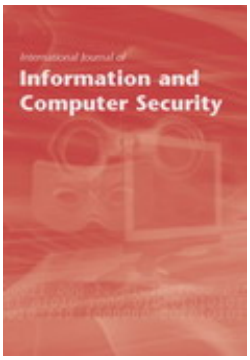


International Journal of Information and Computer Security

ISSN online: 1744-1773 - ISSN print: 1744-1765
https://www.inderscience.com/ijics

Behavioural analysis and results of malware and ransomware using optimal behavioural feature set

Laxmi B. Bhagwat, Balaji M. Patil

DOI: 10.1504/IJICS.2023.10059326

Article History:

Received:	01 October 2022
Last revised:	01 March 2023
Accepted:	20 March 2023
Published online:	19 February 2024

Behavioural analysis and results of malware and ransomware using optimal behavioural feature set

Laxmi B. Bhagwat* and Balaji M. Patil

School of Computer Engineering and Technology,
Dr. Vishwanath Karad World Peace University,
Pune, India
Email: laxmi.bhagwat@mitwpu.edu.in
Email: balaji.patil@mitwpu.edu.in
*Corresponding author

Abstract: Ransomware is the subset of malware that is considered the most jeopardising malware. In a malware/ransomware attack, attacker encrypts all the essential data files and demands the ransom to get all the important files that it has encrypted. There are many methods or techniques, but a dynamic approach is used by many researchers to detect malware/ransomware attack. In the dynamic approach, the behavioural characteristics play a very important role in the detection of ransomware attacks. This paper presents a comprehensive analysis for the selection of the optimal behavioural feature set using various feature selection techniques. As a unique part of our research, we have found the relation and the difference between the features that can be common or different for malware and ransomware detection. We have obtained the optimal feature set for malware as well as ransomware and obtained an accuracy of 90% for XGBoost and 96.22 for KNN, respectively.

Keywords: ransomware; feature selection; malware; machine learning; Windows API calls; dynamic detection technique.

Reference to this paper should be made as follows: Bhagwat, L.B. and Patil, B.M. (2024) 'Behavioural analysis and results of malware and ransomware using optimal behavioural feature set', *Int. J. Information and Computer Security*, Vol. 23, No. 1, pp.57–78.

Biographical notes: Laxmi B. Bhagwat is a graduate in Computer Science and Engineering and completed her post-graduation in Information Technology. She is a certified Cyber Crime Investigator. She is currently pursuing her PhD in Cyber Forensics. She has a total experience of 23 years in the Nobel profession of teaching. Her research of interest is algorithms and data structures, cyber security and cyber forensics.

Balaji M. Patil is a Professor in Computer Engineering Department of MIT Pune. He received his PhD in Computer Science and Engineering from Uttrarakhand Technical University, Dehradoon, India. He has 23 years of teaching experience and working with MITWPU Pune as an AHoS. During this duration, he has taught the subjects in the domain of Networking and Cyber security. His primary research interests include network management, cyber security, cyber forensics and IoT.

1 Introduction

As today's world is digitised and everyone makes use of digital devices with internet connectivity, these devices are susceptible to various malware attacks. The attacker is always having a tap on who can be the possible victims by finding vulnerabilities. The various type of malware like viruses, worms trojans, logic bombs, ransomware, adware, etc. is used to carry out malicious activity on the target machine. Among these types of malware, ransomware is the subset of malware that is extortion based. Recent statistics show that there is a 97% increase in malware attacks in the past two years.

Among these, a large portion is due to ransomware. Wannacry was the ransomware that caught the attention of the cyber security experts as there was huge damage done to the healthcare industry. Because of it, there is a significant threat to the world, as this is malware that generates high revenues and is creating a viable criminal business model. Hence, the systems of private companies, individuals, or public service providers are at stake and can suffer severe disruption and financial loss.

Ransomware is a subset of malware that is designed to target individuals or organisations. It is a type of malware that allows the attackers to gain full control of your system and restricts access to personal and confidential files unless a ransom is paid. The different phases of ransomware attack are deployment, installation, command-and-control, destruction, and extortion.

If we can capture the activities that are at the installation phase we can have early detection of the malware/ransomware attack and we can avoid the further damage to the system.

The research thus far did not focus on the behavioural characteristics of ransomware and malware all together. While performing the analysis of ransomware we had a question of "are the behavioural characteristics for the ransomware and malware the same?" Hence, after the literature survey we came to know that the mention of the characteristics was not done collectively thus far. So, in this paper we have done the research for the behavioural characteristics for both collectively. This is the novel part of our research.

Our contribution in this paper is as follows:

1 Methodology used for the detection of malware and ransomware both collectively.

2 Detail analysis of the behavioural characteristics to obtain the optimal feature set for the accurate detection of malware and ransomware.

3 We present the common behavioural features for malware and ransomware.

4 We present the list of different behavioural features for malware and ransomware which we selected as the API calls when a process is executing.

5 The category of the features (API calls) like file related, process related, etc. so that it can relate to a specific behaviour of the process while executing.

Though ransomware is a subset of ransomware there are certain behavioural characteristics that are different. The research thus far did not focus on finding the behavioural characteristics of both ransomware and malware collectively. We have done the analysis for both in this paper, which is a novel part of our research. The aim behind this analysis is it is assumed that when detection of malware or ransomware as malware using a static or dynamic approach means the set of features will be the same, but we

wanted to find the differences or similarities in the features for the malware and ransomware detection.

2 Related work

Vinod and Viswalakshmi (2018) have used the Scatter Assessment method for a succinct list of features. The features that were selected minimised and maximised the variances of inter and intra-class to elude mimicry attack.

Crypto ransomware is the most dangerous malware and it needed attention so Al-Rimy et al. (2018) have given a solution for the detection of crypto-ransomware by constructing an effective and efficient framework for early detection of such type of attack to that before the damage to encryption takes place.

Hampton et al. (2018) compared API calls for windows that are made by 14 different kinds of ransomware and were analysed by finding the frequency of the API calls.

NetConverse was introduced by Alhawi et al. (2018) for the detection of ransomware using the machine learning technique. The dataset was created using the network traffic. Using the network-based conversation logs the features were selected and the classification models were applied.

Abbasi et al. (2020) have proposed feature selection which is divided into two stages. One is the ransomware detection phase for which the optimal features were selected using the wrapper-based swarm optimisation algorithm for feature selection. The second stage was classification which gave a better performance for binary as well as a multi-class classification for group-wise ransomware feature selection.

Daku et al. (2018) have evaluated ransomware behaviour by using reports of 150 ransomware executable samples from various ransomware families. The main task was to identify the best behavioural attributes which were achieved by the iterative features selection method. Then machine learning algorithms were used for the classification of ransomware.

Maiorca et al. (2017) proposed a system API-based machine learning technique for the detection of ransomware for the Android platform. They have developed a malware detector R-PackDroid for can detect generic malware, ransomware, and benign samples with good accuracy.

Takeuchi et al. (2018) used SVM as the machine learning algorithm for the detection of ransomware by looking into the API calls that are made while ransomware samples are executed.

