Machine Learning for Android Ransomware Detection

Sikha Bagui
Department of Computer Science
University of West Florida
Pensacola, FL
bagui@uwf.edu

Tyler Woods
Department of Computer Science
University of West Florida
Pensacola, FL
tylerw21396@gmail.com

***Abstract*—Effective detection of ransomware is becoming increasingly important due to the influx of smartphones and the increasing amount of personal data being stored in smartphones. Ransomware, which often demands payment for the safe return of data, encrypts a user's personal data and renders it useless without proper decryption. In this paper, we present a model for an Android ransomware intrusion detection system that is an improvement over the previous works on the detection of Android malware families. This was accomplished through sufficient data preprocessing using information gain and the effective use of machine learning classifiers, Decision Tree, Naïve Bayes, and OneR. Network traffic data was used for this classification. Of the three classifiers, the decision tree classifier produced the best classification results.**

***Index Terms*—Android, Information Gain, J48 Decision Tree, Machine Learning, Malware, Naïve Bayes, OneR, Ransomware.**

I. INTRODUCTION

Smartphones continue to become more and more prevalent in our everyday lives, to the point where it is almost unheard of to know someone who does not carry a smartphone. There are several operating systems that smartphones use, such as Android, iOS, Windows, and BlackBerry. However, Android continues to dominate the smartphone operating system space, more so outside the United States. In the United States, Android makes up ~51% of the market share and iOS makes up ~48% of the market share [1]. Worldwide, Android makes up ~76% and iOS makes up ~22% [2]. Due to smartphones' ease of use and popularity, it is not surprising that a majority of network traffic is generated by smartphones, of which over 75% is Android network traffic. And, more personal information than ever before is being stored and transmitted via smartphones, hence it is to be expected that there is an increase in malware specifically targeting Android. It is becoming increasingly necessary to be able to secure the functionality and user's data on smartphones.

Malware is intentionally designed to cause harm. There are various types of malware, such as ransomware, adware, scareware, etc. In this work, the focus is on the Ransomware category of malware. Ransomware is one of the more popular attacks, where a user's data is held hostage and a payment is demanded for its safe return. This has a costly impact on businesses, which are being targeted in ever more sophisticated ways [3]. Payment is often demanded from businesses in the form of cryptocurrencies, to maintain anonymity. Hence, the threat of ransomware is apparent and real, especially for businesses. In some events, even after payment, there may be no guarantee that the data will be released back to the user. So, without proper detection, it may be a little too late.

Ransomware takes the most important part of the smartphone hostage, the user's data. It encrypts the data, making it completely unusable without the key to decrypt it. This is particularly threatening because the data may be irreplaceable. Though this makes the process of backing up data ever more important, this is often not done until the user falls victim, when it is too late.

The novelty of this paper lies is in improving the detection of ransomware on Android using network traffic data. The CICAndMal2017 dataset [4] is used for this research. Information Gain is applied for feature selection and supervised machine learning algorithms, J48 (Decision Tree), Naïve Bayes and OneR, are used for classification. The analysis is performed by each ransomware family taken individually. No previous work has analyzed the data by each ransomware family individually. The ten most important attributes for each ransomware family are presented, and the ten most important attributes for all the ransomware families taken together are also presented. Binary classification is performed for various numbers of attributes based on information gain.

II. RELATED WORKS

Very few papers have focused on using network traffic data to detect ransomware on smartphones. [5] performed a comprehensive study on smartphone malware.

Since there were no valuable current datasets that could be used to detect android malware from network traffic, especially in terms of presenting the current state of network traffic, [6] generated a new, more than 80 network traffic feature completely labelled IDS dataset, CICIDS2017, using CICFlowMeter software. They used RandomForestRegressor for feature selection and presented the best four features for

each attack. This work compared various common machine learning classifiers, but this work does not specifically look at attacks on Android smartphones.

[7] focused on the Android platform and collected more than 1,200 malware samples that covered the majority of the Android malware families until 2011. They systematized or characterized existing Android malware and performed classification, but this study is a little dated.

[8] introduced NetConverse, a machine learning analysis of Windows ransomware network traffic to achieve a high, consistent detection rate. Their dataset was created from conversation-based network traffic features and achieved a true positive detection rate of 97.1% using the Decision Tree classifier. [9] developed an API based ransomware detection system to provide a static analysis paradigm for detecting Android ransomware apps.

Other papers focused on other aspects of malware detection. [10] focused on the threat of malware cloned applications and the damage caused to affected Android users. [11] presented techniques used for detection of malware using cloud-based intrusion detection. [12] presented different malware detection approaches: static analysis, dynamic analysis, detection in the cloud, anomaly detection and signature-based detection. [13] provided a comprehensive overview of security solutions for smartphones. [14] focused on how malicious code or advertisements in legitimate smartphone apps are repackaged by attackers, and detection strategies for this. [15] used the Android system service call sequence, determined by using a co-occurrence matrix, as a feature, and used machine-learning algorithms to classify the feature sequence as malware or not. [16] used decision tree classification to detect malware.

Except for [6], there is no other work that focused on using network traffic for ransomware detection on the Android. Moreover, [6] did not present an analysis by the individual ransomware families, which has been done in this work.

III. The Dataset

CICAndMal2017 [17] is an Android dataset containing 84 network traffic features (detailed in Appendix A) from the CICFlowMeter [18] software. These network traffic features were obtained from PCAP files captured from experiments executing Android malware on test environments. Test environments were real smartphones, not emulators. The dataset consists of more than 10,854 samples, of which 4,354 are malware and 6,500 are benign. The benign samples, or apps, were collected from the Google Play market and were published in 2015, 2016, and 2017. 5,491 of the 10,854 samples were installed on real devices, of which 5,065 samples were benign and 426 were malware. The malware samples are composed of four categories: Adware, Ransomware, Scareware, and SMS Malware.

