences. As pointed out, we derive Sequence $\#1$ to Sequence $\#4$ from Sequence $\#5$ based on having significant appearances or counts among ransomware samples. To describe this longest sequence, it starts with a process, *e.g.*, belonging to a ransomware process, that creates a file object to perform write operations on the device residing on system memory (*W*). Upon changing the information about a file or file handle, *i.e.*, requesting to delete the file when it is closed or cancel a previously requested deletion, the Operating System sends the following request as a receipt that the file object stores the meta-data of the accessed file/file handle (*I*). Then, the following two callback operation requests (*A* and *R*) signify that the file object is acquiring information on the file handle's structure size and is followed by releasing such information. Such processes ($I \rightarrow A \rightarrow R$) after the write IRP request can repeat multiple times to support the process's goal of handing the files in the file system before it sends the close IRP request to indicate that the file object is done performing all the actions (*C*). In short, this sequence presents a process's unique file accessing flow regarding IRP operations. Other derived sequences allow us to formulate One-Class classification tasks effectively as features for the algorithms to train and test. For every studied ransomware sample's IRP log dataset, we compute all the sequences' counts over a 5-minute time frame. It allows us to generate different statistical measurements through box-plot analysis, such as minimum, first quartile (Q1) – the median of the lower half of the records, median, mean, third quartile (Q3) – the median of the upper half of the records, maximum, and maximum outlier. As we gather such results for every 5 minutes from the dataset, we populate Table I and Table II with the median values for 90 minutes of the execution cycle, that is median values of 18 (which is derived from $90/5$) records.

TABLE II
COMPARISON OF STATISTICAL COUNTS BETWEEN BENIGN AND RANSOMWARE PROCESSES OF NOTABLE SEQUENCES OBSERVED FROM IRP MAJOR OPERATION TYPE FEATURE FOR ALL RANSOMWARE SAMPLES' IRP LOGS OVER 5 MINUTES TIME FRAME (MEDIAN VALUES OF 90 MINUTES EXECUTION)

Sequences	*Min*		*Median*		*Mean*		*Max*	
	Benign	*Ransomware*	*Benign*	*Ransomware*	*Benign*	*Ransomware*	*Benign*	*Ransomware*
#1	0	90.5	143.75	1,337.5	157.06	1,615.15	480.5	5,597.5
#2	0	17.5	0	554.5	0.12	669.25	1	2,077
#3	0	0	0	67.25	0.17	138.88	2	1,043
#4	0	2	0	26.75	0.06	210.25	1	1,801.5
#5	0	0	0	3	0	18.9	0	128.5

We further our analysis to inspect the same analogy on sequences' counts for benign processes. We parse through IRP Major Operation Type feature's records of all the benign processes and select the highest counts for each dataset over a 5-minute time frame. After accumulating all the findings, we generate statistical measurements for benign processes through box-plot analysis similar to what we did for ransomware processes. We present our comparison chart in Table II. Here, we notice that the median counts of benign processes from Sequence #2 to Sequence #5 is 0. Additionally, we do not find Sequence #5 present in any benign processes in our studied IRP dataset. The differences between benign and ransomware processes for every extracted sequence are significant, strengthening our motivation to derive different patterns during ransomware encryption successfully. As we gather the unique counts of all the sequences for 21 ransomware families in terms of 5 minutes, we begin exploring the similarities of such sequences' presence among all the ransomware families. We incorporate Principal Component Analysis (PCA) to reduce the multivariate sequence counts' records in two dimensions, PCA-1 and PCA-2. We utilize PCA with different numbers of sequences as features to plot multivariate data in two dimensions, which we can later visualize to find a common cluster as a resemblance of discovering similarities among all the studied ransomware families.

These findings encourage us to further our analysis with One-Class classification to explore the percentage of points from the records of sequences' counts that do not fit in a common cluster that is to be derived by classification algorithms. In other words, we aim to identify a boundary from the generated observations with the help of One-Class classification algorithms to discover future ransomware samples. For this task, we choose four algorithms: One-class Support Vector Machine (SVM) with the non-linear kernel (RBF) [25], Isolation Forest [15], Local Outlier Factor (LOF) [4], and Robust Covariance. With these algorithms, we conduct *Novelty Detection* analysis, which is classifying new observations that may or may not differ in some respect from the observations we train the classifiers with [22]. In our case, we expect to detect as few new observations as possible that do not fit in the cluster derived from the training data provided.

D. Strategy to Answer RQ2: Multiclass Classification

To recognize all 21 ransomware families through its IRP logs, we carry out multiclass classification tasks with a list of the tree-based algorithms (such as Decision Tree, Random Forests, Extra Tree, and Bagging) and Neural Networks.

