Journal of Engineering Science and Technology
Vol. 14, No. 1 (2019) 291 - 304

SCRUTINIZED SYSTEM CALLS INFORMATION USING J48 AND Jrip FOR MALWARE BEHAVIOUR DETECTION

FAIZAL M. A.*, WARUSIA YASSIN, NUR HIDAYAH M. S.

Faculty of Information and Communications Technology, Universiti Teknikal Malaysia Melaka (UTeM), Hang Tuah Jaya, 76100 Durian Tunggal, Melaka, Malaysia
*Corresponding Author: faizalabdollah@utem.edu.my

Abstract

Malware is considered as one of most emerging threats due to Cybercriminals work diligently to make most of the part of the users' network of computers as their target. A number of researchers keep on proposing the various alternative framework consisting detection methods day by days in combating activities such as single classification and the rule-based approach. However, such detection method still lacks in differentiate the malware behaviours and cause the rate of falsely identified rate, i.e., false positive and false negative increased. Therefore, integrated machine learning techniques comprise J48 and Jrip are proposed as a solution to distinguish malware behaviour more accurately. This integrated classifier algorithm applied to analyse, classify and generate rules of the pattern and program behaviour of system call information in which, the legal and illegal behaviours could identify. The result showed that the integrated classifier between J48 and Jrip significantly improved the detection rate as compared to the single classifier.

Keywords: Classifier, J48 and Jrip, Machine learning, Malware detection, System call.

1. Introduction

According to Grini [1], malware is a program that has a malicious intention, whereas Egele et al. [2] has defined it as a generic term that encompasses viruses, Trojans, spyware and other intrusive codes. Malware refers to viruses, worms, ransomware, Trojan horses, key-loggers, root-kits, spyware, adware and malicious programs [3]. As the networks develop massively in size and complexity, many researchers propose techniques for classification and detection of malware especially in system call as it is the most important event of being traced for recognizing malware behaviour. System calls are defined as the fundamental interface between an application and the operating system kernel [4]. In the operating system, all the tasks and services required by the malware to execute malicious action through system calls. According to Kolosnjaji et al. [5], as a result, any execute malware activity can be monitored by observing the patterns in the system calls include opening a file, running a thread, writing to the registry or opening a network connection. Thus, it is important to track the activity of the system call through malware execution in order to characterize the malware behaviour.

The various research focused on malware detection based on machine learning and classification method stated that the input data for statistical approach can use the features of the behaviour of malware such as system call. Another researcher also tries to use advanced methods of machine learning in the probabilistic model in detecting malicious system call sequences [6].

Even though various classifier, e.g., Naive Bayes, Decision Tree, Support Vector Machine, Neural Network and so forth have been proposed, to derived information form these classifier required a deep knowledge and understanding within the subject matter such as altering the parameter [7] and adding new features that might influence the output. Thus, an algorithm that will facilitate the user, comprehensive and less complex is most preferred. This motivate researcher to come with the solution by introducing rule-based classification models in which, the rules produced by analyses the logic of the classifier decision and eliminates the uncertain consequences represented by classifiers [7].

However, efficiency and accuracy are two essential aspects of performing malware classification and detection in the system call. In order to classify the behaviour of malware the frequency of system calls made by each malware activity, a framework consisting data collection, feature extraction and integrated learning approach are considered. Therefore, the J48 and Jrip classifier with association rules mining will be used in this paper to select significant features of a dataset. The rules are built by using this two classifier and then the rules have been used to assign ranks of the feature. To this effect, we have achieved an acceptable level of accuracy in classifying the activity as malicious or benign using the J48 and Jrip algorithm.

The remainder of this paper is presented as follows: Section 2 discusses the related study and section 3 presents the methodology used for this paper. Section 4 presents the discussion of the results. Section 5 concludes the paper and presents future work directions.

2. Literature Review

Recently, many malware researchers have focused on data mining to distinguish unknown malware. Data mining is known as the process of discovering patterns in data [8]. Meanwhile, machine learning algorithms have been used broadly for different data mining problems. Many of the researchers have proposed the method of malware classification and detection by using several of the classifiers in order to obtain high accuracy. Yet, selecting an appropriate classification algorithm is very essential since it influences the detection accuracy and performance. There is some previous work for classification algorithm that has been done for several researchers as shown in Table 1.