For this work, only the Ransomware category was analyzed, hence only this data was extracted. Due to the deceptive nature of malware attacks, three stages of data capturing were utilized to ensure the capturing of good samples of network traffic. The first stage captured network traffic immediately after installing the malware (1-3 minutes). The second stage captured network traffic 15 minutes before restarting the device. And, the third stage captured network traffic 15 minutes after restarting the device. Fig. 1 presents the Ransomware families and the percentages of attacks in each of the Ransomware families.

Each of these families (shown in Fig. 1) behave differently in terms of the network traffic generated. However, though they are all forms of ransomware, their level of complexity varies based on the skillset of the developer. Hence the dataset consists solely of network traffic generated from these apps and the CICFlowMeter software is responsible for the extraction of their features. These features are used to train a model to be able to classify network traffic as either benign or from a specific ransomware family.

The 84 attributes that the CICFlowMeter software extracts also consists of statistical features such as flow duration, number of packets, and network protocols. PCAP files with captured benign and malware network traffic were analyzed for bidirectional flow by the software. Flow is a sequence of packets from a source to a destination. The software analyzed the forward and backward flow of both benign and malware traffic. And, the data is labeled based on whether it is benign or from one of the ransomware families.

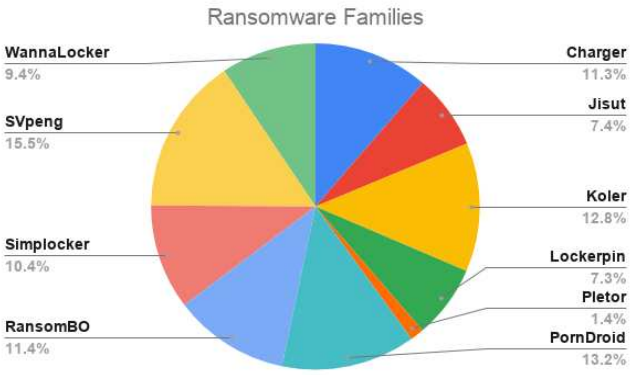

Fig. 1. Ransomware families in CICAndMal2017 [9] dataset

IV. Data Preprocessing

Each ransomware family initially contained 84 attributes. The initial preprocess step was to remove attributes that were too specific to the environment, hence the four attributes, Timestamp, Flow ID, Destination IP, and Source IP, were removed. Continuous attributes were discretized using Weka [19]. Next, a feature selection measure, Information Gain, was applied with the rest of the attributes.

A. Information Gain

Information Gain (IG), based on a measure of entropy, measures the relevance of a feature relative to a given class. Entropy measures the impurity in the data. Suppose there are m classes. Let S be a set of training samples where the class label is either benign or attack. Let S contain s_i samples of class C_i, where $i = 1...m$. The probability of an arbitrary sample belonging to class C_i is s_i/s, where s is the total number of samples in set S. For classifying a given sample, the expected

information is [20]:

$$I(s_1, s_2, \ldots, s_m) = -\sum_{i=1}^{m} \frac{s_i}{s} \log_2 \frac{s_i}{s}$$

A feature A with values $\{a_1, a_2, \ldots, a_v\}$ can be used to partition S into the subsets $\{S_1, S_2, \ldots S_v\}$, where S_j contains those samples in S that have value a_j of A. Let S_j contain s_{ij} samples of class C_i. The expected information based on this partitioning by A is known as the entropy of A. The entropy is the weighted average, shown by [20]:

$$E(A) = \sum_{j=1}^{v} \frac{(s_{1j} + \cdots + s_{mj})}{s} I(s_{1j}, \ldots, s_{mj})$$

Information Gain obtained by this partitioning on A is defined by [20]:

$$\text{Gain(A)} = \text{I}\,(\text{s}_1, \text{s}_2, \ldots, \text{s}_\text{m}) - \text{E(A)}$$

Information Gain was run on the 80 attribute dataset using Weka [19]. All the attributes with zero information gain were not used in further analysis, since these attributes would have no contribution to the classification process. On average, there were 19 attributes from each of the families that had zero information gain. Fig. 2 shows the number of attributes retained after information gain of zero, less than 0.01, 0.02, 0.03 and 0.04, were used to remove the attributes with those respective information gain values.

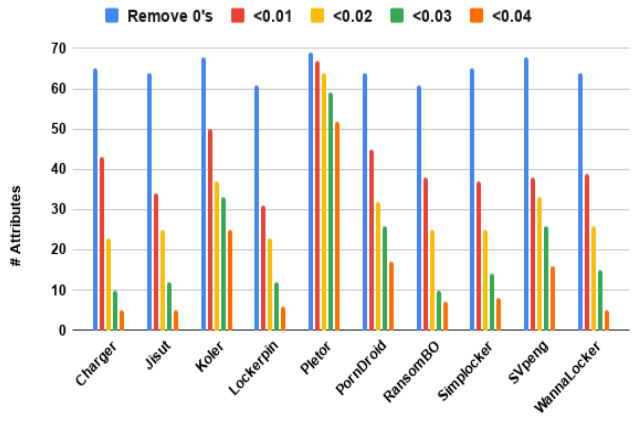

Fig. 2. Number of attributes retained in each Ransomware family after Information Gain based trimming.

Out of all the ransomware families, Pletor kept the most attributes at each of those information gain value cut-offs. Even with a cut-off of <0.04 information gain, there were still 52 attributes. This might be due to the less amount of data in the Pletor ransomware family. There were only 4,716 lines of captured network traffic for the Pletor family, in comparison to most other families having an average of 38,249 lines. Pletor was very much underrepresented.

