TELKOMNIKA Telecommunication, Computing, Electronics and Control
Vol. 19, No. 4, August 2021, pp. 1396~1406
ISSN: 1693-6930, accredited First Grade by Kemenristekdikti, Decree No: 21/E/KPT/2018
DOI: 10.12928/TELKOMNIKA.v19i4.20402

Online traffic classification for malicious flows using efficient machine learning techniques

Ying Yenn Chan[1], Ismahani Bt Ismail[2], Ban Mohammed Khammas[3]
[1,2]School of Electrical Engineering, Universiti Teknologi Malaysia, Malaysia
[3]Department of Computer Networks Engineering, Collage of Information Engineering, Al-Nahrain University, Iraq

Article Info

Article history:

Received Sep 2, 2020
Revised Mar 9, 2021
Accepted Mar 20, 2021

Keywords:

Machine learning
Malicious traffic flows
Online classification
Snort alerts
Statistical features

ABSTRACT

The rapid network technology growth causing various network problems, attacks are becoming more sophisticated than defenses. In this paper, we proposed traffic classification by using machine learning technique, and statistical flow features such as five tuples for the training dataset. A rule-based system, Snort is used to identify the severe harmfulness data packets and reduce the training set dimensionality to a manageable size. Comparison of performance between training dataset that consists of all priorities malicious flows with only has priority 1 malicious flows are done. Different machine learning (ML) algorithms performance in terms of accuracy and efficiency are analyzed. Results show that Naïve Bayes achieved accuracy up to 99.82% for all priorities while 99.92% for extracted priority 1 of malicious flows training dataset in 0.06 seconds and be chosen to classify traffic in real-time process. It is demonstrated that by taking just five tuples information as features and using Snort alert information to extract only important flows and reduce size of dataset is actually comprehensive enough to supply a classifier with high efficiency and accuracy which can sustain the safety of network.

Corresponding Author:

Ying Yenn Chan,
School of Electrical Engineering
Universiti Teknologi Malaysia
81310 Skudai, Johor Bahru, Malaysia
Email: yychan2@graduate.utm.my

1. INTRODUCTION

According to Webroot Threat Report that was written in 2019, 93.6% of malware spotted on one single computer. This is the highest annual rate that have ever seen, even though the number has risen beyond 90% since 2014. More than two-thirds of IT security professionals consider that a successful cyber-attack is coming up in 2020 [1], [2]. Numerous type of traffic classification techniques have been used such as port based, payload based, statistical approach and behavioral based with a common aim of classifying data packets or flows effectively. However, network attack tactics have gradually become more complex and can hardly be detected [3]. For example, the growing and new trends of application developers to avoid the detection leave this network traffic classification field open for further research. In a nutshell, the target of this paper is to propose a solution to real time network traffic classification that could overcome current research gap for better human safety in the cyber world.

Network traffic classification [4], [5] is flows identification of the network traffic and positioning each of the flows to various classes according to their feature information like port number [6], [7], payload [8]-[11], [12], traffic behaviours [13]-[16] and flow information [4], [8], [9], [12], [17]. In this paper, an efficient flow information based classification of suspicious network traffic flow is introduced. Here, ‘efficient’ means an extra rule-based system, Snort will be used to reduce size of training data to improve detection accuracy and efficiency. The evaluation started by offline traffic classification. Computational performance in terms of accuracy and efficiency is compared among machine learning (ML) algorithms using the reduced size of training data set that consists of only the most severe malicious data. This comparison is done to choose the best classifier that will be used for the online traffic classification later. The paper is organized as follows. Section 2 describes the literature review. Section 3 presents methodology of this paper. Section 4 provides the results and performance of the proposed idea. Conclusion is in section 5.

2. LITERATURE REVIEW

2.1. Overview

Several common methods have been used for traffic classification. Port-based approach has been commonly used and is considered the quickest and low-resource consuming method in the case of classifying network traffic packets. There are some applications that have fixed (or traditionally used) port numbers, such as WWW and email. Thus, it is easy to detect the traffic that belongs to these applications. However, there are also applications do not have a fixed port number, such as peer-to-peer (P2P), games, and multimedia. Instead, they will use the port numbers of other widely used applications such as (hypertext transfer protocol), HTTP/file transfer protoco (FTP) connections [18]. Therefore, this approach sometimes could yield poor outcomes because attackers tend to play tricks easily by modifying the port number periodically in the system and pretend like a usual normal application [6].