From the Table 1, Pfoh et al. [9] proposed the string subsequence kernel to detect malware attacks [10]. The combination of SVM and string kernel attains great accuracy. Detecting malicious traffic at network level several algorithms have been used to categorize malware communications that are produced by dynamic malware analysis. The significant feature has been extracted from generated traffic log file in order to distinguish malicious packet.

Table 1. Previous work (classification algorithm).

Author	Purpose	Technique/ classifier	Result
Pfoh et al. [9]	String kernel has been used to construct the sequential data inherent in a system call trace	Support Vector Machine (SVM).	Obtained 98.88% accuracy rate for system call.
Boukhtouta et al. [11]	Detect malicious traffic at the network level.	J48, Naıve Bayes, K-Means and Support Vector Machine (SVM), Boosted J48.	J48 and Boosted J48 outperform other classifier with 99% accuracy.
Ahmad et al. [12]	Tokenization approach based on system call has been used for mobile attacks.	SVM, Random Forest, Naive Bayes, and J48.	Naïve Bayes outperform other classifier with 99.86% accuracy.
Chaba et al. [13]	Detecting the behavior of the malicious running on a host system.	Naive Bayes, Random Forest and Stochastic Gradient Descent (SGD).	SGD achieve high rate of accuracy with 95.5%.
Rathore, M.M. et al. [14]	Hadoop has been used in real-time IDS for ultra-high speed big data scenario.	J48, REPTree, Random Forest, Conjunctive rule, SVM, and Naive Bayes.	J48 and REPTree are the best classifiers in terms of 99.9% accuracy.

In addition, based on research by Boukhtouta et al. [11], a new classification model produces unique behaviour patterns based on system call sequence pattern. The result shows that the new model produces the higher value of accuracy. Besides, the author detecting the behaviour of the malicious that running on a host system based on system call sequences [12]. The dataset using in this experiment are based on generated from system call log and the quality of dataset has been improved using filtering algorithm.

Furthermore, Chaba et al. [13] proposed a system using Hadoop implementation to detect real-time IDS for ultra-high speed big data environment. The four layer of IDS architecture such as capturing layer, filtration and load balancing layer, processing and decision-making has been used in his research. Lastly, based on literature review, we can conclude that J48 decision tree classifier gives the highest detection accuracy rate with 99%. Therefore, this classifier will be used to examine the occurrence of system calls and unknown registry made by each malware activity.

On another hand, the rule induction approaches discover the rule on the basis of greedy and per class label, whereas these classifier algorithm build the trained model from training set, which comprises the part of available class labels. Thus, various researchers proposed many of rule-induction and combined algorithm in decades. Table 2 shows previous work for rule-induction and combined algorithm.

Based on Table 2, Rathore et al. [14] proposed a combined algorithm based on machine learning approach between ensemble learning and preprocessing techniques called VETO. Using VETO, the malware and benign file have been used to extract the sequences of operational codes. Later, a series of learning algorithm used to evaluate these datasets while the prediction output generates based on veto voting using ensemble learning. Apart of other available voting approach, the VETO proposed to detect malware activity more accurately.

Table 2. Previous work (rule-induction and combined algorithm).

Author	Purpose	Technique /classifier	Result
Shahzad and Lavesson, [15]	The author has proposed a malware identification scheme based on machine learning approach called VETO consisting an ensemble learning and preprocessing techniques. The prediction is conducted through a set of combined algorithm while the result is finalized based on the veto voting classifier.	VETO (Combined Algorithm)	The Jrip achieved 35 false positive and Veto at 111 while false negative at 41 and at 8, respectively.
Qabajeh et al. [16]	The author proposed a new dynamic rule induction (DRI) techniques to overcome the later PRISM issues on pruning and rule-sharing items. In order to generate perfect rules, the frequency values of an attribute are dynamically altered whenever rules generated in DRI.	DRI (Rule Induction Algorithm)	DRI generate the most perfect rules as compare to PRISM (rule induction technique).
Zakeri et al. [17]	The author proposed a method, which is significantly focused on crucial heuristic features as well as fuzzy classifier namely FURIA. Moreover a pre-processing approach also considered to increase the prediction accuracy via avoiding the suspicious exceptions in legitimate files.	Information Gain and Fuzzy Rule Induction Classifier (Combined algorithm)	FURIA, Jrip and J48 has recorded similar accuracy rate at 99% and for false positive rate FURIA achieved better result at 0.008% while RIPPER and J48 at 0.02%.