V. Classification Algorithms

The J48 Decision Tree, Naïve Bayes and OneR classifiers were used in this work. All three classifiers were run using Weka [19].

A. J48 Decision Tree

There are multiple decision tree algorithms. In this work, the J48 decision tree algorithm, which is a Java reimplementation of the C4.5 algorithm [20], is used. C4.5 is an extension of Quinlan's earlier ID3 algorithm [21]. J48 is one of the most popular decision tree algorithms.

A decision tree is a tree structured representation where each node represents a feature and each link or branch represents a decision. The leaf represents the result. In the decision tree algorithm, Information Gain is used to decide the ordering of the features in the nodes, and the feature with the highest information gain is at the root of a decision tree. Information gain is re-calculated based on the branches, and the feature with the next highest information gain on every branch is chosen as the next root, and this process occurs recursively until the tree reaches the leaf node or has one class, attack or benign. Hence, every node and branch prior to the leaf node is associated with a decision function [20].

B. Naïve Bayes

Naive Bayes is a simple statistical classifier based on the Bayes' theorem that predicts class membership probabilities, that is, the probability that a given tuple belongs to a particular class [20]. The Naïve Bayes classifier assumes class conditional independence. This means that the effect of an attribute value on a given class is independent of the values of the other attributes. The Naïve Bayes classifier does not perform as well in the presence of irrelevant [22] or highly correlated attributes [23], hence feature selection is very important for the Naïve Bayes classifier.

C. OneR

OneR or One Rule is a simple classifier that, for each value of each attribute, finds the most frequent class and assigns the most frequent class to that attribute value. The attribute with the smallest error rate is selected [24, 25].

The algorithm for OneR is [24, 25]:

```
For each attribute,
   For each value of the attribute,
   Make a rule as follows:
        Count how often each class appears
        Find the most frequent class
        Make the rule assign that class to this
   attribute value
   Calculate the error rate of this attribute's rules
Choose the attribute with the smallest error rate
```

Hence, there is one branch for each value and each branch gets assigned the most frequent class.

Though it is not a commonly used classifier, OneR does well in datasets that do not embody very complex rules [26, 27], and has achieved high accuracy in many datasets [26, 27]. It classifies an object on the basis of a single attribute, that is, they are one-level decision trees. OneR is a fast classifier, but the downside is the accuracy of the classifier, so it depends on the importance of the execution time to the accuracy if it would be used.

VI. Results and Discussion

For the results, first the top ten attributes, based on information gain, for each ransomware family are presented. Next the top ten attributes overall, for all the ransomware families taken together (based on frequency of information gain) are presented, and finally classification results are presented.

A. Top Ten Attributes for Each Ransomware Family

The top 10 attributes for each family were extracted based on their information gain (IG), as shown in Tables 1 – X. The higher the rank of the attribute, the more significant the attribute, based on information gain. So, a rank of one is the most important attribute based on information gain. It can be noted that the information gain values for the top ten attributes for Pletor were generally higher than the information gain values of the other ransomware families.

Table I
Charger's top 10 attributes

Rank	Attribute	IG
1	Init_Win_bytes_forward	0.0749
2	Init_Win_bytes_backward	0.0726
3	Fwd Packet Length Max	0.0501
4	Flow Duration	0.0499
5	Flow IAT Max	0.0474
6	Flow Packets/s	0.0459
7	Flow IAT Min	0.0446
8	Flow IAT Mean	0.0417
9	Fwd Packets/s	0.0412
10	Average Packet Size	0.0389

Table II
Jisut's top 10 attributes

Rank	Attribute	IG
1	Init_Win_bytes_forward	0.076
2	Init_Win_bytes_backward	0.0719
3	Source Port	0.0518
4	Fwd Packet Length Max	0.0433
5	Flow IAT Max	0.0387
6	Max Packet Length	0.0342
7	Packet Length Mean	0.0334
8	Flow Duration	0.0323
9	Average Packet Size	0.0306
10	Subflow Fwd Bytes	0.03

Table III
Koler's top 10 attributes

Rank	Attribute	IG
1	Init_Win_bytes_backward	0.1047
2	Init_Win_bytes_forward	0.0973
3	Flow IAT Max	0.0748
4	Flow IAT Min	0.0597
5	Flow Duration	0.0585
6	Fwd Packet Length Max	0.0575
7	Flow IAT Mean	0.0553
8	Fwd IAT Min	0.0537
9	Fwd Packets/s	0.0522
10	Flow Packets/s	0.0512

Table IV
Lockerpin's top 10 attributes

Rank	Attribute	IG
1	Init_Win_bytes_forward	0.0671
2	Init_Win_bytes_backward	0.0645
3	Flow IAT Min	0.0535
4	Flow IAT Max	0.0463
5	Flow Duration	0.042
6	Fwd IAT Min	0.0388
7	Fwd Packet Length Max	0.0373
8	Bwd IAT Min	0.0329
9	Flow Packets/s	0.0308
10	Packet Length Std	0.0306

Table V
Pletor's top 10 attributes

Rank	Attribute	IG
1	Init_Win_bytes_forward	0.3946
2	Destination Port	0.2662
3	Init_Win_bytes_backward	0.2617
4	Fwd Packet Length Max	0.2516
5	Fwd Header Length	0.2163
6	Max Packet Length	0.2002
7	Subflow Fwd Bytes	0.1958
8	Total Length of Fwd Packets	0.1958
9	Source Port	0.1938
10	Fwd IAT Min	0.1714

Table VI
PornDroid's top 10 attributes

Rank	Attribute	IG
1	Init_Win_bytes_backward	0.0932
2	Init_Win_bytes_forward	0.0848
3	Flow IAT Max	0.0607
4	Flow IAT Min	0.0558
5	Fwd Packet Length Max	0.0547
6	Flow Duration	0.0509
7	Fwd Packets/s	0.0466