Payload-based approach examines the packets’ contents to identify the types of traffic. This method identifies all signatures that are found in the payload of the application layer. After the algorithm successfully collected a set of unique payload signatures, it results in good performance for most types of traffic, such as HTTP, FTP, and simple mail transfer protocol (SMTP) [6]. However, the effectiveness of these methods also has greatly reduced due to the fact that the customers use encrypted flows, while governments decided to band to have third parties to involve in the system to examine payloads for safety purposes. In addition, the inspection process of packets payload syntax could give a heavy operational load and delay [9].

Heuristic approach works by checking the suspicious behaviours of targeted files, monitoring files in the system such as (system documents and service), observing process in the system and application programming interfaces. There are two types of detection such as static detection and dynamic detection, in which they have one important difference is to decide on the options of running or not the detected documents and do the checking of suspicious operation. However, this method cannot promise to find all the Trojans, plus installing the system on every computer of the whole network is an issue as well. If any of the computers in the network that are not covered by using this method, chances are there that malware could disperse to the other computers until the entire network is infected from that specific unprotected computer [15].

Statistical approach using ML algorithm depends on the classifying of statistical information such as frequency and length of bytes, size of packets and inter-arrival time of packets transmitted. This technique is fast and capable of detecting and analysing the class categories of the unknown applications. Therefore, by having a complete statistical information of the targeted packets during inspections, classifying the hidden protocol will be easy. This approach has become popular as encrypting traffic of applications becoming a new trend that causes challenges for any of the proposed classification tools that previously claimed to be able to achieve high accuracy in applications classification. However, the accuracy might drop if the training data is insufficient as high amount of data is needed as learning process [4], [9]. In Weka [19], various ML algorithms are tested and compared. Naïve Bayes [20] is a great tool for knowledge representation as well as reasoning. It could calculate the probability of a new variables subset when a subset of random variables, also known as evidence variables are provided. Naïve Bayes is a non-complex probabilistic classifier which follows Bayes theorem as shown in (1);

$$P(c|x) = \frac{P(X|C)P(c)}{P(x)} \tag{1}$$

where *P(c|x)* is the posterior probability, *P(x|c)* is the probability of predictor given class, *P(c)* is the class prior probability and *P(x)* is the predictor prior probability. Benefit of using Naïve bayes is that it is possible to predict the important classification parameters with a trivial amount of training data.

2.2. Related works

Researchers have generated enormous ways to tackle the issue using methods presented in the Section 2.1. For example, Shim *et al.* [11] has proposed a method where it is able to generate payload signature automatically. This study can cut down on the amount of time spent generating signatures that required to be done manually. However, application traffic input, has to be manually collected. Moreover, it is difficult to do signature extraction although the traffic was gathered using one single function and also mixing the traffic up with other features could happen easily.

Galal *et al.* [14] has proposed a behavior-based features model that helps to define suspicious activities exhibited by malware. The major difficulty of this method is the runtime overhead. According to Bekerman *et al*. [15], 972 behavioral features were extracted across different protocols and network layers, but might classify data packets wrongly, causing false positive. Once attacks change behaviors, the classifiers cannot work well and need to be retrained. Finamore *et al.* [12] has integrated both flow and payload statistical feature based clustering for classifying unknown traffic. However, the amount of clusters has to be enormously high to attain good accuracy in classification performance, which then results in an issue of having to use a large dataset for a small applications [17]. Furthermore, the goodness of their features may limited by encrypted application layer protocols. Zhang *et al.* [17] also has proposed an approach that has the ability of identifying anonymous flows created by unknown applications and using the associated information in the actual network traffic to enhance classification result. However, this method often lead to high false detection.

Therefore, after looking at these research gaps, the proposed solution in this paper is a malicious traffic classification using ML method and statistical features i.e five tuples with further assisted by Snort alert for an efficient classification. Unlike the existing similar works mentioned above [12], [17] of using all traffic priorities, information from the Snort alert in terms of malicious traffic priorities is used to augment the ML which it is used to extract out only important features and avoid redundant data, while at the same time speed up the training process and improve the selected Naïve Bayes classifier accuracy.