Kshirsagar and Joshi, [18]	The author proposed a framework comprises a set of rules for intrusion detection in live traffic. First, the algorithm applied to construct behavior pattern of system audit data and extract the important features using improved apriori algorithm. Later, a set of rules generated based on the definition of these features using Jrip.	Apriori Algorithm and Jrip (Combine algorithm)	Not available.
Bhaya and Ali [19]	An author has reviewed a series of current data mining algorithm, which applied for unknown and known malware identification.	Method reviewed are RIPPER, Naive Bayes, Neural Network, Decision Tree, Support Vector Machine, N-Gram,	Author has concluded that data mining approach can be applied to detect and classify malware behavior. However, the selection of algorithm must be based on few consideration such as scalable, fast and flexible.

Additionally, Qabajeh et al. [16] suggested the rule induction algorithm approach, which employs two values of the threshold to reduce the search space. In DRI, the occurrence of attribute values, which applied to compose the consequent in-line rule that requires data deletion updated frequently. Hence, on every single time a rule generated, the related attribute occurrence values are dynamically adjusted with the aim to construct the perfect rules.

Moreover, Zakeri et al. [17] explained that they intend to improve the detection of malware files. The author proposed a combination of learning method between Information Gain (IG) for pre-processing stage and Fuzzy Rule Induction for classifier stage, which known as Fuzzy Unordered Rule Induction Algorithm (FURIA). The IG can construct a ratio for every single related feature and estimate its significance in a process to determine either the subject is malicious or not. In addition, the FURIA algorithm has considered as this algorithm adequate to learn fuzzy rules and rule sets as compare to traditional rules.

Zakeri et al. [17] proposed a framework comprises Apriori and Jrip algorithm for live-traffic intrusion detection. Using an improved Apriori algorithm, the relevant features extracted after the behavior pattern of system audit data derived. In another word, this algorithm has applied for generate association rules, which acquired via frequent item sets data. Subsequently, Jrip employed to construct a series of rules based on earlier feature definition. However, Kshirsagar and Joshi [18] has presented a survey based article regarding the ability of Data Mining (DM) approach on malware behavior detection. DM approach usually used in analyzing and disclose concealed knowledge within the data before predict its behaviors. Hence, DM has gain popularity and still being used in various field including cyber security in identifying and classify malware activities.

Briefly, from the aforementioned related works various alternative techniques have been proposed by the researcher in distinguishing malware detection in the system call. Nonetheless, the method still lacks in differentiate the behaviours of malware in the system call and affect the rate of false negatives. Currently, the type

of malware attack has encountered significant changes and seems the behaviour of a normal situation has difficult to be identified. Therefore, this research is focused on detecting the behaviour of malware in the system call. Chaba et al. [13] supported this research in which, recommended a combined algorithm could be effective for certain problem such as detection of malware. Thus, a combined learning, which is J48 and the Jrip algorithm has been proposed in this work.

3. Research Methodology

In this research, the new proposed framework is implemented as shown in Fig. 1. There is three phase in the proposed framework, which is data collection, feature extraction and integrated classifier.