Construction of Artificial Neural Network (ANN or Neural Networks). We construct an Artificial Neural Network as per [2] to compare our findings with the algorithms mentioned above. Its structure is a fully connected network with one input layer, one hidden layer, and one output layer. The size of its input layer neuron is the number of features in the training set. In our case, the optimal setting for the number of hidden layers is less than twice the size of the input layer. The output layer contains 21 neurons to predict all the ransomware families. To describe its network further, we utilize the Rectified Linear Unit (ReLU) activation function for both the input and hidden layer, while we use the softmax activation function for the output layer [23]. We leverage an Adam Optimization Algorithm [14] and Sparse Categorical Cross Entropy loss function [16] for model compilation. We incorporate the early stopping method during training to ensure the network's generalization ability. We monitor validation loss for up to 3 iterations to trigger this action if the model shows no learning development in the training phase. We select 15% of the training records as the validation set.

We not only aim to detect ransomware families effectively, but also we evaluate all the aforementioned algorithms' performances concerning different time chunks of the datasets, *i.e.*, we report the classifiers' performances in terms of Accuracy, Precision Score, Recall Score, and F1 Score by starting from 15 until 40 minutes of IRP logs of ransomware processes only. To process the dataset, we remove a good number of features from the IRP logs as such do not have much impact on predicting the classes, such as Process Name, File Name, File Object, Inform, Transaction, Pre Operation Time, Post Operation Time, and Argument 1, Argument 2, ... , Argument 6. We perform One-Hot Encoding to transform such into the one-hot numeric array for the categorical flag-based features, *e.g.*, IRP Major Operation Type, IRP Flags, and Status. We select only the ransomware processes' IRP logs from all the samples and introduce a feature, named *Class*, to label which family it represents to perform multiclass classification prediction on it. To ensure standard machine learning development practices, we incorporate Stratified 5-Fold Cross-Validation while compiling each classifier discussed above [7]. During the split of train and test instances, these folds preserve the percentage of IRP logs' classes, representing 21 families.

IV. RESULTS

We answer RQ1 and RQ2 in this section.

A. Answer to RQ1: One-Class Classification

We begin discussing our empirical findings of One-Class Classification with four algorithms, One-Class SVM, Isolation Forest, Local Outlier Factor (LOF), and Robust Covariance, in terms of two performance parameters: Error Train – the percentage of training observations that do not fit in the cluster derived from the respective algorithm; and Error Novel Regular – the percentage of testing observations that do not

TABLE III
COMPARISON CHART OF DIFFERENT NOVELTY DETECTION ALGORITHMS' PERFORMANCE FOR RANSOMWARE SAMPLES' DERIVED SEQUENCE COUNTS FROM THE IRP MAJOR OPERATION FEATURE OF IRP LOGS

Algorithm	*Sequences*	*Performance Parameter*	*Performance in Time Chunks (in minutes)*						
			15	*20*	*25*	*30*	*35*	*40*	*Median*
One-Class SVM	#1, #2, and #3	Error Train	10.64%	11.11%	10.26%	8.51%	8.18%	9.52%	9.89%
		Error Novel Regular	*6.25%*	*19.05%*	11.11%	9.38%	*2.70%*	14.29%	10.25%
	#4 and #5	Error Train	10.64%	11.11%	10.26%	9.57%	8.18%	9.52%	9.92%
		Error Novel Regular	25.00%	19.05%	11.11%	15.62%	10.81%	4.76%	13.37%
	#1, #2, #3, and #4	Error Train	12.77%	11.11%	11.54%	10.64%	10.00%	9.52%	10.88%
		Error Novel Regular	*6.25%*	28.57%	*7.41%*	*6.25%*	8.11%	9.52%	*7.76%*
	All	Error Train	12.77%	*7.94%*	8.97%	11.70%	10.00%	9.52%	9.76%
		Error Novel Regular	12.50%	23.81%	*7.41%*	12.50%	8.11%	7.14%	10.31%
Isolation Forest	#1, #2, and #3	Error Train	10.64%	11.11%	10.26%	10.64%	10.00%	10.32%	10.48%
		Error Novel Regular	*6.25%*	28.57%	18.52%	9.38%	5.41%	7.14%	8.26%
	#4 and #5	Error Train	10.64%	11.11%	10.26%	10.64%	10.00%	10.32%	14.96%
		Error Novel Regular	25.00%	14.29%	18.52%	15.62%	13.51%	11.90%	10.48%
	#1, #2, #3, and #4	Error Train	10.64%	11.11%	10.26%	10.64%	10.00%	10.32%	10.48%
		Error Novel Regular	12.50%	23.81%	18.52%	12.50%	10.81%	11.90%	12.50%
	All	Error Train	10.64%	11.11%	10.26%	10.64%	10.00%	10.32%	10.48%
		Error Novel Regular	*6.25%*	28.57%	14.81%	12.50%	10.81%	*2.38%*	11.66%
Local Outlier Factor	#1, #2, and #3	Error Train	8.51%	9.52%	10.26%	9.57%	9.09%	9.52%	9.52%
		Error Novel Regular	*6.25%*	28.57%	18.52%	12.50%	10.81%	4.76%	11.66%
	#4 and #5	Error Train	*4.26%*	*7.94%*	*5.13%*	*4.26%*	*5.45%*	*6.35%*	*5.29%*
		Error Novel Regular	18.75%	*19.05%*	14.81%	9.38%	13.51%	11.90%	14.16%
	#1, #2, #3, and #4	Error Train	8.51%	9.52%	8.97%	*4.26%*	*5.45%*	9.52%	8.74%
		Error Novel Regular	18.75%	23.81%	14.81%	9.38%	5.41%	*2.38%*	12.10%
	All	Error Train	8.51%	*7.94%*	10.26%	7.45%	8.18%	*7.14%*	8.06%
		Error Novel Regular	*6.25%*	23.81%	18.52%	9.38%	10.81%	4.76%	10.10%
Robust Covariance	#1, #2, and #3	Error Train	25.53%	25.40%	25.64%	25.53%	25.45%	25.40%	25.49%
		Error Novel Regular	43.75%	42.86%	33.33%	25.00%	21.62%	21.43%	29.17%
	#4 and #5	Error Train	25.53%	25.40%	25.64%	25.53%	25.45%	25.40%	25.49%
		Error Novel Regular	31.25%	28.57%	25.93%	21.88%	27.03%	23.81%	26.48%
	#1, #2, #3, and #4	Error Train	25.53%	25.40%	25.64%	25.53%	25.45%	25.40%	25.49%
		Error Novel Regular	43.75%	42.86%	37.04%	25.00%	27.03%	19.05%	32.04%
	All	Error Train	25.53%	25.40%	25.64%	25.53%	25.54%	25.40%	25.53%
		Error Novel Regular	43.75%	33.33%	25.93%	18.75%	21.62%	21.43%	23.78%