3. RESEARCH METHOD

The project was executed using Dell (Inspiron-14) laptop in which its operating system was Windows 10 Enterprise 64-bit, version 1809 (build 17761.1158). Processor is Intel ® Core ™ i5-5200 U CPU @ 2.20GHz. The installed RAM of this laptop is 4.00GB. This device will be used as the main laptop throughout the project implementation. While during online classification, another laptop will be added to transmit real time data packets to the main laptop. The second laptop (laptop B) is a Dell laptop that uses Windows 10 Pro, 64-bit Operating System. Processor of the laptop is Intel® Core™ i5-4310 U CPU @ 2.00GHz. In addition, it has the same RAM of 4.00GB.

3.1. Project framework

Figure 1 shows the overall framework of this paper. There are two phases consisting of offline and online classification. Various stages need to be done to get an optimum accuracy and performance in phase 1 as a stable foundation for phase 2 online classification. As illustrated in Figure 1, the aims are flows reconstruction, features extraction, Snort filtering flows with high severity, classifier model generation as well as comparing results between different classifier algorithms. Key evaluators are accuracy and efficiency of the model. Accuracy is the rate of precision and exactness, while, efficiency is the processing time to build the model. According to the best accuracy and efficiency, the fit classifier model can be obtained. Phase 2 will focus on online classification which implementing the similar steps in phase 1 in order to get the same performance as offline classification to prove its effectiveness.

3.1.1. Five-tuples as input features

A 5-tuple can be defined as a set consisting of five distinct numbers that comprise a transmission control protocol/internet protocol (TCP/IP) connection. It consists of a source and destination IP address, port number, and the protocol used. Each protocol type will has its own ID number, for example 'TCP' is port 6, and while 'UDP' is port 17. In our case, protocol ID will replace protocol type in the dataset. The arrangement of features are as follows: destination IPs, destination ports, source IPs, source ports, transport layer protocol and types of flows. Below is the example of the traffic flows.

```
192.168.202.79, 50465, 192.168.229.251, 80, 6, Malware
192.168.229.254, 443, 192.168.202.79, 46119, 6, Malware
```

3.1.2. Offline classification process

Offline classification in phase 1 begins when a large amount of offline malware data is downloaded from reliable websites [21]-[23]. As the files are from different sources, files merging in Wireshark is necessary. After the files are compiled properly in one, the data packets are transferred to Caploader [24] to be reconstructed into flows [18], [25] so that 5 tuples of the data packets can be extracted out in csv format using Caploader. On the other hand, external assistance from Snort will be required because Snort is capable of inspecting the network packets injected for possible malicious traffic through the predefined rule and log into the alert file when the packet signature matches with one of the rules. After using intrusion detection system (IDS) mode and scanning all the incoming packets, classify them to classes according to priority ranging from 1-4 as well as filtering out the non-malicious data packets, a file in pcap format and a log text file will be generated. The log text file is the outcome of Snort analysis, all malicious packets will be listed out one by one together with the priority levels and other packets information while the pcap file consisting of only data packets that are malicious.