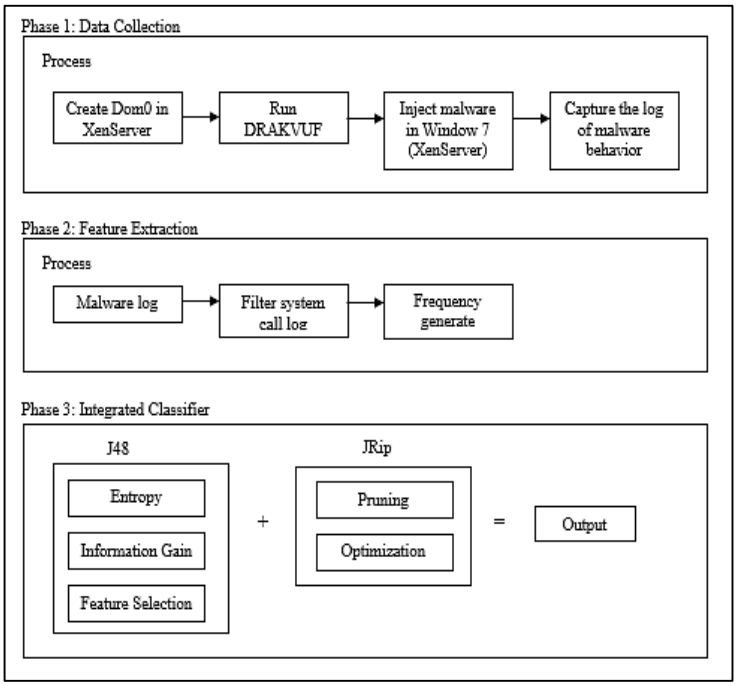

Fig. 1. Proposed framework.

3.1. Data collection

Lengyel et al. [20] commented that, in this phase, the data of system call, which involve malware and benign dataset are collected using DRAKVUF where the frequency rate of data collection is about 4 minutes due to the detection of the malware behavior factor. Firstly, Dom0 will be created in XenServer, which is used in Window 7 as its virtual medium. After running the Windows 7 Operating System, the malware that has been selected will be copied inside the DomU. Before injecting the malware inside the DRAKVUF, there are a few steps that need to be considered: 1) Identify the PID process inside kernel32.dll that can be used as a process to inject the malware. 2) Activate screen log utilities to capture all log produce by DRAKVUF. The next step is injecting the malware using the PID

process that has been chosen from the previous step. The screen log utilities will capture the DRAKVUF log for 4 minute. The DRAKVUF log will be filtered using the grep command to capture only system call log produced by the DRAKVUF. All the log will be save into a file and will be used to the next process to generate the frequency for the system call. The procedure will be repeated for capturing the normal application log except injecting the malware inside the DomU.

3.2. Feature extraction

In feature extraction phase, the system call log from the malware application will be filtered. Then, the data frequency selected from the log will be map to the windows 7 system call, which has 473 attributes. Then, the data frequency selected from the log will generate and convert into csv file. Next, the frequency csv file will generate into excel and the data will be analyses to produce the output result based on a system call.

3.3. Integrated classifier

In integrated classifier phase, J48 and Jrip classifier will be used in this research. J48 classifier is a Java implementation based on the C4.5 algorithm, which uses the technique of decision tree to organize the data classification.

The classifier needs to create a decision tree based on the attribute values of the available training data and classify the attributes when a set of the feature in training data is identified [21]. The classification criterion of the selected attribute is based on the calculation of entropy and information gain as it is the best attributes to separate the data.

Entropy specifies the amount of information that is held where the higher the entropy the more information it contains while information gain emphasis on the significance of the feature or attribute, which help in selecting the best split of the data. Nonetheless, the limitation of the J48 algorithm is it can lead to overfitting, less predictive in any situation and does not work very well on small data training. Thus, J48 decision tree can be divided into three stages namely entropy stage and information gain stage and feature selection stage. The algorithm of the J48 as Fig**.** 2**.**

Meanwhile, Dubitzky et al. [7] proposed the Jrip, which is known as repeated incremental pruning. These algorithm has been selected as it possesses several capabilities include strategy in revising and replacing the generated rules, which could increase the accuracy for detection, able to handle noisy data and overcome overfitting issues as well as suitable for imbalanced class distributions such as for system calls data.

However, the drawbacks of this algorithm are it requires a large amount of memory and generate very long rules, which are very hard to prune. Moreover, this algorithm contains a stage for optimizing a rule set by removing learned rule and re-learn it in different perspective or conditions that are not similar with learned rules. This will lead to increasing the accuracy as the rule dynamically updated. The Jrip, which is considered in this work can be divided into four principal stages namely initialization stage, building a stage that involving growing and pruning steps, optimization stage and deletion stage. The algorithm procedures work as Fig. 3.

START PROCEDURES
//Initialize dataset or training data

//1. Begin building stage
a) Create node as root and label with splitting attribute
b) Add arc to node as root
c) Predicate the split and assign label

//2. Entropy stage
The data has been calculated by using entropy formula to measure the disorder of data. Then, the value of entropy is used to calculated information gain.