TABLE VI (CONTINUED)

8	Subflow Fwd Bytes	0.0451
9	Total Length of Fwd Packets	0.0451
10	Average Packet Size	0.0419

TABLE VII
RANSOMBO'S TOP 10 ATTRIBUTES

Rank	Attribute	IG
1	Init_Win_bytes_backward	0.0774
2	Flow IAT Min	0.0582
3	Init_Win_bytes_forward	0.0554
4	Flow IAT Max	0.0523
5	Fwd IAT Min	0.0492
6	Flow Duration	0.0489
7	Flow IAT Mean	0.0378
8	Fwd Packet Length Max	0.036
9	Fwd IAT Max	0.0303
10	Fwd Packets/s	0.0286

TABLE VIII
SIMPLOCKER'S TOP 10 ATTRIBUTES

Rank	Attribute	IG
1	Init_Win_bytes_backward	0.0798
2	Flow IAT Min	0.0617
3	Init_Win_bytes_forward	0.0522
4	Flow IAT Max	0.0512
5	Flow Duration	0.0495
6	Fwd Packet Length Max	0.0467
7	Flow Packets/s	0.0408
8	Fwd IAT Min	0.0399
9	Flow IAT Mean	0.0392
10	Fwd Packets/s	0.0348

TABLE IX
SVPENG'S TOP 10 ATTRIBUTES

Rank	Attribute	IG
1	Flow IAT Min	0.1171
2	Init_Win_bytes_backward	0.1022
3	Init_Win_bytes_forward	0.0909
4	Flow IAT Max	0.0789
5	Fwd IAT Min	0.0764
6	Flow Duration	0.072
7	Bwd IAT Min	0.0717
8	Flow Packets/s	0.0638
9	Flow IAT Mean	0.0632
10	Fwd Packets/s	0.0611

TABLE X
WANNALOCKER'S TOP 10 ATTRIBUTES

Rank	Attribute	IG
1	Init_Win_bytes_backward	0.0843
2	Init_Win_bytes_forward	0.06
3	Fwd Packet Length Max	0.0521
4	Flow IAT Min	0.0518
5	Total Length of Fwd Packets	0.0385
6	Subflow Fwd Bytes	0.0385
7	Packet Length Mean	0.0365
8	Flow IAT Max	0.0352
9	Packet Length Variance	0.0347
10	Packet Length Std	0.0347

Next, the attributes that were the most common amongst the ransomware families, based on their frequency within the top ten attributes by information gain, were identified. This is presented in Table XI. From Table XI, it can be noted that *Init_Win_bytes_forward* and *Init_Win_bytes_backward* occurred within the top ten attributes in all ten ransomware families. From Tables I - X, it can also be noted that these two attributes were the first two most important attributes in 6 of the 10 ransomware families and within the first three important attributes in the rest of the ransomware families. *Fwd Packet Length Max* occurred in nine of the ten ransomware families (it did not occur within the top ten in the SVPeng ransomware family) and *Flow IAT Max* also occurred in nine of the ten ransomware families (it did not occur within the top ten in the Pletor ransomware family); and so on.

TABLE XI
TOP 10 OVERALL ATTRIBUTES

Rank	**Attribute**	**Frequency**
1	Init_Win_bytes_forward	10
2	Init_Win_bytes_backward	10
3	Fwd Packet Length Max	9
4	Flow IAT Max	9
5	Flow IAT Min	8
6	Flow Duration	8
7	Fwd Packets/s	6
8	Flow Packets/s	5
9	Flow IAT Mean	5
10	Subflow Fwd Bytes	4

A brief explanation of all the attributes of Table XI follows.

1. *Init_Win_bytes_forward:* The total number of bytes sent in the initial window in the forward direction.
2. *Init_Win_bytes_backward:* The total number of bytes sent in the initial window in the backward direction.
3. *Fwd Packet Length Max:* The maximum value in bytes in the forward direction.
4. *Flow IAT Max:* The maximum value of the inter-arrival time of the flow.
5. *Flow IAT Min:* The minimum value of the inter-arrival time of the flow.
6. *Flow Duration:* The entire duration of the network flow.
7. *Fwd Packets/s:* The number of packets per second in the forward direction.
8. *Flow Packets/s:* The number of packets per second in the flow.

9. *Flow IAT Mean:* The mean value of the inter-arrival time of the flow.
10. *Subflow Fwd Bytes:* The average number of packets in a subflow in the forward direction.

B. Classification Results

1) Evaluation Metrics

Classification results were measured in terms of Accuracy, Precision, Recall the F-Measure.

Accuracy is the ratio of a model's correct data (TP + TN) to the total data, calculated by:

$$\frac{(TP+TN)}{Total\ number\ of\ instances} \quad (1)$$

Precision is the positive predictive value, or the percent of attack instances that are truly classified as attacks. Precision is calculated by:

$$\frac{TP}{TP+FP} \quad (2)$$

Recall or Attack Detection Rate (ADR) or sensitivity is the effectiveness of a model in identifying an attack. The objective is to get a higher ADR. Recall or ADR is calculated by:

$$\frac{TP}{TP+FN} \quad (3)$$

And F-measure is the harmonic mean of precision and recall. The higher the F-measure, the more robust the classification model. The F-measure is calculated by:

$$2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (4)$$

TP is true positive, FP is false positive, FN is false negative and TN is true negative. True positive is the number of ransomware attacks that are correctly classified as attacks. False positive is the number of ransomware attacks that should actually be classified as benign. True negative is the number of benign cases correctly classified as benign. False negative is the number of benign instances that should be classified as a ransomware attack.