Fukushima et al. (2010) tried to reduce the false positive rate because ransomware will not do when it is executing. They have created the dataset for both benign and malware for the detection of malware and showed that the detection rate is 60% and less false positive rate.

Bhagwat and Patil (2020) have surveyed different methods that the researchers have given solutions for ransomware detection. They have also mentioned the limitations of the implemented solutions.

Sgandurra et al. (2016) presented a Machine Learning technique EldeRan which does analyses and classification of ransomware dynamically. They have achieved a ROC curve of 0.995.

Skaletsky et al. (2010) have implemented a process-level system to give a solution for the dynamic behaviour of ransomware on Windows. The solutions are given for the obstacles that are faced by the processes at the time of application/kernel transitions, etc.

Ramírez-Gallego et al. (2016) have presented a fast minimum redundancy maximum relevance algorithm on various platforms to beat the computational burden for sequential they have implemented three algorithms for fast mRMR.

Ahmed et al. (2020) had done deep experimentation and evaluation of ransomware behaviour. They have proposed an enhanced maximum-relevance and minimum-redundancy (EmRmR) method which is a filter to remove the noisy features from the feature set. They removed the feature which was redundant or did not have strong relevance when the API calls were traced for ransomware behaviour.

Bhagwat and Patil (2021) used the ransomware dynamic detection technique. They have used various machine learning algorithms and found the detection accuracy using these algorithms. KNN outperformed as compared to random forest (RF), SVM, and logistic regression (LR).

Pektaş and Acarman (2017) used the n-gram method for API-call sequences on the registry, filesystem, and network. Have used machine learning for the classification of samples as worms, viruses, and backdoors malware.

Shaukat and Ribeiro (2018) have used the hybrid technique of dynamic and static analysis to create a close-packed set of features. They present RansomWall detection of the cryptographic ransomware for different ransomware family samples. They have evaluated the different machine learning models for the detection of samples as benign or ransomware.

Salehi and Sami (2012) used two classes for feature selection. One was API names and the other was arguments to the API calls. They claim to have achieved higher accuracy for identifying a sample as malware or benign,

Pirscoveanu et al. (2015) have developed an experimental setup that includes a Cuckoo Sandbox for the samples to get executed for tracking the behaviour of malware.

Ki et al. (2015) adopted the concept of DNA sequencing to API call sequencing to pull out the patterns of malicious API calls for malware. Using these patterns, they could detect unknown samples of malware.

Alazab et al. (2011) have used data mining algorithms for zero-day malware detection by classification of codes as benign or malicious. For this, they have used the behavioural attributes as the API calls. They have done the analysis from the results of various data mining algorithms. Using this technique, they were able to detect the 0-day malware perfectly.

Hwang et al. (2020) divided the task of ransomware detection into two stages. The first stage is capturing the characteristics of the ransomware using the Markov model. And the second stage for detection accuracy used the RF machine learning model. Mixing the two-stage models achieved an accuracy of 97.3%.

Hampton et al. (2018) focused on the strains of 14 ransomwares which infect the Window OS. They have shown their results by doing the comparison of the normal behaviour of the operating system with the Windows API calls done by the ransomware.

Gržinić and González (2022) have given the summary of the methods used and tools for analysis of this era of malwares so that the researchers get to know the real picture and challenges for malware detection.

According to the literature survey we found the potential for research for dynamic analysis for Windows API calls as changes made by the malware and ransomware in the

OS level. So, we decided to research more on the behavioural attributes for Windows API calls.

3 Detection techniques used for malware/ransomware

As per our survey static and dynamic techniques were predominantly used by many researchers. As per Idika et al. (2017) and Liska and Gallo (2016), the quality of the malware detector is dependent upon the technique that is used. The different analysis and detection techniques these researchers have mentioned are Static and Dynamic detection techniques.

Static-based detection-detection of ransomware/malware using static-based technique means analysing an application's code before its execution to determine if it is capable of any malicious activities. Static detection is done by analysing the binary code for a certain pattern. This pattern is called as signature and hence while detection the signature is matched. If the signature of the executable is matched, then it is declared malicious and detected as ransomware/malware. This type of detection technique is not capable of detecting zero-day attacks.

Dynamic-based (behavioural) Detection-Dynamic-based analysis detection entails the live monitoring of processes, to determine if any are behaving with any malicious intent. So, at the time of execution of the malware/ransomware attempt is made for the detection. Zero-day attack detection is a major concern that can be done using dynamic detection and analysis techniques. For zero-day attack detection, the machine learning approach can be used. Vinod and Viswalakshmi (2018), Hampton et al. (2018), Abbasi et al. (2020), Daku et al. (2018), Maiorca et al. (2017) and Bhagwat and Patil (2021) used the machine learning technique for the detection of a ransomware attack.

4 Methodology

4.1 Experimental setup

We have generated our own dataset by having the ransomware executables execute in a virtual environment. We have used the most popular open-source Cuckoo Sandbox (Cuckoo Foundation, 2014–2019). The host machine was Ubuntu 18.04 version on which the cuckoo was installed with Windows XP as the virtual guest machine. The system architecture (Bhagwat and Patil, 2021) of the setup is shown in Figure 1.

There are three main components in the test bench we have created for our malware and ransomware detection:

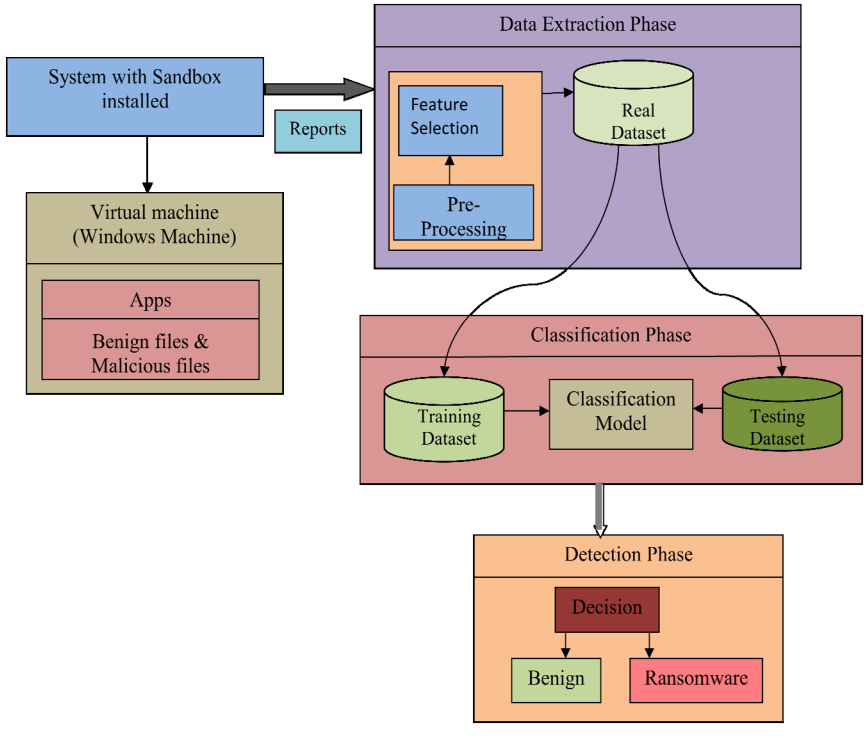
1 We have used the Ubuntu 16.04.7 LTS as the host machine. We have Cuckoo Sandbox installed with Windows operating system. We have used the Windows XP image ISO to install VM while installing the Cuckoo Sandbox. We have disabled all the security related settings like for firewall and antivirus protection and update for security so that we could get the real trace of all the activities that are carried by a ransomware attack.