fit in that cluster. We aim to achieve as minimum score as possible in both cases. During our experimentation, we test each classification algorithm with four different feature settings, such as Sequence #1, #2, and #3; Sequence #4 and #5; Sequence #1, #2, #3, and #4; and All Sequences, along with the integration of Principal Component Analysis (PCA). Additionally, we explore the experimental results for every setting with sequence counts extracted from 15 to 40 minutes of ransomware IRP logs. As we have 21 ransomware families' sequence counts, we randomly separate 16 families' observations (approx. 75%) to train the classifiers and test them with the remaining five families. In this way, we investigate how well the algorithms can perform with the ransomware samples it did not see during the training process, which reassembles a real-world example. After learning from the training observations, all the classifiers derive a boundary or region. Then, we compute the percentage of training (Error Train) and testing (Error Novel Regular) instances that fall outside this space.

We present a comparison chart comprising all the classifiers' performances concerning every mentioned experimental setting in Table III. Among the experimented algorithms, we achieved better results with One-Class SVM and Local Outlier Factor (LOF). To highlight the best part of empirical findings, we obtain 4.26% and 6.25% of Error Train and Error Novel Regular scores, respectively, for 15 minutes of extracted sequences from ransomware families. Similarly, we attain 7.94%, 5.23%, 4.26%, 5.45%, 6.35% as Error Train and 19.05%, 7.41%, 6.25%, 2.70%, 2.38% as Error Novel Regular for 20, 25, 30, 35, and 40 minutes of observations respectively. Additionally, we introduce the Median column in the table to summarize which sequence patterns as features and algorithms perform better. From this column, we can state that the Local Outlier Factor (LOF) algorithm with Sequence #4 and #5 possesses the least amount of Error Train, which is 5.29% while the One-Class SVM algorithm with Sequence #1, #2, #3, and #4 acquires 7.76% as Error Novel Regular, which is also the least number for this set. These achieved results firmly support our effort to extract a set of distinguishable ransomware encryption patterns that have been further tested with five unseen ransomware families' sequence counts with the help of One-Class Classification-enabled machine learning algorithms. Notably, the Robust Covariance algorithm lags in performance for every experimental setting; however, we include its results in our study for the researchers and practitioners in this field.

B. Answer to RQ2: Multiclass Classification

This section describes the empirical findings of each machine learning classifier studied in our study. We report the

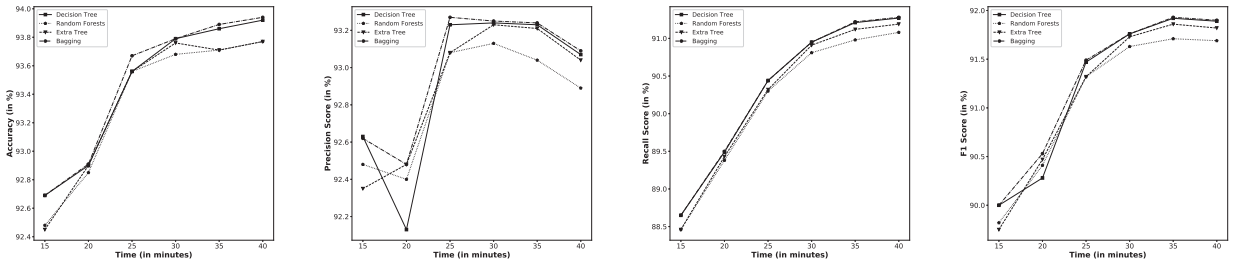

Fig. 3. Performance of Decision Tree, Random Forests, Extra Tree, and Bagging Classifiers in Terms of Accuracy (Top Left), Precision Score (Top Right), Recall Score (Bottom Left), and F1 Score (Bottom Right).