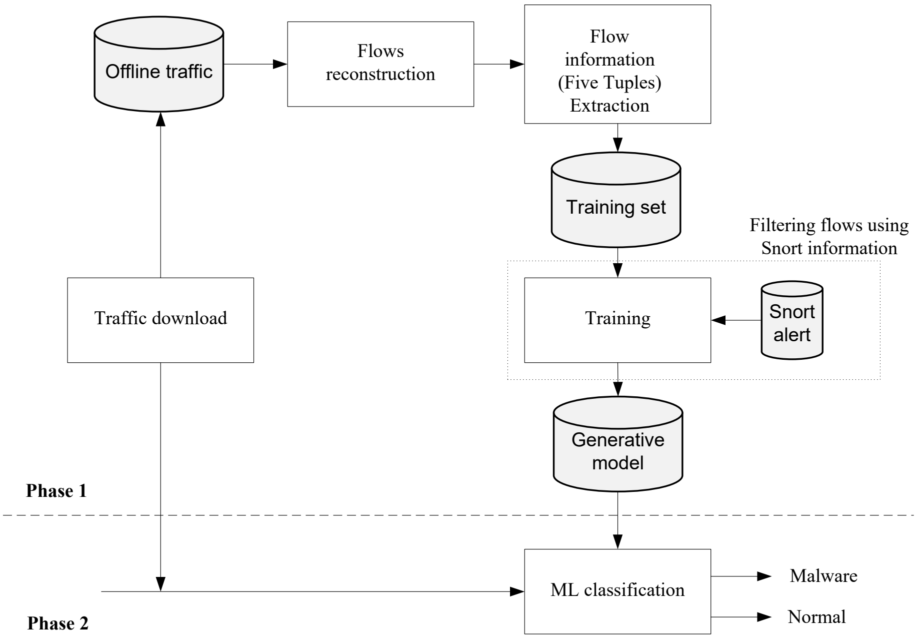


Figure 1. Overview framework

Figure 2 presents in detail the filtering flows using Snort information part from Figure 1. According to Figure 2, with Snort alert file as shown in Figure 3, five tuples of priority 1 will be filtered out, along with the list of clean and normal data features, arranged accordingly in csv file, and labels are added to form the dataset that will be used to create generative model afterward. This is a step of reducing the training set size and forming a good and compact training dataset. Flows with priority 1 shows the most harmful malicious packet according to the Snort rules. Extracted data flows with priority 1 means all of the data are pretty malicious. Statistic of the training set is shown in the Table 1. To prove the effectiveness of using only priority 1 dataset, a complete all priorities dataset is prepared as well for comparison as shown in Table 1. The csv file of the datasets are converted to the Arff format for Weka [19] software to undergo further analysis on different algorithms such as Bayes Net [20], [26], Naïve Bayes [27], [28], Random Tree [29], J48 [30], [31] and ZeroR [32], [33] to get the best efficiency and accuracy. In this paper, five algorithms that suitable for text classification [34] are used and analyzed. After the best algorithm is chosen, a classifier model will be saved and generated hence marked the end of phase 1.

3.1.3. Online classification process

Then, followed by online classification in phase 2, command lines with different roles are included in the C++ program. The processes in the program are exactly the same as the offline classification process, but in real time as shown in Algorithm 1. The traffic will be transmitted from laptop B to the main laptop through PlayCap, the Wireshark installed in the main laptop captured, analyzed the flows and classified them accordingly into malicious and normal flows following their features similarity to the generative model.

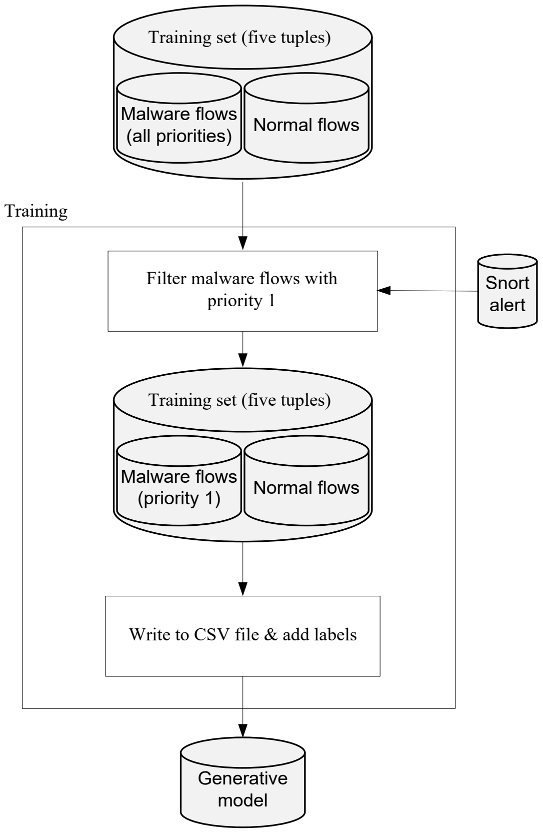
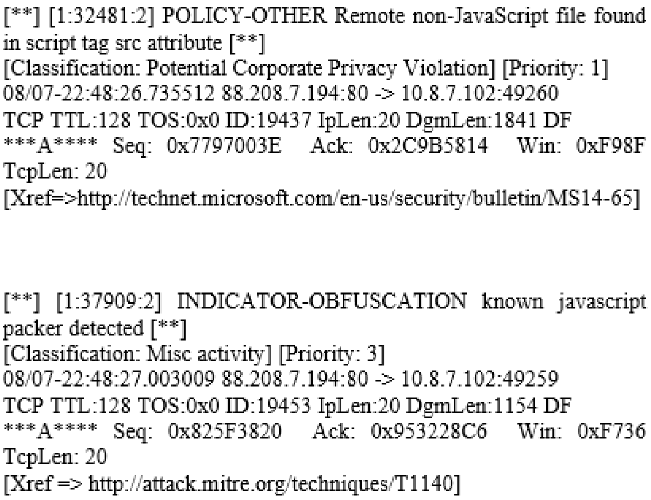