2 Ransomware detection phase where all the creation of our dataset and feature selection is done.

3 Classification phase which does the training and testing of using our dataset and construct our model for machine learning which further is used for making decision for a file to be malicious or benign.

For a collection of the behavioural attributes around 155 ransomware files from VirusShare (http://www.virusshare.com) were given for analysis to the Cuckoo Sandbox which generates all the different types of report files. We have used the report.json file which contains all the behavioural logs of ransomware execution.

Figure 1 System architecture (see online version for colours)



As this JSON file is very large in size it is very difficult to scan all the report files manually. So we have written a parser for parsing this JSON file so that we can extract the relevant information for the analysis recorded by the Cuckoo Sandbox. The parser parses the JSON file for the behaviour tag and in the behaviour tag processes tag for the API calls at the time of app execution in the sandbox.

4.2 Dataset

Our first aim was to create a dataset for any type of malware and a specific type of malware, i.e., ransomware. The dataset for malware as well as ransomware was created separately. The procedure that was carried out for both was the same. We have manually created the dataset by gathering information from sites like VX Vault (http://vxvault.net), VirusShare (http://www.virusshare.com), portable freeware (Portable Freeware Collection, https://www.portablefreeware.com), and Softonic (Maiorca et al., 2017). We

have taken 250 malicious samples and 50–60 benign samples. In this way, a dataset of around 300 samples was created. All these samples that were downloaded were cross-checked and verified for their correctness on VirusTotal (http://www.virustotal.com).

The dataset was created by parsing the JSON file for the API calls that were called at the time of application executable execution. The API calls which are the features/ attributes for malware detection were searched in the report.json file. We were interested in the count of each call as the attribute value. We had written the parser and parsed the JSON file for the API calls and created the CSV file for the API (features/attribute) count. This file was used for feature selection and classification using Machine learning for the detection of an application file as ransomware or benign.

4.3 Feature selection for malware and ransomware

The accuracy of any machine learning algorithm depends on the set of features that are used at the time of training and testing of the classifier. So, it becomes essential to select the accurate no. of features so that the malware/ransomware detection is done accurately. Using various feature selection techniques, we can get the most relevant features that will increase the accuracy for the models we use for machine learning. There are three methods for the feature selection (https://www.simplilearn.com/tutorials/machine-learning-tutorial/feature-selection-in-machine-learning).

1 filter method

2 wrapper method

3 embedded method.

We have used all the three methods. ANOVA and chi^2 are filter methods, RFECV are wrapper methods and Extra Tree is an Embedded method. The following section describes the methods we have applied for the selection of optimal features as the feature set.

4.3.1 Feature selection for malware

Feature selection is the most important part in machine learning. The first task we performed was to find the list of features (API calls) that any type of malware has. Table 1 shows the list of 40 features for any kind of malware. We have 40 features (API calls) in the CSV file of the dataset that is created using the parser. The dataset we have created consists of some attributes whose contribution to the detection of malware is negligible and would only prove to hamper the training of the machine learning models. We have used recursive feature elimination and an extra tree classifier for feature selection. With the help of recursive feature selection, we got some features to be considered. With an extra tree classifier, we select the features to be taken into consideration based on the count.

4.3.1.1 Recursive feature elimination method

We got the following graph for the number of features against the percentage of correct classification using recursive feature elimination with cross-validation (RFECV). RFECV works by finding the subset of features from the complete list of features by using either a

machine learning algorithm or using any statistical method. We used decision tree (DT) classifier (Uppal, 2014) as the estimator which uses the Gini impurity for splitting the tree and measuring the accuracy in RFECV. RFECV is a wrapper method where the features are eliminated one by one whose contribution is less towards the contribution accuracy. This is an elimination recursive process unless the mentioned k no. of features are left. Using Gini impurity as the mathematical procedure and ordering it in descending order we got the top 11 features given by the RFECV feature selection method.

Table 1 Features of malware without feature selection

1	RegCloseKey
2	*NtReadFile*
3	*RegQueryValueExA*
4	NtQueryValueKey
5	*RegQueryValueExW*
6	*NtCreateThreadEx*
7	*RegOpenKeyExW*
8	*NtAllocateVirtualMemory*
9	*RegOpenKeyExA*
10	LdrGetDllHandle
11	LdrUnloadDll
12	NtQuerySystemInformation
13	*NtResumeThread*
14	*NtCreateFile*
15	GlobalMemoryStatusEx
16	NtOpenKey
17	LdrLoadDll
18	NtClose
19	Unnamed
20	*ReadProcessMemory*
21	*NtProtectVirtualMemory*
22	*NtWriteFile*
23	*SetFilePointer*
24	*RegCreateKeyExA*
25	NtDelayExecution
26	*CreateThread*
27	*NtOpenThread*
28	*RegSetValueExA*
29	UuidCreate
30	*NtSetInformationFile*
31	*GetTempPathW*

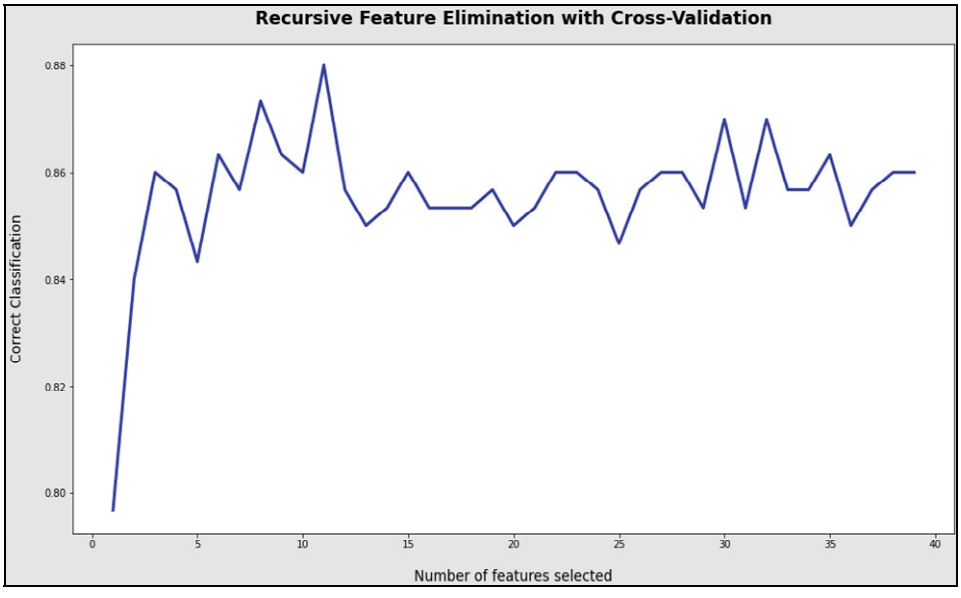
Note: The features in italic are the set of features that are common in malware and ransomware as malware.