2) Running the Classifiers

The classifiers were run using Weka with 10-fold cross validation. The following configurations were used in Weka:

- J48: batchSize = 100, confidenceFactor = 0.25, and minNumObj = 2
- Naive Bayes: batchSize = 100 and numDecimalPlaces = 2
- OneR: batchSize = 100, minBucketSize = 6, and numDecimalPlaces = 2

The classifiers were run as binary classifiers, that is, each run contained one ransomware attack and benign data, at a ratio close to 50% for each part. Each ransomware attack family data was run using the top ten attributes (as presented in Table XI), and then averaged (by attack). Figure 3 presents the average accuracy for each ransomware attack by each classifier. Pletor had the highest accuracy rate for all three classifiers, close to 94.8%. Overall, it can be noted that J48 had a higher accuracy than the other two classifiers. Naïve Bayes performed better than OneR. Since OneR performed the lowest, we can assume that no one attribute will do a good job in identifying attacks in this data and that there is somewhat complex relationship between the attributes in this data.

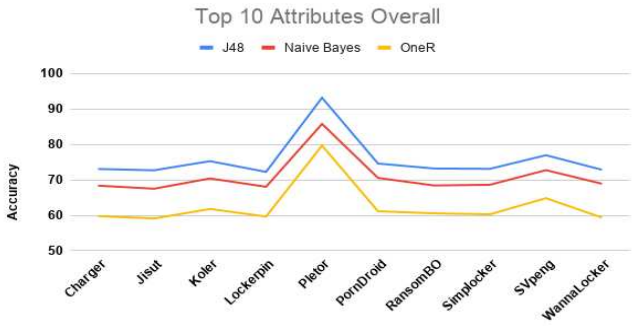

Fig. 3. Accuracy of J48, Naive Bayes, and OneR classifiers by ransomware family using the top 10 attributes

The average precision, recall, and F-measure of the three classifiers, J48, Naïve Bayes, and OneR, are presented in Table XII. As previously presented, these runs were also performed as binary classifiers (per attack family) using the top 10 attributes presented in Table XI, and then averaged for all attacks. From Table XII it can be noted that J48 had the highest precision and OneR had the lowest precision. J48 also had the highest recall and F-measure, while OneR had the lowest recall and F-measures.

TABLE XII
CLASSIFIERS' AVERAGE PRECISION, RECALL, AND F-MEASURE

	Precision	Recall	F-Measure
J48	0.7576	0.7573	0.757
Naive Bayes	0.7101	0.7094	0.7093
OneR	0.6272	0.6264	0.6257

From Figure 3 and Table XII it can be observed that J48 performed the best, hence in Table XIII we present the details of only the J48 runs for each ransomware attack for the different number of attributes with cut offs of zero, less than 0.01, 0.02, 0.03 and 0.04, based on information gain. Table XIII presents the execution time in addition to all the statistical metrics. It can be noted that lower execution times come with a lower number of attributes, but often at the cost of accuracy and other metrics. For most ransomware families, a reduced number of attributes did not reduce the accuracy very significantly, so for an effective ransonware intrusion detection system solution, it will be best to determine the number of attributes to use based on both accuracy as well as the time requirements to train the model.

TABLE XIII
J48 STATISTICAL METRICS FOR ALL RANSOMWARE FAMILIES

	No of Attributes	TP Rate	FP Rate	Accuracy	Precision	Recall	F Measure	ROC Area	Time to build model (sec)
WannaLocker	64	0.773	0.227	77.33%	0.773	0.773	0.773	0.847	1.16
	39	0.763	0.237	76.27%	0.763	0.763	0.763	0.841	0.72
	26	0.748	0.252	74.83%	0.749	0.748	0.748	0.83	0.42
	15	0.743	0.257	74.31%	0.743	0.743	0.743	0.826	0.23
	5	0.701	0.299	70.12%	0.701	0.701	0.701	0.785	0.07
Svpeng	68	0.806	0.194	80.63%	0.806	0.806	0.806	0.878	2.04
	38	0.794	0.206	79.38%	0.794	0.794	0.794	0.872	1.3
	33	0.792	0.208	79.15%	0.792	0.792	0.792	0.87	1.16
	26	0.79	0.21	78.98%	0.79	0.79	0.79	0.869	0.95
	16	0.783	0.217	78.31%	0.783	0.783	0.783	0.863	0.63
Simplocker	65	0.776	0.224	77.60%	0.776	0.776	0.776	0.849	1.38
	37	0.759	0.241	75.85%	0.759	0.759	0.759	0.837	0.74
	25	0.755	0.245	75.47%	0.755	0.755	0.755	0.834	0.49
	14	0.747	0.253	74.73%	0.747	0.747	0.747	0.826	0.31
	8	0.721	0.279	72.06%	0.722	0.721	0.72	0.802	0.15
RansomBO	61	0.768	0.232	76.83%	0.768	0.768	0.768	0.842	1.53
	38	0.76	0.24	75.95%	0.76	0.76	0.759	0.835	0.89
	25	0.755	0.245	75.51%	0.755	0.755	0.755	0.831	0.55
	10	0.73	0.27	73%	0.73	0.73	0.73	0.808	0.22
	7	0.711	0.289	71.06%	0.711	0.711	0.711	0.786	0.14
PornDroid	64	0.787	0.213	78.65%	0.787	0.787	0.786	0.861	1.74
	45	0.778	0.222	77.83%	0.778	0.778	0.778	0.853	1.15
	32	0.768	0.232	76.82%	0.768	0.768	0.768	0.847	0.85
	26	0.768	0.232	76.82%	0.768	0.768	0.768	0.846	0.75
	17	0.764	0.236	76.35%	0.764	0.764	0.763	0.843	0.47
Pletor	69	0.949	0.051	94.86%	0.949	0.949	0.949	0.974	0.05
	67	0.949	0.051	94.86%	0.949	0.949	0.949	0.974	0.05
	64	0.948	0.052	94.82%	0.948	0.948	0.948	0.974	0.04
	59	0.949	0.051	94.85%	0.949	0.949	0.949	0.974	0.04
	52	0.95	0.05	94.98%	0.95	0.95	0.95	0.976	0.03
Lockerpin	61	0.755	0.245	75.46%	0.755	0.755	0.755	0.829	0.74
	31	0.74	0.26	73.99%	0.74	0.74	0.74	0.816	0.34
	23	0.737	0.263	73.73%	0.737	0.737	0.737	0.814	0.26
	12	0.731	0.269	73.11%	0.731	0.731	0.731	0.81	0.14
Koler	68	0.793	0.207	79.32%	0.793	0.793	0.793	0.867	1.84
	50	0.79	0.21	78.97%	0.79	0.79	0.79	0.864	1.3
	37	0.784	0.216	78.41%	0.784	0.784	0.784	0.86	1.01
	33	0.782	0.218	78.24%	0.782	0.782	0.782	0.86	0.93
	25	0.774	0.226	77.37%	0.774	0.774	0.774	0.854	0.71
Jisut	64	0.783	0.217	78.28%	0.783	0.783	0.783	0.852	0.71
	34	0.772	0.228	77.17%	0.772	0.772	0.772	0.846	0.4
	25	0.768	0.232	76.80%	0.768	0.768	0.768	0.847	0.31
	12	0.752	0.248	75.24%	0.752	0.752	0.752	0.832	0.16
	5	0.702	0.298	70.18%	0.702	0.702	0.702	0.789	0.06
Charger	65	0.774	0.226	77.42%	0.774	0.774	0.774	0.851	1.15
	43	0.763	0.237	76.33%	0.763	0.763	0.763	0.842	0.98
	23	0.751	0.249	75.05%	0.751	0.751	0.751	0.834	0.47
	10	0.73	0.27	73%	0.73	0.73	0.73	0.814	0.21
	5	0.71	0.29	71.04%	0.71	0.71	0.71	0.796	0.09