Training set (five tuples)
Malware flows (all priorities)
Normal flows
Training
Filter malware flows with priority 1
Snort alert
Training set (five tuples)
Malware flows (priority 1)
Normal flows
Write to CSV file & add labels
Generative model

Figure 2. Production of priority 1 dataset



```
[**] [1:32481:2] POLICY-OTHER Remote non-JavaScript file found
in script tag src attribute [**]
[Classification: Potential Corporate Privacy Violation] [Priority: 1]
08/07-22:48:26.735512 88.208.7.194:80 -> 10.8.7.102:49260
TCP TTL:128 TOS:0x0 ID:19437 IpLen:20 DgmLen:1841 DF
***A**** Seq: 0x7797003E Ack: 0x2C9B5814 Win: 0xF98F
TcpLen: 20
[Xref=>http://technet.microsoft.com/en-us/security/bulletin/MS14-65]


[**] [1:37909:2] INDICATOR-OBFUSCATION known javascript
packer detected [**]
[Classification: Misc activity] [Priority: 3]
08/07-22:48:27.003009 88.208.7.194:80 -> 10.8.7.102:49259
TCP TTL:128 TOS:0x0 ID:19453 IpLen:20 DgmLen:1154 DF
***A**** Seq: 0x825F3820 Ack: 0x953228C6 Win: 0xF736
TcpLen: 20
[Xref => http://attack.mitre.org/techniques/T1140]
```

Figure 3. Snort alert file

Table 1. Statistic of all priorities and prioritiy 1 training dataset

Proto Type	Proto ID	All Priorities Quantity		All Priorities	Priority 1 Quantity		Priority 1
		Norm. Flows	Malw. Flows	Total Flows	Norm. Flows	Malw. Flows	Total Flows
TCP	6	19309	53210	72519	19309	53129	72438
UDP	17	5211	292	5503	5211	292	5503
HOPOPT	0	34	2	36	34	0	34
ICMP	1	529	12280	12809	529	239	768
IGMP	2	277	0	277	277	0	277
Unknown	99	0	1107	1107	0	885	885

Algorithm 1. Real time classification

```
program start
variable seconds=30  //capture for 30s
variable n=1 //indicating first row of csv file  (header)
set ip.dst,tcp.dstport,ip.src,tcp.srcport,protocol,type => allFlowLabelNew // set
new header
for all captured data packet do
ip.dst, tcp.dstport, ip.src, tcp.srcport, ip.proto, null => allFlowLabelRaw
//elements captured from pcapng file
end for
while capture function == '1'  //capturing process
for seconds !=0 do
   Rep.pcapng=capture all
   seconds=seconds-1
end while
while csvconverter function == '1'
   res.csv = extract allFlowLabelRaw from Rep.pcapnng //extract five tuples from
pcapng
end while
if n of allFlowLabelRaw!=n of allFlowLabelNew then
 n of allFlowsLabelRaw =n of allFlowLAbelNew //change header to be same like
training dataset
end if
while classification function == '1' do
res.txt = output prediction of Weka NaiveBayes Classifier //classification process
end while
program end
```