$\text{Entropy(p)} = -\sum_j^n = 1 \frac{|p_j|}{|p|} \log \frac{|p_{jl}}{p}$

$\text{Entropy(j|p)} = \frac{|p_j|}{|p|} \log \frac{|p_{jl}}{p}$

//3. Information gain stage
Every attribute is calculated based on the potential information given by a test on the attribute. Then it will test on the attribute. The information gain is calculated based on:

Gain (j,p) = Entropy(j- Entropy(j|p))

//4. Feature selection stage
a) The best attribute is selected

END PROCEDURES

Fig. 2. J48 algorithm.

START PROCEDURES

//Begin initialization stage
Initialize a set of Rule *R*={} for every single Class of *C* of the less and most frequent

//1. Begin building stage
Re-iterate the growing and pruning step until meet one of the below criteria:
a) error rate surpass 50% *or*
b) the description length (DL) of the rule higher than the lowest description length found so far *or*
c) no more discovered instances of *C*

//1.1 Begin growing step
proceed to grow rule via adding condition terms to the rule greedily until the rule is accurately perfect 100%, in which every single potential value of every single attribute have to test and select the highest condition in term of Information Gain value: p(log(p/t)-log(P/T)).

//1.2 Begin pruning step
Every single rule is growingly pruned and the formula 2p/(p+n)-1 considered to measure pruned value.

//2. Optimization stage
The beginning rule set $\{R_i\}$ discovered in this stage. By employing procedures of growing and pruning steps, two variants of every single rule R_i from random data generated. One of the variant generated via empty rule while the later through the original rule which is added with antecedents. The pruning metric applied over here is (TP+TN)/(P+N). Consequently, the lowest DL for entire variant is calculated and only the minimum DL are considered as the final representative of R_i in the ruleset. The building stages will be run again to generate more rules if residual positives still exist after observing $\{R_i\}$ rules.

//2. Deletion stage
Delete the entire rules from ruleset which have maximum DL and added the remaining rules into resultant set.

END PROCEDURES

Fig. 3. Jrip algorithm.

4. Results and Discussion

In this section, the accuracy result will be displayed and documented in brief a detailed analysis to draw the conclusion and the findings. The performance of classification using selected features were measure in the term of accuracy, which is equal to ((TN + TP)/(TP + FP + TN + FN)). Table 3 illustrate the classification table.

where:

- TP⟶ True positive, the number of malware correctly classified

- TN⟶ True negative, the number of benign correctly classified.
- FP⟶ False positive, the number of benign detected as malware.
- FN⟶ False negative, the number of malware detected as benign.

By using the confusion matrix from Table 3, the infer parameters:

- False Positive = FP/(FP + TP)
- False Negative = FN/(FN + TN)
- Detection Attack Rate = TP/(FN + TP)
- Overall Detection Rate = (TN + TP)/(TN + FP + FN + TP)

The outcomes of the system call classification are presented and discussed. The system call data was extracted and has been analysed by using two different classifiers (J48 and Jrip). The integrated of these two classifiers also has been analysed to obtain high detection of malware based on classification table. Table 4 show the result of classification Jrip, J48 and the integrated classifier between Jrip and J48.

Table 4 demonstrates the result of classifier Jrip produces 23.07% false positive as six of the data in the sample has been misclassified as malware, which is 'iexplore_syscall', 'firefox_syscall', 'putty_syscall', 'notepad_syscall2', 'calc_syscall' and 'iexplore_syscall'. Meanwhile, in Table 4, the FPR of classifier J48 was reduced to 10.52% but 'putty_syscall' and 'notepad_syscall2' are two data has been misclassified as malware. Contrast with the result of the integrated classifier Jrip+J48, which has 4.54% of false positive because only one data system calls have been misclassified as a malware, which is 'firefox_syscall'. In this case, the difficulty of a classifier to distinguish between malware and benign resulted in misclassified all the data as malware. Thus, the combination of Jrip+J48 proves that this classifier has an ability to detect malware or attack in the system call as it results lower false positive compared with Jrip and J48 classifier.

Table 3. Classification table.

		Predicted	
		Normal	**Attack**
Observed	Normal	TN	FP
	Attack	FN	TP

Table 4. Classification result.