VII. CONCLUSION

In this study we presented the ten most important attributes that can be used to classify Android ransomware network traffic. The ten attributes are presented for each attack family individually as well as for all the ransomware families together. The study also shows that the J48 classifier had the highest classification of the three classifiers, J48, Naïve Bayes and OneR. Shorter execution times can also be achieved with a lower number of attributes, using the J48 classifier.

ACKNOWLEDGEMENT

This work has been partially by the Askew Institute of the University of West Florida, Pensacola, Florida, USA.

REFERENCES

[1] "Mobile Operating System Market Share North America," StatCounter Global Stats. [Online]. Available: https://gs.statcounter.com/os-market-share/mobile/north-america.

[2] "Mobile Operating System Market Share Worldwide," StatCounter Global Stats. [Online]. Available: https://gs.statcounter.com/os-market-share/mobile/worldwide.
[3] "2019 State of Malware," Malwarebytes. [Online]. Available: https://resources.malwarebytes.com/files/2019/01/Malwarebytes-Labs-2019-State-of-Malware-Report-2.pdf.
[4] A. H. Lashkari, A. F. A. Kadir, L. Taheri, and A. A. Ghorbani, "Toward Developing a Systematic Approach to Generate Benchmark Android Malware Datasets and Classification," 2018 International Carnahan Conference on Security Technology (ICCST), 2018.
[5] M. Talal, A. A. Zaidan, B. B. Zaidan, O. S. Albahri, M. A. Alsalem, A. S. Albahri, et al., "Comprehensive review and analysis of anti-malware apps for smartphones," Telecommunication Systems, vol. 72(4), pp. 285–337, 2019, doi: 10.1007/s11235-019-00575-7.
[6] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization," Proceedings of the 4th International Conference on Information Systems Security and Privacy, 2018.
[7] Y. Zhou and X. Jiang, "Dissecting Android Malware: Characterization and Evolution," 2012 IEEE Symposium on Security and Privacy, 2012.
[8] O. M. K. Alhawi, J. Baldwin, and A. Dehghantanha, "Leveraging Machine Learning Techniques for Windows Ransomware Network Traffic Detection," Advances in Information Security Cyber Threat Intelligence, 2018, pp. 93–106.
[9] S. Alsoghyer and I. Almomani, "Ransomware Detection System for Android Applications," Electronics, vol. 8, no. 8, p. 868, May 2019.
[10] M. Baykara and E. Çolak, "A review of cloned mobile malware applications for android devices," 2018 6th International Symposium on Digital Forensic and Security (ISDFS), Antalya, 2018, pp. 1-5, doi: 10.1109/ISDFS.2018.8355388.
[11] R. Zachariah, K. Akash, M. S. Yousef and A. M. Chacko, "Android malware detection a survey," 2017 IEEE International Conference on Circuits and Systems (ICCS), Thiruvananthapuram, 2017, pp. 238-244, doi: 10.1109/ICCS1.2017.8325997.
[12] S, Kalpana & Karthikeyan, S, "A survey on rise of mobile malware and detection methods," 2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS), pp. 1–5, 2017, doi: 10.1109/ICIIECS.2017.8276158.
[13] M. T. Ahvanooey, Q. Li, M. Rabbani, and A. R. Rajput, "A Survey on Smartphones Security: Software Vulnerabilities, Malware, and Attacks," International Journal of Advanced Computer Science and Applications, vol. 8(10), pp. 30–45, 2017.
[14] K. Chen, Y. Zhang and P. Liu, "Leveraging Information Asymmetry to Transform Android Apps into Self-Defending Code Against Repackaging Attacks," in IEEE Transactions on Mobile Computing, vol. 17, no. 8, pp. 1879-1893, 1 Aug. 2018, doi: 10.1109/TMC.2017.2782249.
[15] C. Wang, Z. Li, X. Mo, H. Yang, and Y. Zhao, "An Android Malware Dynamic Detection Method Based on Service Call Co-occurrence Matrices," Annals of Telecommunications, vol. 72, pp. 607–615, May 2017.
[16] P. Gopalakrishnan, S. Narayanan, R. Kamath, A. Ramani, "Analyzing Diverse Data Mining Techniques to Detect the Malware based on Signature," Test Engineering and Management, vol. 83, pp. 17717-17724, April 2020.
[17] "Search UNB," University of New Brunswick est.1785. [Online]. Available: https://www.unb.ca/cic/datasets/andmal2017.html.
[18] "CICFlowMeter (Formerly ISCXFlowMeter)," Network Traffic Flow analyzer. [Online]. Available: http://www.netflowmeter.ca/netflowmeter.html.
[19] Weka 3: Machine Learning Software in Java, Weka 3 - Data Mining with Open Source Machine Learning Software in Java. [Online]. Available: https://www.cs.waikato.ac.nz/ml/weka/. [Accessed: 22-August 2020]
[20] J. Han, and M. Kamber, Data Mining: Concepts and Techniques, CA: Morgan Kaufmann Publishers, USA, 2012.
[21] J. R. Quinlan, C4.5: Programs for Machine Learning, San Mateo, CA: Morgan Kaufmann Publishers, 1993.
[22] P. Bermejo, J. A. Gamez, and J. M. Puerta, "Speeding up Incremental Wrapper Feature Subset Selection, with Naïve Bayes Classifier," Knowledge-Based Systems, vol. 55, pp. 140-147, January 2014.
[23] I. Inza, P. Larranga, R. Etxeberria, and B. Sierra, "Feature Subset Selection by Bayesian network-based optimization," Artificial Intelligence, vol. 123(1-2), pp. 157-184, October 2000.
[24] "Machine Learning - (One: Simple) Rule - (One Level Decision Tree)," Machine Learning - (One|Simple) Rule - (One Level Decision Tree). [Online]. Available: https://datacadamia.com/data_mining/one_rule.
[25] OneR. [Online]. Available: https://www.saedsayad.com/oner.htm.
[26] G. Holmes, C. Nevill-Manning and I. Witten, "The Development of Holte's 1R Classifier," in Proceedings 1995 Second New Zealand International Two-Stream Conference on Artificial Neural Networks and Expert Systems, Dunedin, New Zealand, 1995 pp. 239, doi: 10.1109/ANNES.1995.499480.
[27] R. C. Holte, "Very Simple Classification Rules Perform Well on Most Commonly Used Datasets," Machine Learning, vol. 11, pp. 63-91, 1993.