The flowchart in Figure 4 illustrated the whole process of the online traffic classification whereas statistics of flows captured are shown in Table 2. Considering online classification needs an incoming real time data packets to be transmitted, laptop B will be prepared and served as a router to send data packets to the main laptop. PlayCap will be installed in laptop B and inserted with the prepared pcap dataset. It will send the packets one by one from the prepared file to the main laptop as a ‘real time’ incoming data. While in the main laptop, it will capture all the data packets received by using Wireshark. For experiment purpose, it will first start off with packets capturing in 30 seconds by using Dumpcap from Wireshark and saved in a PcapNg file. Then, Tshark extract out the five tuples information from TCP flows and write in a csv file. The C++ program will take charge of changing the header of data in the saved csv file to the same header as the training dataset so that it can be read in Weka. After it is settled, Weka will classify the flows into two classes which are malware and normal according to their statistical features based on the generative model. Output of online predictions will be compared with offline procedures outcome to verify its functionality to be implemented in real life.

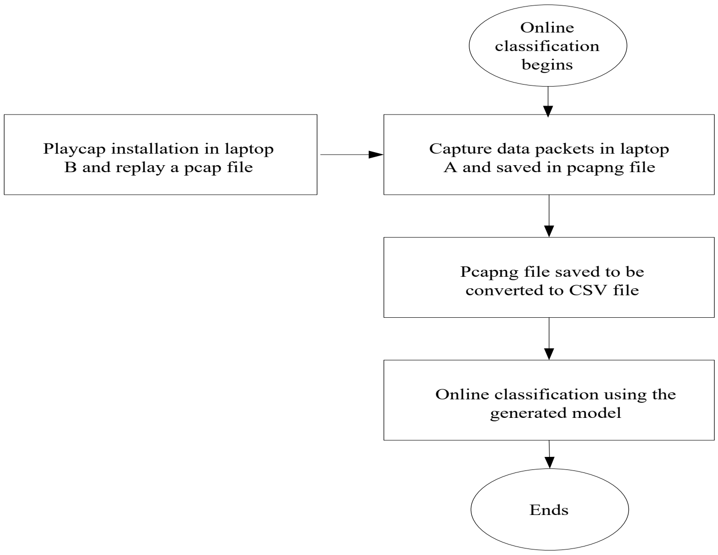

Online classification begins
Playcap installation in laptop B and replay a pcap file
Capture data packets in laptop A and saved in pcapng file
Pcapng file saved to be converted to CSV file
Online classification using the generated model
Ends

Figure 4. Online classification flowchart

Table 2. Statistic of online test set

Protocol Type	Protocol ID	Quantity of Data Flows	Class	Total Quantity of Data Flows
TCP	6	703	Malware	703

3.2. Evaluation parameters

In order to analyse and evaluate the results, confusion matrix shown in Table 3 is used to speculate each classifier. TP, FP, TN and FN symbolize the number of correctly identified malware flows, number of wrongly identified normal flows, number of correctly identified normal flows and number of wrongly identified malware flows, respectively. Taking these symbols of confusion matrix, parameters that assess the performance of classification can be well-defined. Parameters to evaluate a classifier performance are shown in the Table 4, (2) to (7), respectively.

Table 3. Confusion Matrix

	Identified as Normal Flows	Identified as Malware Flows
Normal Flows	*TN*	*FP*
Malware Flows	*FN*	*TP*

Table 4. Parameter equations

Performance Parameters	equ
$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$	(2)
$TP\ rate = \frac{TP}{TP+FN}$	(3)
$FP\ rate = \frac{FP}{FP+TN}$	(4)
$Precision = \frac{TP}{TP+FP}$	(5)
$Recall = \frac{TP}{TP+FN}$	(6)
$F-Measure = \frac{2*Precision*Recall}{Precision+Recall}$	(7)

4. RESULTS AND ANALYSIS

In order to investigate the functionality of the proposed method in this paper, a series of tests were conducted thoroughly in two parts, offline and online analysis. The purpose of offline part is to find out the effectiveness of five tuples of priority 1 dataset in traffic classification using a variety of algorithms in Weka. While online analysis intends to further evaluate the effectiveness of finalized model on incoming real time data.