TABLE IV
COMPARISON CHART OF DIFFERENT MACHINE LEARNING ALGORITHMS' PERFORMANCE FOR MULTICLASS CLASSIFICATION USING IRP LOGS OF 21 RANSOMWARE FAMILIES

Performance Parameter		*Decision Tree*	*Random Forests*	*Extra Tree*	*Bagging*	*ANN*
Accuracy (in minutes)	15	92.69%	92.48%	92.45%	92.69%	79.78%
	20	92.90%	92.85%	92.90%	92.91%	80.29%
	25	93.65%	93.56%	93.56%	93.67%	80.01%
	30	93.79%	93.68%	93.76%	93.79%	80.55%
	35	93.86%	93.71%	93.80%	93.89%	80.53%
	40	93.92%	93.77%	93.86%	93.94%	80.55%
Precision (in minutes)	15	92.63%	92.48%	92.35%	92.62%	92.03%
	20	92.13%	92.40%	92.48%	92.48%	89.10%
	25	93.23%	93.08%	93.08%	93.27%	89.80%
	30	93.24%	93.13%	93.23%	93.25%	91.60%
	35	93.23%	93.04%	93.21%	93.24%	72.82%
	40	93.07%	92.89%	93.04%	93.09%	89.74%
Recall (in minutes)	15	88.65%	88.46%	88.46%	88.65%	69.88%
	20	89.49%	89.38%	89.43%	89.50%	71.43%
	25	90.44%	90.30%	90.32%	90.44%	71.52%
	30	90.95%	90.81%	90.91%	90.95%	72.47%
	35	91.21%	90.98%	91.12%	91.22%	90.41%
	40	91.27%	91.08%	91.19%	91.28%	73.01%
F_1 (in minutes)	15	90.00%	89.82%	89.75%	90.00%	76.65%
	20	90.28%	90.41%	90.47%	90.53%	76.68%
	25	91.47%	91.32%	91.32%	91.49%	77.45%
	30	91.76%	91.63%	91.73%	91.76%	79.25%
	35	91.92%	91.71%	91.86%	91.93%	79.31%
	40	91.89%	91.69%	91.82%	91.90%	77.99%

classifiers' performances for 15, 20, 25, 30, 35, and 40 minutes of IRP logs to evaluate how effectively such classification algorithms can perform prediction with different amounts of records, as it is integral in our study to explore early detection capabilities. We present Table IV to illustrate the performances of Decision Tree, Random Forests, Extra Tree, Bagging, and Artificial Neural Network (ANN) classifiers in terms of Accuracy, Precision score, Recall score, and F1 score. We achieve neck-and-neck results from every case's tree or decision rule-based algorithms. It is noted that we take the mean values of all the results generated by stratified 5-fold cross-validation. To further highlight, these four classifiers give us Accuracy of 92.45% – 93.94%, Precision score of 92.35% – 93.27%, Recall score of 88.46% – 91.28%, and F1 score of 89.75% – 91.90% between 15 and 40 minutes of ransomware IRP logs. ANN, in comparison, obtains the highest scores of 80.55% as Accuracy, 92.03% as Precision, 90.41% as Recall, and 79.31% as F1 score. All the achieved results strongly indicate that we have successfully detected 21 ransomware families effectively; especially, the classifiers' performances with 15 minutes of IRP logs are exceptional. It suggests that early detection of the ransomware family from its IRP logs is very much possible with this list of algorithms.

Additionally, we present the reported scores for Decision Tree, Random Forests, Extra Tree, and Bagging classifiers broken into Accuracy, Precision, Recall, and F1 scores in Fig. 3. These visualizations indicate how close the results are for the mentioned algorithms in every case and the growth in performance parameters as we add more logs for the classifiers to be trained. In other words, the more IRP logs we feed the classifiers, the better performance we achieve. We also note that the growth from 15 to 40 minutes is within 3% every time, which is not significant and proof that the classifiers have successfully learned the underlying pattern of IRP logs for all 21 ransomware families in a reasonably early stage that serves our purpose to address RQ2.