Table 1 Features of malware without feature selection (continued)

32	*GetFileSize*
33	*CreateProcessInternalW*
34	*SetFileTime*
35	InternetCrackUrlA
36	*OpenServiceA*
37	InternetConnectA
38	HttpSendRequestA
39	legitimate
40	NtDuplicateObject

Note: The features in italic are the set of features that are common in malware and ransomware as malware.

Figure 2 RFECV (for malware) (see online version for colours)



The Gini impurity (https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html) for splitting the tree and measuring the accuracy is given by

$$IG(p) = 1 - \sum_{i=1}^{j} pi^2$$

where the j represents the number of classes in the label, and pi is the fraction of items labelled with class i in the set.

It was observed that the highest value of correct classification was recorded for around 11 features. To select the 11 features from a list of 40 we applied for each feature, the normalised total reduction using the mathematical value called the Gini importance of the feature.

Using an extra tree classifier, the best 11 features were selected. After obtaining the best features machine learning models were developed for the classification of samples as malware/benign. Table 2 shows the list of 11 features that were selected for classification of samples as benign or malware.

Figure 3 Result of extra tree feature selection (for malware) (see online version for colours)

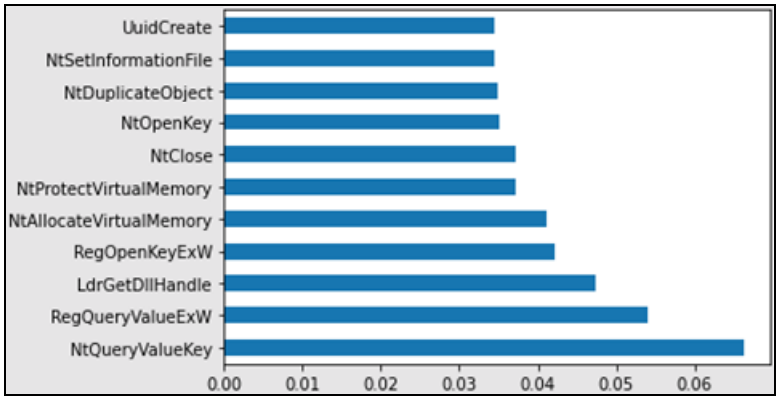


Table 2 List of 11 features of malware after feature selection

RegCloseKey
NtCreateFile
NtAllocateVirtualMemory
UuidCreate
NtOpenKey
LdrGetDllHandle
NtDuplicateObject
NtProtectVirtualMemory
RegOpenKeyExW
RegQueryValueExW
NtQueryValueKey

Note: Features in italic are common to ransomware features after selecting 15 ransomware features.

4.3.2 Results and analysis for malware

In the feature selection phase we select the relevant features of malware for analysis. The machine learning classifier phase involved the training of several machine learning algorithms that helped to identify the optimum detection model. According to the literature study done, we found machine learning classifiers such as LR, RF, K-nearest neighbour (KNN), support vector machine (SVM), and XGBoost, and we have used these for our analysis. Next, we present the mathematical (https://en.wikipedia.org/wiki/Decision_tree_learning#Gini_impurity) models that we have used for the evaluation of the models for implementing the malware and ransomware detection.

4.3.2.1 Logistic regression

LR gives the probability of *Y* which is the dependent variable which is binary variable the function for logistic is used to model the relationship between the response and independent variables. For instance, the probability of *Y* belonging to class 1 or 0 can be written as follows:

$$\Pr\left(Y = 1 \text{ or } 0 \middle| X\right) = p(X) = \frac{e\beta 0 + \beta 1X1 + \ldots + \beta pXp}{1 + e\beta 0 + \beta 1X1 + \ldots + \beta pXp}$$

Since $p(X)$ provides the probability that the response variable having value 1/0, a threshold value should be defined to classify into two or more categories. Default value for threshold is 0.5, and if an instance tuple has value less than 0.5 is classified as 0 and greater than 0.5 is classified as 1.

4.3.2.2 Support vector machines

SVM's main purpose is to separate classes for the data in the best possible way. Support vectors which are called observations are used to find out the decision boundaries. A kernel is used by SVM that integrate the nonlinearity of the boundaries which maximise the margins which is the distance between the observations and the boundaries. The equation for the kernel is given as:

$$K\left(xi, xi'\right) = \left(1 + \sum\nolimits_{j=1}^{p} xijxi''j\right)^{d}$$

d is a degree of the polynomial. According to Laborda and Ryoo (2021) and Shin et al. (2005), radial kernel is the most widely used kernel and is defined mathematically as:

$$K\left(xi, xi'\right) = \exp\left(\left(-\gamma \sum\nolimits_{j=1}^{p} xijxi''j\right)^{2}\right)$$

where γ is a positive constant. The SVM can be combined with a nonlinear kernel and SV classifier which has a form as

$$f(x) = \sum_{i \in S} \beta 0 + \alpha i K(x, xi)$$

where *S* is the collection of indices of support vectors, αi is non-zero if a training observation is a support vector.

4.3.2.3 K-nearest neighbours

KNN classifier acts on the assumption that 'similar inputs have similar outputs'. If given a positive integer *K* and a test observation $x0$, the classifier identifies *K* points which are closest to $x0$ in the training tuples represented by $N0$. After this it estimates the conditional probability for the class *j* which are proportion of points in $N0$ which has response value equal to *j*. The equation is as follows

$$\Pr\left(Y = j \middle| X = x0\right) = 1/K + \sum_{i \in N0} I(yi = j)$$

where $I(yi = j)$ is a variable corresponding to 1 if $yi = j$ and 0 if $yi \neq j$. KNN then classifies the test observation $x0$ to the highest class probability.

Our main aim was to find the most suitable machine learning algorithm to increase the accuracy in identifying the malware correctly. All these algorithms have different techniques of prediction. But, in the end, we need to find the effectiveness of an algorithm. To find the most suitable algorithm for our problem, a few model evaluation techniques have been listed below which will help us find the accuracy of our model. Table 3 shows the results of our implementation using different classifiers to accurately predict the behaviour of malware. The current system predicts if the malware present is benign or malicious. Among the classifiers listed below the XGBoost has a very good K-fold cross validation accuracy% value.

Table 3 Results of different classifiers for accuracy score%, ROC value and K-fold cross value accuracy%

Classifier	*Accuracy score%*	*ROC value*	*K-fold cross val. accuracy%*
KNN	83.33	0.804	85.00
RF	86.67	0.871	89.58
XGBoost	81.667	0.802	90.00
SVM	78.33	0.375	83.75
Decision tree	83.33	0.782	87.5
Logistic regression	78.33	0.678	83.75

4.3.3 Feature selection for ransomware

Using the same method in Subsection 4.3.1 for collection of dataset for any type of ransomware was used for creating a dataset for ransomware also. Table 4 shows the list of 45 features of ransomware. We have 45 features (API calls) in the CSV file of the dataset that was created using the parser.