4.1. Test scenario–offline classification

In offline analysis, firstly, five algorithms were experimented and analysed as shown in Table 5 for two types of datasets (all priorities data set and training data set consisting of priority 1 flows from Table 1). The tested algorithms are Naïve Bayes, random tree, J48, Bayes Net and ZeroR. The first four algorithms are chosen because they work better with text classification whereas the ZeroR is used as the minimum benchmark of model performance. This evaluation is to ensure that by only using the priority 1 dataset is sufficient and strong enough to generate a decent and promising generative model.

4.1.1. Algorithms effect on datasets using 10 fold cross validation

The prediction is done by using 10 fold cross validation test option. The value of folds is elected so that each subset of data is sufficient to be statistically illustrative of the wider full dataset. The choice of fold number is usually 5 or 10, when the value gets higher, the size variance between the training set and subsets becomes smaller. In this case, usually 10 fold can achieve the average accuracy for a classifier [35], [36]. Hence 10 fold cross validation is selected without further experimentation. The accuracy and efficiency of each of the algorithms tested on all priorities data set and solely priority 1 data set are listed in Table 5. The number of instances between the two training datasets are differ by 12346 instances. The full dataset contains 92251 while the extracted dataset has only 79905.

Table 5. Classification performance using 10 fold cross validation

	All Priorities		Priority 1	
Model	Accuracy (%)	Efficiency(s)	Accuracy (%)	Efficiency(s)
Naïve Bayes	99.8678	0.18	99.9224	0.06
RandomTree	93.5912	0.86	99.4168	0.30
J48	99.6565	0.32	99.6008	0.31
ZeroR	72.5208	0.06	68.2623	0.06
BayesNet	99.8581	0.75	99.9675	0.60

According to Table 5, the performance of priority 1 dataset is generally better or similar to all priorities dataset. Smaller dataset took lesser time than full priorities hence more efficient. Except for J48 and ZeroR that have slight lower accuracy, all other algorithms tested on priority 1 dataset are estimated to be able to predict better when new dataset injected. In an overall trend, experiment shows that by using extracted priority 1 dataset, it has a better performance in terms of efficiency and accuracy. Among all algorithms, Naive Bayes will be the most convincing classifier due its accuracy of 99.92% and efficiency of 0.06s when priority 1 dataset and cross validation of 10 folds were applied.

4.1.2. Performance of finalised algorithmsci–Naïve Bayes

Table 6 presents Naïve Bayes classification performance for the priority 1 dataset in detail. There are a number of important parameters to be emphasized on other than the accuracy and efficiency. Confusion matrix illustrated the raw number of correctly and incorrectly classified instances. The addition of aa, ab, ba and bb will be equal to the total number of instances of 79905 while a and b are the class label (normal and malware). High precision means the algorithm is able to bring significantly applicable results than the irrelevant ones while high recall indicating that more relevant results are returned. F-measure specifies the model accuracy by combining recall and precision of the model. Other parameters such as receiver operating characteristics (ROC) Area and Kappa statistic also give a verification on the accuracy level of the model. The most optimum classifier would have the value of ROC and Kappa approaching or equals to 1. Again as proven in Table 6, it is a promising classifier as its ROC area as well as the Kappa statistic is almost equal to 1.

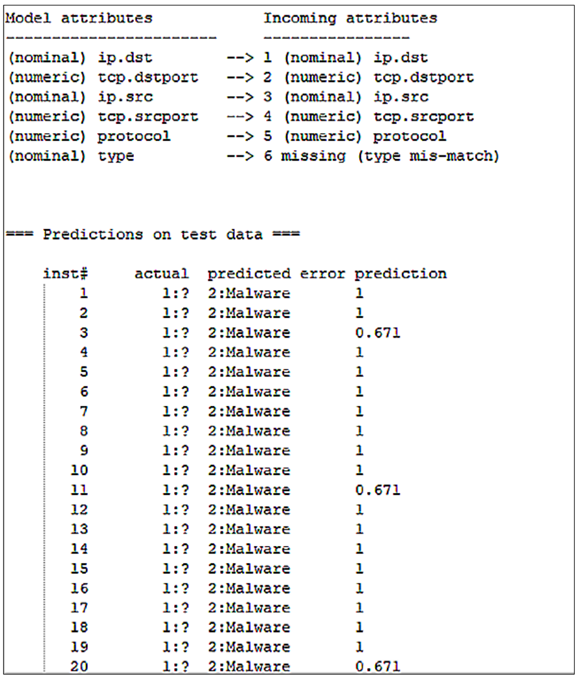
Table 6. Detailed classification performance for Naïve Bayes