V. CASE STUDY: COMPARISON WITH COLLECTED IRP LOGS FROM USERS' MACHINES (BENIGN)

Along with ransomware IRP logs of 383 samples, the collected dataset from [6] comes in with benign IRP logs. The authors managed 11 volunteers, including developers, home, and office users. The volunteers installed the built sniffer tool on their machines, from Microsoft Windows 7 to Windows 10, to record their daily activities. These recorded logs were considered the ground truth dataset based on the assumption that cyber-attacks compromised none of these machines. Each machine's captured log is stored in sessions. We base our comparison analysis on such user sessions from all the machines in two ways: (1) we compare the statistical counts of all the highlighted file system features, such as the total number of individual files accessed as well as unique file objects created and the highest mean value of buffer length as well as entropy per process; and (2) we provide similar comparison findings on the counts of the sequences observed from the IRP Major Operation Type feature as discussed above.

We begin by describing our findings of the voluntary users' behavior on the file system features, *e.g.*, Unique File Ac-

cessed, Unique File Objects, Buffer Length, and Entropy. We choose the session with the highest counts on such elements from the multiple sessions on each machine for our analysis. After generating counts for all four parts from 11 machines, we compute the minimum, median, mean, and maximum statistical measurements. To compare the generated values with the ransomware samples, we select the counts over a 5-minute time frame of 90 minutes of ransomware execution, and we portray the average impact of 383 ransomware samples in a five-minute window. The rationale behind this approach is that the differences between the respective counts are significant because the number of files a ransomware process will access in the file system during its execution (or in a session) will be much higher than that of a regular user using their machine. We present our findings in a tabular format in Table V. We notice that the total number of unique files accessed and unique file objects created by the voluntary users in a typical session is much less than ransomware for all the mentioned statistical measurements. On the other hand, we observe that the mean values of the other two features, the highest buffer length, and entropy from a process of users' machines are significantly higher than ransomware processes. From these results, we remark that in a 5-minute time frame, the ransomware process(es) during its execution are likely to access a notably large number of files; however, such processes are less likely to perform many modifications on the accessed files as much as an average user tends to modify.

TABLE V
COMPARISON OF STATISTICAL COUNTS OF FILE SYSTEM FEATURES BETWEEN A COMPLETE SESSION OF 11 USERS' MACHINES AND OVER 5 MINUTES AVERAGE TIME FRAME OF RANSOMWARE EXECUTION

File System Features	*Min*		*Median*		*Mean*		*Max*	
	Users	*Ransomware*	*Users*	*Ransomware*	*Users*	*Ransomware*	*Users*	*Ransomware*
File Accessed (Unique)	30	1,667	519	3,065	1,583.64	2,851	9,277	3,715
File Objects (Unique)	47	1,135	378	1,899	470.18	2,059	1,521	3,231
Buffer Length (Mean)	8,192	5,870	32,768	21,125	42,891.8	20,734	141,626.25	37,435
Entropy (Mean)	0.066	0.077	0.549	0.125	0.502	0.12	0.79	0.16

Now, we discuss our comparison analysis on the counts of the notable sequences observed from the IRP Major Operation Type feature (as shown in Fig. 2). We generate the counts of all five sequences from the same users' sessions and focus on developing the counts on file system features. We select one process from each session with the highest sequence counts. Then, we compute similar statistical measurements from the counts we derive. As the users perform much more modifications on the file systems during the considered sessions, the file I/O operations based on IRP logs are expected to become high. Hence, the counts of all the sequences will be significant. For this reason, we compare such generated benign counts with all the studied ransomware samples' counts deriving from its entire 90-minute execution cycle. We present our findings in Table VI to compare the counts of all five sequences of voluntary users and ransomware. The table shows that the ransomware process(es) shows much fewer counts in Sequence 1 and Sequence 5. Thus, it appears to be difficult to identify ransomware encryption being taken in place from the frequency of such mentioned sequences. However, we can differentiate ransomware behavior from benign user actions based on the counts of Sequences 2, 3, and 4, as its counts are dominant compared to the captured IRP logs from the voluntary user machines.

TABLE VI
COMPARISON OF STATISTICAL COUNTS OF NOTABLE SEQUENCES OBSERVED FROM IRP MAJOR OPERATION TYPE FEATURE BETWEEN A COMPLETE SESSION OF 11 USERS' MACHINES AND RANSOMWARE

Sequences	*Min*		*Median*		*Mean*		*Max*	
	Users	*Ransomware*	*Users*	*Ransomware*	*Users*	*Ransomware*	*Users*	*Ransomware*
#1	604	13,668	17,658	27,353	51,984.27	29,813.5	319,548	63,642
#2	110	1,668	1,748	12,663	2,922.09	11,698.33	10,773	22,586
#3	0	165	63	1,574.5	328	3,235.78	1,863	34,984
#4	0	475	276	1,483	2,788	4,154.11	15,600	17,848
#5	0	23	110	96.5	636.18	322.72	3,154	2,015