According to Bhagwat and Patil (2021) we used extra tree and RFECV for the selection of features from the list of 45 features. In this paper, we are detailing the implementation of the dataset that we have created consisting of some attributes whose contribution to the detection of ransomware is negligible and would only prove to hamper the training of the machine learning models. We have used various feature selection methods: variance, ANOVA, chi^2, Pearson's correlation, extra tree, and RFECV. The reason to consider all these feature selection algorithms is to have the best behavioural attributes for increasing detection accuracy. Among these extra tree and RFECV are wrapper feature selection methods. Following is the list of features (API calls) recorded in the JSON file for the ransomware exe files without applying the feature selection methods. The result graphs for the feature score/ranking are shown in Figures 4 to 8.

From Table 4, we found that the features (API calls) that were mostly file related, registry related and process-related. Table 5 shows the details for each category.

Bhagwat and Patil (2021) previous work done for feature selection we have used extra tree and RFECV to find the best 15 features. In our research further and extension to the previous work, we have applied the above-mentioned three feature selection methods in addition to the previous two methods. Table 6 shows the 15 features which were selected by analysing the ranking of these features using these five selection methods. We found the count of the features that appeared in the top 17 to 20 in feature

ranking by these different methods. From Table 6 we found that the features (API calls) were mostly file related, registry related, and process-related. Table 7 shows the details for each category.

Table 4 Features for ransomware detection (45 features)

RegOpenKeyExW
RegOpenKeyExA
RegQueryValueExA
RegQueryValueExW
RegCloseKey
NTOpenKey
NTQueryValueKey
RegCreateKeyExA
RegSetValueExA
NTEnumerateKey
NTProtectVirtualMemory
NTAllocateVirtualMemory
NTFreeVirtualMemory
NTCreateThreadEX
NTResumeThread
ReadProcessMemory
CreateThread
NTOpenThread
NTCreateSection
NTMapViewof Section
CreateProcessInternalW
NTUnmapViewOfSection
NTWriteVirtualMemory
NTGetContextThread
NTSetContextThread
NTTerminateProcess
OpenServiceManagerA
OpenServiceA
GetVolumeNameForVolumeMountpointW
GetTempPathW
NTOpenFile
NTCreateFile
NTSetInformationFile
NTWriteFile
NTQueryInfoFile

Note: The features in italic are the set of features that are common in malware and ransomware as malware.

Table 4 Features for ransomware detection (45 features) (continued)

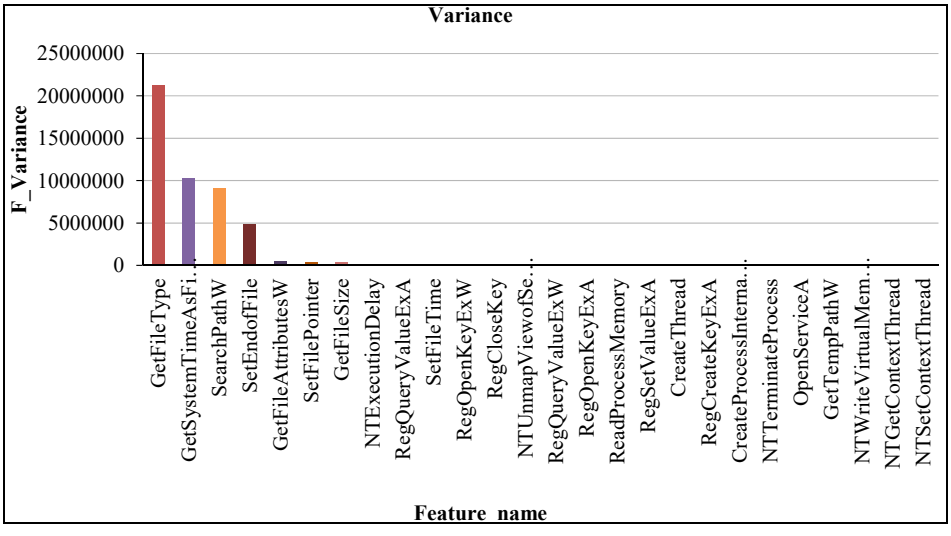
NTReadFile
SearchPathW
GetFileAttributesW
GetFileType
GetFileSize
SetFilePointer
SetEndofFile
SetFileTime
GetSystemTimeAsFileTime
NTExecutionDelay

Note: The features in italic are the set of features that are common in malware and ransomware as malware.

Table 5 Findings from the features list of 45 features (ransomware)

Feature category	*No. of features in that category*
File related features	10
Registry related features	7
Process related features (open to terminate process)	28

Figure 4 Feature score using variance (top F_variance value for features) (see online version for colours)



From Table 1 of 40 features for malware and Table 4 of 45 features of ransomware 23 features are common. Also, we have compared the two methods for RFECV and extra tree for feature selection for both malware and ransomware. The reason behind comparing RFECV and extra tree for feature selection is that both are wrapper methods for feature selection.

Figure 5 Feature score using RFECV (top F_score value for features) (see online version for colours)

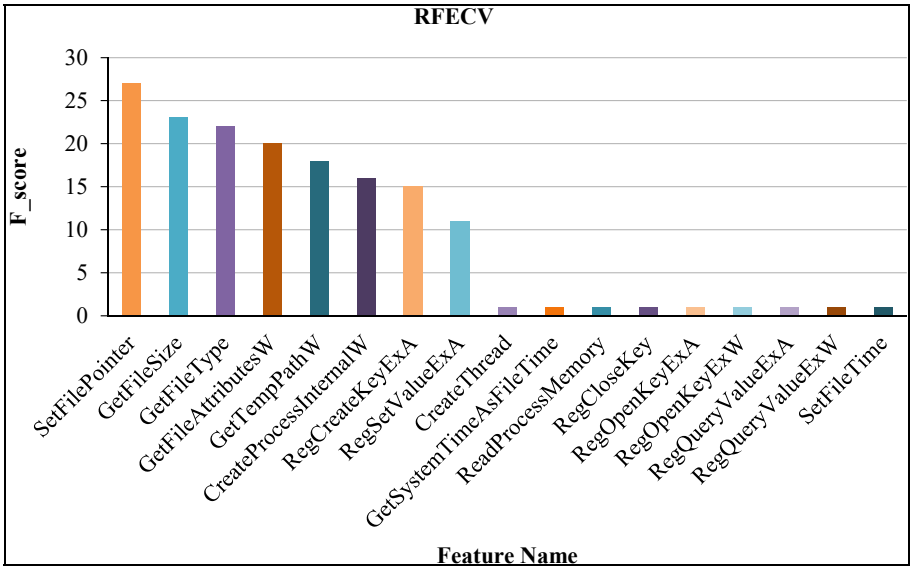
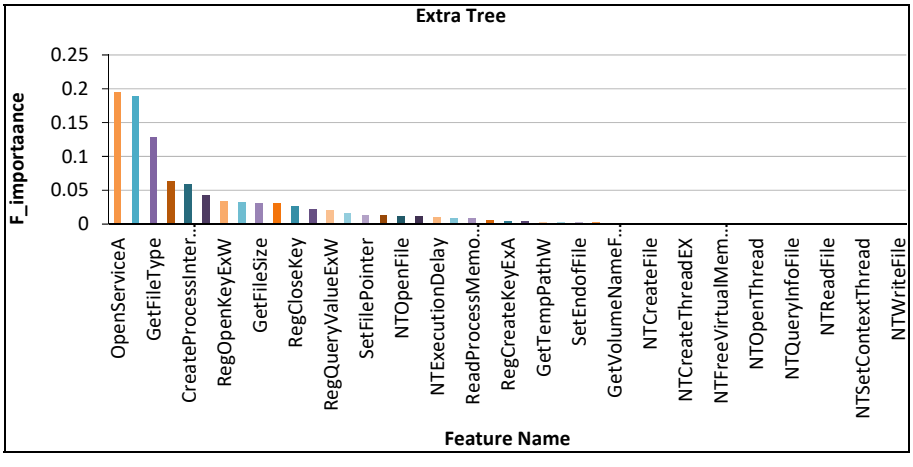