Algorithm	**False Positive (FP)**	**False Negative (FN)**	**Detection Attack Rate (DAR)**	**Accuracy**
Jrip	23.07%	50%	95.24%	75%
J48	10.52%	28.57%	90.47%	85.71%
Jrip + J48	4.54%	0%	100%	96.42%

Moreover, the integrated classifier of Jrip+J48 yields lower false negative with 0% because there is no attack found in the system. In the meantime, classifier J48 and Jrip produce false negative with 28.57% and 50% respectively. Jrip has highest false negative compared with J48 as Jrip has been misclassified 'w17_syscall' and 'w20_syscall' as normal while classifier J48 only misclassified 'w20_syscall' as

normal. The result shows that it will be risky for an organization if there is attack or data that has been misclassified and not discovered in the system network. Hence, the integrated classifier of Jrip+J48 is the greatest choice as it has lower false negative and yielded rules for classification.

Furthermore, integrated classifier Jrip+J48 has the highest detection attack rate with 100% compared with J48 and Jrip, which 90.47% and 95.24%. This shows that integrated classifier Jrip+J48 have abilities to identify categorizing the attack accurately. Since the result of detection attack have 100% abilities to detect the malware, it shows that the classifier has capable to differentiate the classification of normal and attack since it has the better expectation. Therefore, integrated classifier Jrip+J48 fits and good in expecting the outcome variable since it indicates the increase in of the correct percentage for the classification of the attack compared with another classifier.

In addition, the integrated classifier between J48 and Jrip significantly improved the result despite the data being imbalanced. Other than that, we identified that the integrated classifier of these two classifiers turned out to produce the excellent result among the other approach. Besides, the integrated classifier yielded the best outcomes in identifying malware when classified against the benign samples. Finally, we conclude that, the integrated classifier between J48 and Jrip as a viable option for the malware detection in the system call.

Further comparison

In this section, the further comparison with other classifier such as Naïve Bayes, OneR, PART, Random Forest, Support Vector Machine (SVM) and Neural Network has been conducted. Table 5 shows, the result of classification.

Based on Table 5, it concludes that the Jrip+J48 classifier and PART classifier has the higher value of accuracy with 96.42% compared to another classifier. However, the PART classifier has the high value of false negative compared to Jrip+J48 classifier with 12.5% as only one data system calls have been misclassified as a malware, which is 'w17_syscall' Meanwhile, ONeR classifier has the lowest value of accuracy among the other classifier with 25%. The value of false negative for OneR is very high, which 75% of it has been misclassified 21 data of system call as a normal data as shown in Table 6.

Besides, PART and OneR does not generate any false alarm as the value of positive rate is 0% but the classifier still can detect malware in the system call. This showed that the organization can be very dangerous since of lots of attacks were not discovered and the false alarm does not produce. Hence, it concludes that the classifier of PART and OneR are not the variable option for malware detection in system call as the classifier cannot accurately distinguish the malware.

In addition, Jrip+J48, Random Forest and Neural Network classifier has the higher value for detection attack rate, which is 100%. Yet, Neural Network and Random Forest still has the high value of false positive rate with 25% and 19.23%. In this case, the classifier of Neural Network has been misclassified 'putty_syscall', 'firefox_syscall', iexplore_syscall', 'iexplore_syscall2' and 'notepad_syscall2', 'winrar_syscall' and 'calc_syscall' as a malware data. Nonetheless, Random Forest not classified 'winrar_syscall' and 'calc_syscall' as a malware data while Jrip+J48

only classified 'firefox_syscall' as a malware data. Thus, it shows that Jrip+J48 classifier is the best in detecting malware attack in the system call compared to another classifier.

Table 5. Classification result.

Algorithm	False Positive (FP)	False Negative (FN)	Detection Attack Rate (DAR)	Accuracy
Jrip	23.07%	50%	95.24%	75%
J48	10.52%	28.57%	90.47%	85.71%
Jrip + J48	4.54%	0%	100%	96.42%
Naive Bayes	13.04%	20%	95.23%	85.71%
OneR	0%	75%	0%	25%
PART	0%	12.5%	95.23%	96.42%
Random Forest	19.23%	0%	100%	82.14%
SVM	4.76%	14.28%	95.23%	92.85%
Neural Network	25%	0%	100%	75%

Table 6. Result of misclassified data by algorithm.