In this case study, we demonstrate the effectiveness of our data-driven empirical study by comparing the behavior between benign users' actions and ransomware encryption processes in terms of statistical measurements. By generating the counts of file system features and notable five sequences observed from IRP Major Operation Type, we analyze the captured IRP logs of 11 user machines to evaluate how well we can distinguish ransomware behavior from the knowledge we gathered after inspecting 383 ransomware samples. We identify that the ransomware process(es) accesses many more files with far fewer modifications in the file system over a 5-minute time frame than a regular user's interactions with the file system. In addition, we discover that the counts of Sequence 2, 3, and 4 indicate the significant behavioral differences between benign and malicious file I/O operations. Therefore, the case study validates that our empirical derivations from ransomware behavior analysis are well generalized. We leave combining multiple feature counts, *e.g.*, file system features and sequence counts, to propose an effective threshold-based approach for ransomware identification through IRP logs as future work.

VI. RELATED WORK

I/O Request Packet (IRP) log has been the center of some state-of-the-art ransomware detection solutions since 2016. In 2015, Kharraz *et al.*was the first to analyze 1,359 ransomware samples to describe the workings and effects of ransomware and the usefulness of monitoring the file system through IRP logs in users' machines for successful ransomware detection [11]. Their work was later got adapted by many security researchers in this domain as a viable ransomware detection technique. In 2016, Continella *et al.* [6] used IRP logs as a focal point in their research to propose a real-time self-healing virtual file system approach resilient to malicious encryption to prevent the effects of ransomware attacks. The authors collected a huge amount of IRP logs during benign users' activities on ransomware-free machines, as they claimed, to understand how ransomware typically interacts with the file system in comparison to benign applications. On the basis of this understanding, they created detection models that distinguish ransomware from benign processes at runtime. Kharaz *et al.* [10] utilized a minifilter driver to collect IRP logs to monitor system-wide file system change and access many objects of the Windows-based Operating System. The authors

experimented with their proposed approach with 13,637 ransomware samples that cover both crypto and locker types of ransomware. In 2018, Mehnaz *et al.* [19] leveraged IRP logs to propose a ransomware surveillance system with the utilization of process monitoring (upon receipt of IRP open, close, read, write, and create operation) and file change monitoring (upon receipt of IRP write operations). The authors successfully tested their devised detection scheme with 14 ransomware families. Like Huang *et al.* [9] and Continella *et al.* [6], the authors also claimed data recovery of the encrypted files performed by ransomware. Additionally, their built detection tool took account of plotted decoy files on experimented machines and monitored such files' change operations. We notice that the technique of planting decoy or honey files for ransomware detection has been well adopted by security researchers in this field [3], [8], [20], [31].

To present the interaction of ransomware with file systems through IRP logs, McIntosh *et al.* [17], [18] shared some useful pieces of information: (1) the types of the files (as well as paths) ransomware generally targeted; (2) the number of the IRP requests to modify file contents; and (3) the total number of file types modified during ransomware operation. Overall, security researchers and practitioners have already considered IRP logs to report the findings of their studies as they aim to prevent or mitigate the damage from ransomware attacks [1], [5], [12]. However, apart from ransomware, it is also worth mentioning that IRP logs are used to propose detection techniques for other types of malware and to learn its behavior [28], [32], [33]. Having discovered the usefulness of IRP logs for ransomware detection, we learn granular level and actionable insights from ransomware behavior.

We also observe that time series analysis has been widely studied in other classes of malware [13], [21]. It allows malware researchers and analysts to distinguish malicious and benign processes' behavior and perform binary classification tasks on the extracted pattern. In this study, our prime intuition behind employing time series analysis on the IRP logs of ransomware and benign processes is to learn the distinguishable patterns with time and evaluate our proposed approaches concerning time to report how early we can successfully detect devastating ransomware attacks.

VII. Discussion and Limitations

We conduct an I/O Request Packet (IRP) driven analysis to propose early ransomware detection. We use the IRP logs collected during the dynamic analysis of 383 ransomware samples performed in [6]. The logs were collected on a Windows 7 machine in 2016. We acknowledge that Microsoft stopped supporting Windows 7 devices on January 14, 2020. Although most computer devices were operational with Windows 7 when the data collection was performed, inspecting the logs captured while performing similar experiments on a Windows 10 or later OS-installed device will be interesting.

We utilize Sequence Mining tasks to identify a common encryption pattern among the studied ransomware families. We derive five common patterns or sequences from the analyzed ransomware samples' IRP logs. With the counts of each pattern over 5 minutes of observations, we observe significant differences between benign and ransomware processes. Additionally, with similar intuition, we have successfully discovered new instances of ransomware samples with the help of One-Class SVM, Isolation Forest, Local Outlier Factor (LOF), and Robust Covariance algorithms. We, however, do not explore other sequencing applications, such as graph-based approach [24], genome sequence [29], etc., to test the similarities of IRP logs between two ransomware samples.