Figure 6 Feature importance using extra tree (top F_importance value for features) (see online version for colours)



There are five common features when the RFECV method was applied to the dataset of malware and ransomware. This means that these are common behavioural characteristics for both, but for the detection of ransomware, some other features are to be considered for accurate detection. Similarly, only four features are common when the extra tree feature selection is applied.

Figure 7 Feature score using chi^2 (top F_score value for features) (see online version for colours)

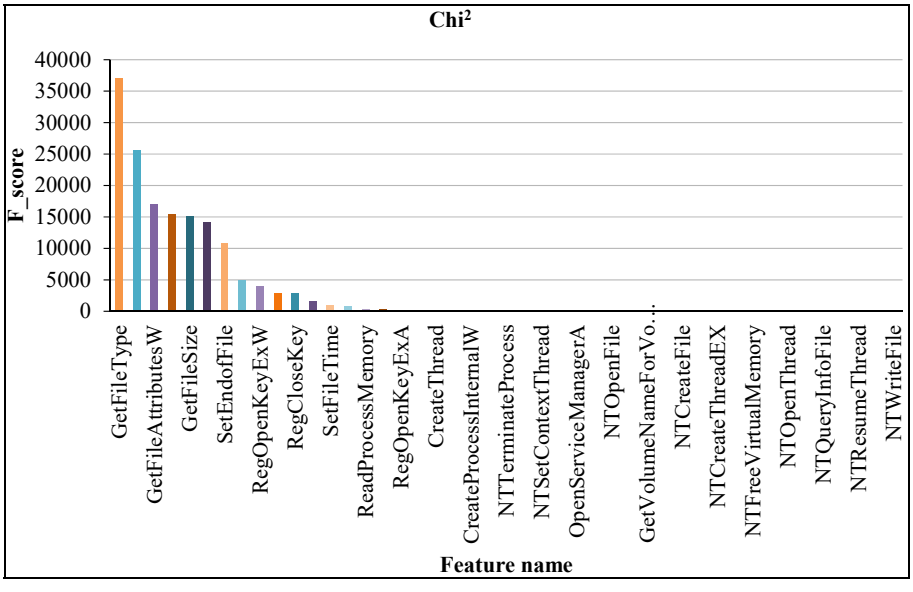
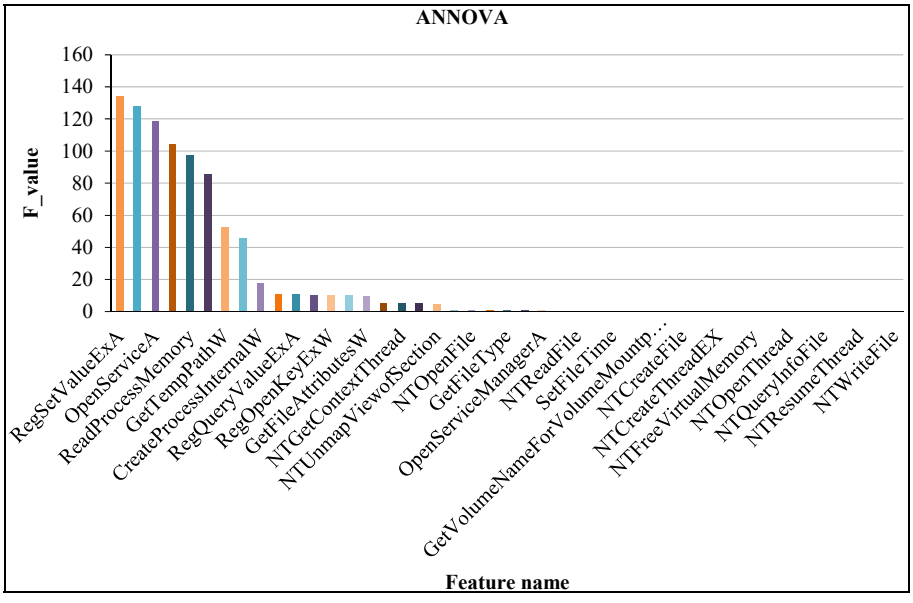


Figure 8 Feature score using ANOVA (top F_value for features) (see online version for colours)



As it is very important to identify the proper set of features for the exact detection of ransomware samples we analysed the features that were ranked by different feature selection methods. Table 8 shows the count of the ransomware feature that was ranked in the top 17 by various feature selection methods Filter and wrapper methods. The graphs for each method with feature importance or feature score for all the features are shown in Figures 4 to 7.

Table 6 Ransomware features selected by feature selection algorithm

1	RegOpenKeyExW
2	RegOpenKeyExA
3	RegQueryValueExA
4	RegQueryValueExW
5	RegCloseKey
6	RegCreateKeyExA
7	RegSetValueExA
8	ReadProcessMemory
9	CreateThread
10	CreateProcessInternalW
11	OpenServiceA
12	GetTempPathW
13	GetFileSize
14	SetFilePointer
15	SetFileTime

Table 7 Findings from the features list of 15 features (Table 6)

Feature category	*No. of features in that category*
File related features	4
Registry related features	7
Process related features (open to terminate process)	3

4.3.4 Result and analysis of ransomware

From the section feature selection for ransomware as malware we found the best 15 features using the above feature selection strategy. After having these best features, we applied the machine learning algorithms ad created the models for the verification and detection of ransomware. The different ML algorithms that were used for classification were LR, SVM, KNN classifier, and RF. We have evaluated our methods on true positive rate, true negative rate, and accuracy. Table 9 shows the result (Bhagwat and Patil, 2021) for the different models on the evaluation parameters.

As seen from the table KNN performed very well with an accuracy of 96.22% for the set of features we had selected using the feature selection method as mentioned in the feature selection for ransomware section.