Algorithm	False Positive (FP)	False Negative (FN)
JRip	iexplore_syscall, firefox_syscall putty_syscall, notepad_syscall2, calc_syscall, iexplore_syscall2	w20_syscall
J48	putty_syscall, notepad_syscall2	w17_syscall, w20_syscall
Jrip + J48	firefox_syscall	-
Naive Bayes	putty_syscall, notepad_syscall2, calc_syscall	w17_syscall
OneR	-	w24_syscall, w89_syscall, w12_syscall, w54_syscall, w22_syscall, w34_syscall, w17_syscall, w58_syscall, w20_syscall, w70_syscall, w21_syscall, w30_syscall, w66_syscall, w55_syscall, w46_syscall, w50_syscall, w87_syscall, w18_syscall, w39_syscall, w94_syscall, w84_syscall
PART	w17_syscall	-
Random Forest	iexplore_syscall, firefox_syscall putty_syscall, notepad_syscall2, iexplore_syscall2	-
SVM	putty_syscall	w17_syscall
Neural Network	putty_syscall, firefox_syscall, iexplore_syscall, iexplore_syscall2, notepad_syscall2, winrar_syscall, calc_syscall	-

Finally, based on above explanation, we can conclude that Jrip+J48 classifier still the best approach for the malware detection in the system call. The result of JRip+J48 classifier has been outperforming other as shown in Table 5. Figure 4 shows a set of rules generated based on the results obtained in Table 5 from the usage of JRip+J48 classifier. The set of rules consists of four rules and stated that if only all the rules are fulfilled, then it will classify the data as normal and respectively as malware if the rules are not met. Therefore, these rules are good for future reference to detect malware with reduced false negative as the main purpose is to detect malware in the system call accurately. The result of Jrip+J48 classifier has been outperforming other classifier and produce the best outcome in recognizing malware attack.

```
JRip.J48 rules:
===========

IF (NtIsUILanguageComitted = 0) AND (NtTerminateProcess = 0)
   THEN Normal
IF (NtMapCMFModule = 6)
   THEN Normal
IF (NtAlpcCreatePort = 9)
   THEN Normal
ELSE Malware
```

Fig. 4. JRip+J48 rule.

5. Conclusions

As a conclusion, there are many techniques that have been used by researchers to detect malware activity. Using only one detection method to detect the overall malware activities becomes a new challenge due to the difference attack behaviour that will affect the feature selection on the detection technique. Hence, identify the most significant feature and classification algorithm in malware detection is a key factor to increase the accuracy of malware detection. The classification algorithm must fit with the dataset and produce high accuracy rate of detection. Besides, from the related works shows that mostly researcher concludes that J48 and Jrip classifier provide the better accuracy. Therefore, our proposed method, the integrated classifier between J48 and JRip is the best approach and more efficient to distinguish malware as it gives high accuracy in this research. The main contribution of this work is proposed method, which consists of three phase and generates the rule of integrated J48 and JRip classifier. The limitation of this study is the feature extracted from the system call and only use two types of malware with different variant. For future works, it is recommended to develop and analyse a real behavioural antivirus platform based on classification via the integrated classifier algorithm.

Acknowledgements

This work has been supported under Universiti Teknikal Malysia Melaka research grant Gluar/CSM/2016/FTMK-CACT/100013. The authors would like to thank to Universiti Teknikal Malaysia Melaka, CyberSecurity Malaysia and all members of INSFORNET research group for their incredible supports in this project.

Nomenclatures

j	Entropy data of class j
p	Entropy data of class p
p_j	Probability of class j
R_i	Rule set
t	Data of class t

Abbreviations

DRI	Rule Induction Algorithm
FURIA	Fuzzy Unordered Rule Induction Algorithm
IG	Information Gain
IDS	Intrusion Detection System
SVM	Support Vector Machine