We point out several statistical measurements, *e.g.*, minimum, Q1, median, mean, Q3, and maximum, of ransomware processes and the differences between ransomware and benign processes to use such derived empirical knowledge to combat a ransomware attack while it is in progress. For example, the median values of different features from the IRP log with the basis of the time series trend on all the studied ransomware samples will allow security researchers and network defenders to consider a threshold to improve their defensive malware monitoring tool(s). Additionally, our approaches to address RQ1 and RQ2 with the description of ransomware behavior will inspire security practitioners to integrate such empirical methods to analyze suspicious process(es) on a Windows-based machine. The ransomware families were caught on or before 2016; therefore, incorporating more recently discovered ransomware families' samples, such as WannaCry, NotPetya, SamSam, Ryuk, REvil, etc., is much desired.

VIII. Conclusion

In this research, we aim to find actionable and granular insights from this low-level file system I/O log, beginning with understanding the behavior of 383 ransomware samples (collected from [6]). We highlight the behavioral results of time series trends and box-plot analysis of all the studied ransomware samples and the difference between ransomware and benign processes. We share the statistical measurements, such as minimum, median, mean, maximum, etc., of ransomware and benign processes to interpret all the notable features of IRP over 90 minutes of execution, partitioning into a 5-minute time frame. We explore finding distinguishable pattern(s) during ransomware encryption among all the studied ransomware families, and thereby, we derive five sequences that can distinctly characterize ransomware processes by separating them from benign processes. Then, we analyze the sequences' counts to examine if we can develop a common cluster that will enable us to discover new ransomware variants. Therefore, we incorporate One-Class Classification algorithms, *e.g.*, One-Class SVM, Isolation Forests, Local Outlier Factor (LOF), and Robust Covariance, to help us create the automated boundary space of such counts by training it with observations from 16 ransomware families. We investigate how well such classifiers' derived clusters can fit the observations from the remaining five ransomware families.

We inspect classifying all 21 ransomware families to obtain a generalized understanding of how families leave their footprint through IRP logs. We employ a long list of machine

learning algorithms to perform experiments on this multiclass classification task. We observe that the empirical findings of Decision Tree, Random Forests, Extra Tree, and Bagging classifiers outperform all other experimented algorithms. For every experimental setting we design, we report an Accuracy of 92.45% – 93.94%, Precision score of 92.35% – 93.27%, Recall score of 88.46% – 91.28%, and F1 score of 89.75% – 91.90% between 15 and 40 minutes of ransomware IRP logs. Our approach can discover new ransomware families and samples in the future when caught out in the wild.

ACKNOWLEDGEMENT

We thank the anonymous reviewers for reviewing this scientific manuscript and providing valuable feedback. We are grateful to Dr. Andrea Continella for sharing their collected I/O Request Packet (IRP)-based dataset, published in [6]. We extend our sincere gratitude to the Cybersecurity Education, Research & Outreach Center (CEROC) at Tennessee Tech University for supporting this research since its inception.

REFERENCES

[1] Or Ami, Yuval Elovici, and Danny Hendler. Ransomware prevention using application authentication-based file access control. In *Proceedings of the 33rd Annual ACM Symposium on Applied Computing*, pages 1610–1619, 2018.

[2] Md Ahsan Ayub, Andrea Continella, and Ambareen Siraj. An i/o request packet (irp) driven effective ransomware detection scheme using artificial neural network. In *2020 IEEE 21st International Conference on Information Reuse and Integration for Data Science (IRI)*, pages 319–324. IEEE, 2020.

[3] Brian M Bowen, Shlomo Hershkop, Angelos D Keromytis, and Salvatore J Stolfo. Baiting inside attackers using decoy documents. In *International Conference on Security and Privacy in Communication Systems*, pages 51–70. Springer, 2009.

[4] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 93–104, 2000.

[5] Aviad Cohen and Nir Nissim. Trusted detection of ransomware in a private cloud using machine learning methods leveraging meta-features from volatile memory. *Expert Systems with Applications*, 102:158–178, 2018.

[6] Andrea Continella, Alessandro Guagnelli, Giovanni Zingaro, Giulio De Pasquale, Alessandro Barenghi, Stefano Zanero, and Federico Maggi. Shieldfs: a self-healing, ransomware-aware filesystem. In *Proceedings of the 32nd Annual Conference on Computer Security Applications*, pages 336–347, 2016.

[7] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics New York, 2001.

[8] José Antonio Gómez-Hernández, L Álvarez-González, and Pedro García-Teodoro. R-locker: Thwarting ransomware action through a honeyfile-based approach. *Computers & Security*, 73:389–398, 2018.

[9] Jian Huang, Jun Xu, Xinyu Xing, Peng Liu, and Moinuddin K Qureshi. Flashguard: Leveraging intrinsic flash properties to defend against encryption ransomware. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 2231–2244, 2017.