4.3.4.1 Comparison with previous work

Daku et al. (2018) (150 ransomware samples) initially found 27 features. They applied iterative feature selection methods for selecting the final 12 features for higher classification. They used different classification algorithms such as J48 DT, Naïve Bayes, and KNN. They got an accuracy of 78% for J48, 77.33 for KNN, and 61.33 for Naïve Bayes.

Table 8 Count of the ransomware feature that was ranked in the top 17 by various feature selection methods

Ransomware feature name	*Count of ransomware features in top 17 using diff. feature selection methods*
GetFileAttributesW	5
RegQueryValueExA	5
RegCloseKey	5
RegOpenKeyExW	5
RegQueryValueExW	5
RegOpenKeyExA	5
SetFilePointer	5
GetFileSize	5
GetFileType	4
ReadProcessMemory	4
RegSetValueExA	4
SetFileTime	4
GetSystemTimeAsFileTime	3
SearchPathW	3
CreateProcessInternalW	3
CreateThread	3
NTExecutionDelay	2
GetTempPathW	2
RegCreateKeyExA	2
OpenServiceA	2
NTGetContextThread	2
SetEndofFile	1
NTUnmapViewofSection	1
NTTerminateProcess	1

Table 9 True positive rate, true negative rate, and accuracy

Sr. no.	*ML algorithm*	*TP rate*	*TN rate*	*Accuracy*
1	Logistic regression	89.06%	100%	94%
2	Support vector machine (SVM)	92.15%	100%	92.45%
3	*K-nearest neighbours (KNN) classifier*	*100%*	*60%*	*96.22%*
4	Random forest	90.47%	0	90.47%

Source: Bhagwat and Patil (2021)

To compare further Hasan and Rahman (2017) had done the malware analysis by capturing the network traffic attributes/features such as source/destination IP address, source/destination IP address port no. and a number of packets/size. They applied machine learning algorithms. The accuracy achieved by the J48 classifier was 97.10% which was the highest. It was followed by LMT with 96.80%, next RF with 96.10%, and KNN with 95.30%.

Hasan and Rahman (2017) implemented RansHunt which applied a hybrid analysis of the ransomware samples. They applied the classification algorithms and achieved 96.1 accuracies for RansHunt, 95.2 for DT (RF), and 94.2 for Naïve Bayes.

Vinod and Viswalakshmi (2018) achieved an accuracy of 99.8 to 100% but they have done the analysis for Android system API calls.

Alhawi et al. (2018) used network traffic features and found a TPR of 97.1% using the DT (J48) classifier.

Takeuchi et al. (2018) did classification using the API call sequencing using the q-gram technique. They applied the SVM machine learning algorithm and achieved an accuracy of 97.48%. They also found the missing rate as 1.64%.

Ahmed et al. (2020) had done feature selection by enhanced maximum relevance and minimum redundancy method which selects features faster as compared to mRmR. Using this EmRmR method they have selected the relevant feature set and applied five machine-learning classifiers SVM, LR, RF, KNN, and DT. They found that DT and SVM achieved better performance for recall and precision.

Pektaş and Acarman (2017) used the n-gram method for API-call sequences on the registry, file system, and network. They have achieved an accuracy of 94% and 92.5%, for the online classification CW algorithm for training and testing of samples.

Hwang et al. (2020) used Markov model as compared to the accuracy that we had for the different classification algorithms we have the best K-fold cross val. accuracy% of 90 % for XGBoost for malware detection and 96.22% for KNN for ransomware detection.

If compared to the accuracy that we had for the different classification algorithms we have the best K-fold cross val. accuracy of 90 % for XGBoost for malware detection and 96.22% for KNN for ransomware detection.

5 Conclusions

In this paper we presented the detailed result and analysis of the best selection of features using various feature selection methods for malware and its subset ransomware. It is very important to have the best feature set for the detection accuracy of machine learning algorithms. Behavioural analysis was done as we wanted to do the detection of malware/ ransomware using a dynamic approach. We have also presented the detailed experimental setup for the creation of our dataset. We created our own dataset with API calls as the feature set. API calls were selected as features because they give the runtime behaviour of the malicious files. These features can be used for the accurate detection of malware/ ransomware. The set of features that we had selected for malware/ransomware gave the best accuracy as compared to the previous work. Using the dynamic detection technique, we found the best set of features for the behavioural characteristics and obtained the accuracy of 90 % for XGBoost for malware detection and 96.22% for KNN for ransomware detection.

In the future, we will like to consider the parameters that are in the API calls as the features along with API calls.

Acknowledgements

We would like to thank Aditya Vihite, Anagha Patil, Mrunmai More, Shradha Retharekar, Vinayak Bajpeyi, Diksha Singh Nitesh Chandra and Aniruddha Kapgate for helping us with the experimental setup.

References

Abbasi, M.S., Al-Sahaf, H. and Welch, I. (2020) 'Particle swarm optimization: a wrapper-based feature selection method for ransomware detection and classification', in Castillo, P.A., Jiménez Laredo, J.L. and Fernández de Vega, F. (Eds.): *Applications of Evolutionary Computation, EvoApplications 2020, Lecture Notes in Computer Science*, Springer, Cham, Vol. 12104 [online] https://doi.org/10.1007/978-3-030-43722-0_12.

Ahmed, Y.A., Koçer, B., Huda, S., Al-Rimy, B.A.S. and Hassan, M.M. (2020) 'A system call refinement-based enhanced minimum redundancy maximum relevance method for ransomware early detection', *Journal of Network and Computer Applications*, Vol. 167, p.102753, ISSN: 1084-8045 [online] https://doi.org/10.1016/j.jnca.2020.102753.

Alazab, M., Venkatraman, S., Watters, P. and Alazab, M. (2011) 'Zero-day malware detection based on supervised learning algorithms of API call signatures', in *Proceedings of the Ninth Australasian Data Mining Conference (AusDM '11)*, Australian Computer Society, Inc., AUS, Vol. 121, pp.171–182.

Alhawi, O.M.K., Baldwin, J. and Dehghantanha, A. (2018) 'Leveraging machine learning techniques for Windows ransomware network traffic detection', in Dehghantanha, A., Conti, M. and Dargahi, T. (Eds.): *Cyber Threat Intelligence, Advances in Information Security*, Springer, Cham, Vol. 70 [online] https://doi.org/10.1007/978-3-319-73951-9_5.

Al-Rimy, B.A.S., Maarof, M.A. and Shaid, S.Z.M. (2018) 'A 0-day aware crypto-ransomware early behavioral detection framework', in Saeed, F., Gazem, N., Patnaik, S., Saed Balaid, A. and Mohammed, F. (Eds.): *Recent Trends in Information and Communication Technology, IRICT 2017, Lecture Notes on Data Engineering and Communications Technologies*, Springer, Cham, Vol. 5 [online] https://doi.org/10.1007/978-3-319-59427-9_78.

Bhagwat, L.B. and Patil, B.M. (2020) 'Detection of ransomware attack: a review', in Bhalla, S., Kwan, P., Bedekar, M., Phalnikar, R. and Sirsikar, S. (Eds.): *Proceeding of International Conference on Computational Science and Applications. Algorithms for Intelligent Systems*, Springer, Singapore [online] https://doi.org/10.1007/978-981-15-0790-8_2.