AUTHORS PROFILE

Dr. Sikha Bagui is Professor and Askew Fellow in the Department of Computer Science, at The University West Florida, Pensacola, Florida. Dr. Bagui is active in publishing peer reviewed journal articles in the areas of database design, data mining, BigData and Big Data analytics, and machine learning. Dr. Bagui has worked on funded as well unfunded research projects and has over 75 peer reviewed publications. She has also co-authored several books on database and SQL. Bagui also serves as Associate Editor and is on the editorial board of several journals.

Mr. Tyler Woods has a Bachelor's degree in Computer Science from The University of West Florida.

APPENDIX A

The 84 network traffic features [15] of the CICAndMal2017 dataset. The ten most important attributes are marked with an asterisk.

Attribute Name	**Description**
Flow ID	(Destination IP) - (Source IP) - (Destination Port) - (Source Port) - (Protocol)
Source IP	Source IP address of flow.
Source Port	Source port number.
Destination IP	Destination IP address.
Destination Port	Destination port number.
Protocol	Transport layer protocol number identification. TCP = 6, UDP = 17
Timestamp	Date and time that the packet was captured. Format: Dd/mm/yyyy-HH:MM:SS
*Flow Duration	Duration of network flow.
Total Fwd Packets	Number of packets in forward direction.
Total Backward Packets	Number of packets in backward direction.
Total Length of Fwd Packets	Number of bytes in forward direction from all packets.
Total Length of Bwd Packets	Number of bytes in backward direction from all packets.
*Fwd Packet Length Max	Maximum value in bytes in forward direction.
Fwd Packet Length Min	Minimum value in bytes in forward direction.
Fwd Packet Length Mean	Mean value in byes in forward direction.
Fwd Packet Length Std	Standard deviation in bytes in forward direction.
Bwd Packet Length Max	Maximum value in bytes in backward direction.
Bwd Packet Length Min	Minimum value in bytes in backward direction.
Bwd Packet Length Mean	Mean value in byes in backward direction.
Bwd Packet Length Std	Standard deviation in bytes in backward direction.
Flow Bytes/s	Number of bytes per second of flow.
*Flow Packets/s	Number of packets per second of flow.
*Flow IAT Mean	Mean value of inter-arrival time of flow.
Flow IAT Std	Standard deviation of inter-arrival time of flow.
*Flow IAT Max	Maximum value of inter-arrival time of flow.
*Flow IAT Min	Minimum value of inter-arrival time of flow.
Fwd IAT Total	Total inter-arrival time in forward direction.
Fwd IAT Mean	Mean inter-arrival time in forward direction.
Fwd.IAT.Std	Standard inter-arrival time in forward direction
Fwd.IAT.Max	Maximum value of inter-arrival time in forward direction
Fwd.IAT.Min	Minimum value of inter-arrival time in forward direction
Bwd.IAT.Total	Total inter-arrival time in backward direction.
Bwd.IAT.Mean	Mean inter-arrival time in backward direction.
Bwd.IAT.Std	Standard inter-arrival time in backward direction.
Bwd.IAT.Max	Maximum value of inter-arrival time in backward direction.
Bwd.IAT.Min	Minimum value of inter-arrival time in backward direction
Fwd.PSH.Flags	Number of times packets sent in flow had pushing flag bit set to 1 in forward direction. Pushing flag allows to send information immediately without filling the buffer; useful for real time applications.
Bwd.PSH.Flags	Number of times packets sent in flow had PSH (pushing) flag bit set to 1 in backward direction.
Fwd.URG.Flags	Number of times packets sent in flow had URG (Urgent) flag bit set to 1 in forward direction. URG flag informs a receiving station that certain data within a segment is urgent and should be prioritized.
Bwd.URG.Flags	Number of times packets sent in flow had URG (Urgent) flag bit set to 1 in backward direction.
Fwd.Header.Length	Header length of packets flowing in forward direction.
Bwd.Header.Length	Header length of packets flowing in backward direction.
*Fwd.Packets/s	Number of packets per second in forward direction.