[10] Amin Kharaz, Sajjad Arshad, Collin Mulliner, William Robertson, and Engin Kirda. {UNVEIL}: A large-scale, automated approach to detecting ransomware. In *25th {USENIX} Security Symposium ({USENIX} Security 16)*, pages 757–772, 2016.

[11] Amin Kharraz, William Robertson, Davide Balzarotti, Leyla Bilge, and Engin 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*, pages 3–24. Springer, 2015.

[12] Dae-Youb Kim, Geun-Yeong Choi, and Ji-Hoon Lee. White list-based ransomware real-time detection and prevention for user device protection. In *2018 IEEE International Conference on Consumer Electronics (ICCE)*, pages 1–5. IEEE, 2018.

[13] Ki-Hyeon Kim and Mi-Jung Choi. Android malware detection using multivariate time-series technique. In *2015 17th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, pages 198–202. IEEE, 2015.

[14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[15] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*, pages 413–422. IEEE, 2008.

[16] Giovanni Mariani, Florian Scheidegger, Roxana Istrate, Costas Bekas, and Cristiano Malossi. Bagan: Data augmentation with balancing gan. *arXiv preprint arXiv:1803.09655*, 2018.

[17] Timothy McIntosh, Julian Jang-Jaccard, Paul Watters, and Teo Susnjak. The inadequacy of entropy-based ransomware detection. In *International Conference on Neural Information Processing*, pages 181–189. Springer, 2019.

[18] Timothy R McIntosh, Julian Jang-Jaccard, and Paul A Watters. Large scale behavioral analysis of ransomware attacks. In *International Conference on Neural Information Processing*, pages 217–229. Springer, 2018.

[19] Shagufta Mehnaz, Anand Mudgerikar, and Elisa Bertino. Rwguard: A real-time detection system against cryptographic ransomware. In *International Symposium on Research in Attacks, Intrusions, and Defenses* pages 114–136. Springer, 2018.

[20] Chris Moore. Detecting ransomware with honeypot techniques. In *2016 Cybersecurity and Cyberforensics Conference (CCC)*, pages 77–81. IEEE, 2016.

[21] Om Patri, Michael Wojnowicz, and Matt Wolff. Discovering malware with time series shapelets. In *Proceedings of the 50th Hawaii International Conference on System Sciences*, 2017.

[22] Marco AF Pimentel, David A Clifton, Lei Clifton, and Lionel Tarassenko. A review of novelty detection. *Signal Processing*, 99:215–249, 2014.

[23] Prajit Ramachandran, Barret Zoph, and Quoc Le. Searching for activation functions. 2018. URL: https://arxiv.org/pdf/1710.05941.pdf.

[24] Josh Saxe, David Mentis, and Chris Greamo. Visualization of shared system call sequence relationships in large malware corpora. In *Proceedings of the ninth international symposium on visualization for cyber security*, pages 33–40, 2012.

[25] Bernhard Schölkopf, Robert C Williamson, Alex J Smola, John Shawe-Taylor, and John C Platt. Support vector method for novelty detection. In *Advances in neural information processing systems*, pages 582–588, 2000.

[26] Scikit-Learn. Encoding categorical features, 2020. URL: https://scikit-learn.org/stable/modules/preprocessing.html#preprocessing-categorical-features.

[27] Marcos Sebastián, Richard Rivera, Platon Kotzias, and Juan Caballero. Avclass: A tool for massive malware labeling. In *International Symposium on Research in Attacks, Intrusions, and Defenses*, pages 230–253. Springer, 2016.

[28] Syed Zainudeen Mohd Shaid and Mohd Aizaini Maarof. Malware behavior image for malware variant identification. In *2014 International Symposium on Biometrics and Security Technologies (ISBAST)*, pages 238–243. IEEE, 2014.

[29] Jorge Maestre Vidal, Marco Antonio Sotelo Monge, and Luis Javier García Villalba. A novel pattern recognition system for detecting android malware by analyzing suspicious boot sequences. *Knowledge-Based Systems*, 150:198–217, 2018.

[30] VirusTotal. Public api v2.0, 2019. URL: https://www.virustotal.com/en/documentation/public-api/.

[31] Jim Yuill, Mike Zappe, Dorothy Denning, and Fred Feer. Honeyfiles: deceptive files for intrusion detection. In *Proceedings from the Fifth Annual IEEE SMC Information Assurance Workshop, 2004.*, pages 116–122. IEEE, 2004.

[32] Fuyong Zhang and Ying Ma. Using irp with a novel artificial immune algorithm for windows malicious executables detection. In *2016 International conference on progress in informatics and computing (PIC)* pages 610–616. IEEE, 2016.

[33] FuYong Zhang, DeYu Qi, and JingLin Hu. Using irp for malware detection. In *International Workshop on Recent Advances in Intrusion Detection*, pages 514–515. Springer, 2010.