References

1. Grini, L.S. (2016). *Feature extraction and static analysis for large-scale detection of malware types and families*. Master's Thesis. Department of Computer Science and Technology. Gjovik University College, Norway.
2. Egele, M.; Scholte, T.; Kirda, E.; and Kruegel, C. (2012). A survey on automated dynamic malware-analysis techniques and tools. *ACM Computing Surveys*, 44(2), Article 6, 1-49.
3. Sweeney, A.M. (2015). *Malware analysis and antivirus signature creation*. Master Thesis. Letterkenny Institute of Technology, Donegal, Ireland.
4. Kerrisk, M. (2010). *The linux programming interface*. San Francisco, California: No Start Press, Inc.
5. Kolosnjaji, B.; Zarras, A.; Webster, G.; and Eckert, C. (2016). Deep learning for classification of malware system call sequences. *Proceedings of the 29th Australasian Joint Conference on Artificial Intelligence*. Hobart, Australia, 137-149.
6. Xiao, H.; and Stibor, T. (2011). A supervised topic transition model for detecting malicious system call sequences. *Proceedings of the Workshop on Knowledge Discovery, Modelling and Simulation*. San Diego, California, United States of America, 23-30.
7. Dubitzky, W.; Wolkenhauer, O.; Cho, K.-H.; and Yokota, H. (2013). *Encyclopedia of systems biology* (3rd ed.). New York: Springer-Verlag.
8. Juma, S.; Muda, Z.; Muhammad, M.A.; and Yassin, W. (2015). Machine learning techniques for intrusion detection system: A review. *Journal of Theoretical and Applied Information Technology*, 72(3), 422-429.
9. Pfoh, J.; Schneider, C.; and Eckert, C. (2013). Leveraging string kernels for malware detection. *Proceedings of the International Conference on Network and System Security*. Madrid, Spain, 206-219.
10. Witten, I.H.; Frank, E.; Hall, M.A.; and Pal, C.J. (2016). *Data mining: Practical machine learning tools and techniques* (4th ed.). Burlington, Massachusetts: Morgan Kaufmann Publishers.

11. Boukhtouta, A.; Lakhdari, N.E.; Mokhov, S.A.; and Debbabi, M. (2013). Towards fingerprinting malicious traffic. *Procedia Computer Science*, 19, 548-555.
12. Ahmad, I.N.; Ridzuan, F.; Saudi, M.M.; Pitchay, S.A.; Basir, N.; and Nabila, N.F. (2017). Android mobile malware classification using tokenization approach based on system call sequence. *Proceedings of the World Congress on Engineering and Computer Science*. San Francisco, United States of America, 6 pages.
13. Chaba, S.; Kumar, R.; Pant, R.; and Dave, M. (2017). Malware detection approach for android systems using system call logs. arXiv, 5 pages.
14. Rathore, M.M.; Ahmad, A.; and Paul, A. (2016). Real time intrusion detection system for ultra-high-speed big data environments. *The Journal of Supercomputing*, 72(9), 3489-3510.
15. Shahzad, R.K.; and Lavesson, N. (2012). Veto-based malware detection. *Proceedings of the Seventh International Conference on Availability, Reliability and Security*. Prague, Czech Republic, 47-54.
16. Qabajeh, I.; Thabtah, F.; and Chiclana, F. (2015). A dynamic rule-induction method for classification in data mining. *Journal of Management Analytics*, 2(3), 233-253.
17. Zakeri, M.; Faraji Daneshgar, F.F.; and Abbaspour, M. (2015). A static heuristic approach to detecting malware targets. *Security and Communication Networks*, 8(17), 3015-3027.
18. Kshirsagar, V.; and Joshi, M.S. (2016). Rule based classifier models for intrusion detection system. *International Journal of Computer Science and Information Technologies*, 7(1), 367-370.
19. Bhaya, W.S.; and Ali, M.A. (2017). Review on malware and malware detection using data mining techniques. *Journal of University of Babylon*, 25(5), 1585-1601.
20. Lengyel, T.K.; Maresca, S.; Payne, B.D.; Webster, G.D.; Vogl, S.; and Kiayias, A. (2014). Scalability, fidelity and stealth in the DRAKVUF dynamic malware analysis system. *Proceedings of the 30th Annual Computer Security Applications Conference.* New Orleans, Louisiana, United States of America, 386-395.
21. Alazab, M.; Venkatraman, S.; Watters, P.; and Alazab, M. (2011). Zero-day malware detection based on supervised learning algorithms of API call signatures. *Proceedings of the Ninth Australasian Data Mining Conference*. *Ballarat*, Australia, 171-181.