Bwd.Packets.s	Number of packets per second in backward direction.
Min.Packet.Length	Minimum length of packets registered in flow (both forward and backward directions).
Max.Packet.Length	Maximum length of packets registered in flow (both forward and backward directions).
Packet.Length.Mean	Mean value of length of the packets registered in flow (both forward and backward directions).
Packet.Length.Std	Standard deviation of length of the packets registered in flow (both forward and backward directions).
Packet.Length.Variance	Variance of length of packets registered in flow (both forward and backward directions).
FIN.Flag.Count	Number of times packets sent in flow had FIN flag bit set to 1. Each side terminates its connection by sending a message with FIN (finish) bit set.
SYN.Flag.Count	Number of times packets sent in flow (in both directions) had SYN (Synchronize) flag bit set to 1. SYN (Synchronize) flag is the TCP packet flag used to initiate a TCP connection. A packet containing solely a SYN flag is the first part of the "three-way handshake" of TCP connection initiation. It is responded to with a SYN-ACK packet.
RST.Flag.Count	Number of times packets sent in flow (in both directions) had RST (Reset) flag bit set to 1. RST reset the connection. FIN says, "I finished talking to you, but I'll still listen to everything you have to say until you're done."
PSH.Flag.Count	Number of times packets sent in flow (in both directions) had PSH (Pushing) flag bit set to 1.
ACK.Flag.Count	Number of times packets sent in flow (in both directions) had ACK (Acknowledged) flag bit set to 1. To establish a connection, TCP uses a three-way handshake. Before a client attempts to connect with a server, the server must first bind to and listen at a port to open it up for connections: this is called a passive open. Once the passive open is established, a client may initiate an active open.
URG.Flag.Count	Number of times packets sent in flow (in both directions) had URG (Urgent) flag bit set to 1.
CWE.Flag.Count	Number of times packets sent in flow (in both directions) had CWR (Congestion Window Reduced) TCP flag set to 1.
ECE.Flag.Count	Number of times packets sent in flow (in both directions) had ECE (Explicit Congestion Notification Echo) TCP flag set to 1.
Down.Up.Ratio	Download and upload ratio.
Average.Packet.Size	Average size of each packet.
Avg.Fwd.Segment.Size	Average segment size observed in forward direction. A TCP segment is the Protocol Data Unit (PDU) which consists of a TCP header and an application data piece which comes from the upper Application Layer. Transport layer data is generally called segment and network layer data unit is generally called datagram, but when UDP is used as transport layer protocol, the data unit is called UDP datagram since the UDP data unit is not segmented (segmentation is made in transport layer when TCP is used).
Avg.Bwd.Segment.Size	Average Segment size observed in backward direction.
Fwd.Header.Length	Header length of packets flowing in forward direction.
Fwd.Avg.Bytes.Bulk	Average number of bytes bulk rate in forward direction. Bulk data transfer is a software-based mechanism designed to move large data files using compression, blocking and buffering to optimize transfer times.
Fwd.Avg.Packets.Bulk	Average number of packets bulk rate in forward direction.
Fwd.Avg.Bulk.Rate	Average number of bulk rate in forward direction.
Bwd.Avg.Bytes.Bulk	Average number of bytes bulk rate in backward direction.
Bwd.Avg.Packets.Bulk	Average number of packets bulk rate in backward direction.
Bwd.Avg.Bulk.Rate	Average number of bulk rate in backward direction.
Subflow.Fwd.Packets	Average number of packets in a subflow in forward direction.
*Subflow.Fwd.Bytes	Average number of bytes in a subflow in forward direction.
Subflow.Bwd.Packets	Average number of packets in a subflow in backward direction.
Subflow.Bwd.Bytes	Average number of bytes in a subflow in backward direction.
*Init_Win_bytes_forward	Total number of bytes sent in initial window in forward direction.
*Init_Win_bytes_backward	Total number of bytes sent in initial window in backward direction.
act_data_pkt_fwd	Count of packets with at least one byte of TCP data payload in forward direction.
min_seg_size_forward	Minimum segment size observed in forward direction.
Active.Mean	Mean time a flow was active before becoming idle.
Active.Std	Standard deviation time a flow was active before becoming idle.
Active.Max	Maximum time a flow was active before becoming idle.
Active.Min	Minimum time a flow was active before becoming idle.
Idle.Mean	Mean time a flow was idle before becoming active.
Idle.Std	Standard deviation of time a flow was idle before becoming active.
Idle.Max	Maximum time a flow was idle before becoming active.
Idle.Min	Minimum time a flow was idle before becoming active.
Label	Classification of flow (adware, benign, ransomware, etc.)