Bhagwat, L.B. and Patil, B.M. (2021) 'Detection of ransomware on Windows system using machine learning technique: experimental results', in Garg, D., Wong, K., Sarangapani, J. and Gupta, S.K. (Eds.): *Advanced Computing, IACC 2020, Communications in Computer and Information Science*, Springer, Singapore, Vol. 1367 [online] https://doi.org/10.1007/978-981-16-0401-0_32.

Cuckoo Foundation (2014–2019) *Cuckoo Sandbox*, Stichting Cuckoo Foundation.

Daku, H., Zavarsky, P. and Malik, Y. (2018) 'Behavioral-based classification and identification of ransomware variants using machine learning', *2018 17th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/12th IEEE International Conference on Big Data Science and Engineering (TrustCom/BigDataSE)*, pp.1560–1564, DOI: 10.1109/TrustCom/BigDataSE.2018.00224.

Fukushima, Y., Sakai, A., Hori, Y. and Sakurai, K. (2010) 'A behavior-based malware detection scheme for avoiding false positive', *2010 6th IEEE Workshop on Secure Network Protocols*, pp.79–84, DOI: 10.1109/NPSEC.2010.5634444.

Gržinić, T. and González, E.B. (2022) 'Methods for automatic malware analysis and classification: a survey', *International Journal of Information and Computer Security*, 24 February, Vol. 17, Nos. 1–2, pp.179–203 [online] https://doi.org/10.1504/IJICS.2022.121297.

Hampton, N., Baig, Z. and Zeadally, S. (2018) 'Ransomware behavioral analysis on windows platforms', *Journal of Information Security and Applications*, Vol. 40, pp.44–51, ISSN: 2214-2126 [online] https://doi.org/10.1016/j.jisa.2018.02.008.

Hasan, M.M. and Rahman, M.M. (2017) 'RansHunt: a support vector machines based ransomware analysis framework with integrated feature set', *2017 20th International Conference of Computer and Information Technology (ICCIT)*, *IEEE Conference*.

Hwang, J., Kim, J. and Lee, S. (2020) 'Two-stage ransomware detection using dynamic analysis and machine learning techniques', *Wireless Pers. Commun.*, Vol. 112, pp.2597–2609 [online] https://doi.org/10.1007/s11277-020-07166-9.

Idika, N.C. and Mathur, A.P. (2017) 'A survey of malware detection techniques', Purdue University.

Ki, Y., Kim, E. and Kim, H.K. (2015) 'A novel approach to detect malware based on API call sequence analysis', *International Journal of Distributed Sensor Networks*, June, DOI: 10.1155/2015/659101.

Laborda, J. and Ryoo, S. (2021) 'Feature selection in a credit scoring model', *Mathematics*, Vol. 9, p.746 [online] https://doi.org/10.3390/math9070746.

Liska, A. and Gallo, T. (2016) *Ransomware-Defending Against Digital Extortion*, O'Reilly Media.

Maiorca, D., Mercaldo, F., Giacinto, G., Visaggio, C.A. and Martinelli, F. (2017) 'R-PackDroid: API package-based characterization and detection of mobile ransomware', in *Proceedings of the Symposium on Applied Computing (SAC '17)*, Association for Computing Machinery, New York, NY, USA, pp.1718–1723, https://doi.org/10.1145/3019612.30197.

Pektaş, A. and Acarman, T. (2017) 'Classification of malware families based on runtime behaviors', *Journal of Information Security and Applications*, Vol. 37, pp.91–100, ISSN: 2214-2126 [online] https://doi.org/10.1016/j.jisa.2017.10.005.

Pirscoveanu, R.S., Hansen, S.S., Larsen, T.M.T., Stevanovic, M., Pedersen, J.M. and Czech, A. (2015) 'Analysis of malware behavior: type classification using machine learning', *2015 International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA)*, pp.1–7, DOI: 10.1109/CyberSA.2015.7166115.

Portable Freeware Collection [online] https://www.portablefreeware.com.

Ramírez-Gallego, S., Lastra, I., Martínez-Rego, D., Bolón-Canedo, V., Benítez, J.M., Herrera, F. and Alonso-Betanzos, A. (2016) [online] https://doi.org/10.1002/int.21833 (accessed 9 July 2016).

Salehi, M.G. and Sami, A. (2012) 'A miner for malware detection based on API function calls and their arguments', *The 16th CSI International Symposium on Artificial Intelligence and Signal Processing (AISP 2012)*, pp.563–568, DOI: 10.1109/AISP.2012.6313810.

Sgandurra, D., Muñoz-González, L., Mohsen, R. and Lupu, E.C. (2016) 'Automated dynamic analysis of ransomware: benefits, limitations and use for detection', *Computing Research Repository (CoRR)*, Cornell University, Ithaca, NY, USA, September, abs/1609.03020, arXiv.org E-print Archive.

Shaukat, S.K. and Ribeiro, V.J. (2018) 'RansomWall: a layered defense system against cryptographic ransomware attacks using machine learning', *2018 10th International Conference on Communication Systems & Networks (COMSNETS)*, pp.356–363, DOI: 10.1109/COMSNETS.2018.8328219.

Shin, K-S., Lee, T.S. and Kim, H-j. (2005) 'An application of support vector machines in bankruptcy prediction model', *Expert Systems with Applications*, Vol. 28, No. 1, pp.127–135, ISSN: 0957-4174 [online] https://doi.org/10.1016/j.eswa.2004.08.009.

Skaletsky, A. et al. (2010) 'Dynamic program analysis of Microsoft Windows applications', *2010 IEEE International Symposium on Performance Analysis of Systems & Software (ISPASS)*, pp.2–12, DOI: 10.1109/ISPASS.2010.5452079.

Software Download [online] https://en.softonic.com/downloads/software-download.

Takeuchi, Y., Sakai, K. and Fukumoto, S. (2018) 'Detecting ransomware using support vector machines', in *Proceedings of the 47th International Conference on Parallel Processing Companion (ICPP '18)*, Association for Computing Machinery, New York, NY, USA, Article 1, pp.1–6 [online] https://doi.org/10.1145/3229710.3229726.

Uppal, D. (2014) 'Basic survey on malware analysis, tools and techniques', *International Journal of Computer Science & Applications (IJCSA)*, February, Vol. 4, No. 1, pp.103–112.

Vinod, P. and Viswalakshmi, P. (2018) 'Empirical evaluation of a system call-based Android malware detector', *Arab J Sci. Eng*., Vol. 43, pp.6751–6770 [online] https://doi.org/10.1007/s13369-017-2828-0.

VirusShare, *Malware Sharing Platform* [online] http://www.virusshare.com (accessed 2022).

VirusTotal, *Online Multiple AV Scan Service* [online] http://www.virustotal.com (accessed 2022).

VX Vault [online] http://vxvault.net (accessed 2022).

Websites

https://en.wikipedia.org/wiki/Decision_tree_learning#Gini_impurity.

https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html.

https://www.simplilearn.com/tutorials/machine-learning-tutorial/feature-selection-in-machine-learning.