Percentage of Correctly Classified Instances	Weighted Avg.		Confusion Matrix		
99.9224%	TP Rate	0.999		a	b
	FP Rate	0.000	a	25360	0
	Precision	0.999	b	62	54483
	Recall	0.999	a = Normal		
	F-Measure	0.999	b = Malware		
	Kappa Statistic	0.9982			
	ROC Area	1.000			

4.2. Test scenario–online classification

The statistic of dataset supplied as new test set is shown in Table 2. The online analysis started by supplying data from laptop B using Playcap, while main laptop captured and played as a simulation of traffic classification in real life. Naïve Bayes model is used to predict the newly captured data, with headers edited csv file because the newly supplied dataset must possess the same attributes as the dataset in the saved model. Otherwise it will not be recognized. By using the TCP flows from Table 2 that captured from laptop B, it attempts to match attributes between two datasets before prediction. As shown in Figure 5, the first five attributes were perfectly matched except for the last one, which is the prediction result we are looking for. It is detected as mismatched because the incoming data flow does not have a label as the saved model dataset. Hence, after prediction happened, it generates results that contain both the actual and predicted class for the type label of each instance in the test set as shown in Figure 5. Its error prediction should be exactly the same when Weka GUI is used. In order to validate the online classification result, a Weka Explorer testing was done on the same newly captured PcapNg from Playcap in laptop B by using the saved model. As shown in the Figure 6, the predictions on the test set is exactly the same as the results produced from online classification as shown in Figure 5. The prediction in the output is its probability of correct prediction. Hence, the closer to '1', the better the accuracy of each prediction. While the predicted class also correctly assigned, as to ease the verification, only malware packets are sent from laptop B. 'Malware' is labeled to each instance which is proven that the predictions are all correct and possible to make reality.

Based on the outcome from the experiments conducted, it verifies that ML method using statistical features and assisted by Snort alert is sufficient and precise enough to provide an accurate answer for each instance. The performance of priority 1 dataset which contain only important features and avoid redundant data is generally better or similar to all priorities dataset. Training with smaller dataset takes lesser time than with full priorities and can produce more efficient traffic classification performance.

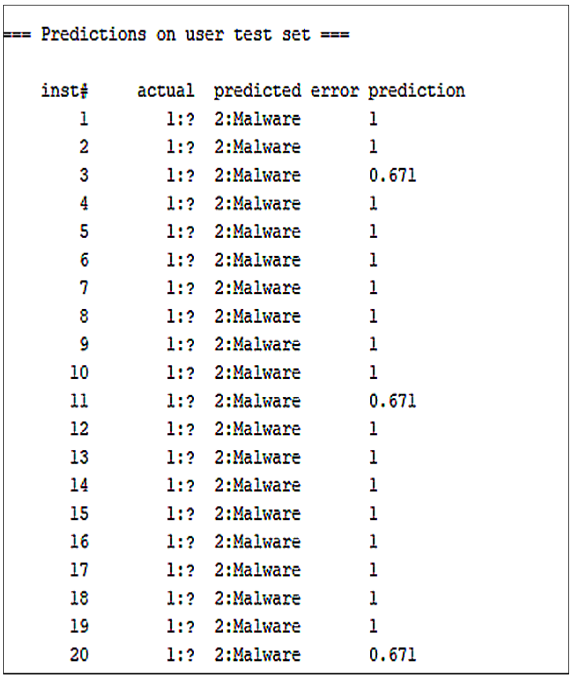


```
Model attributes               Incoming attributes
-----------------------        ----------------
(nominal) ip.dst          --> 1 (nominal) ip.dst
(numeric) tcp.dstport     --> 2 (numeric) tcp.dstport
(nominal) ip.src          --> 3 (nominal) ip.src
(numeric) tcp.srcport     --> 4 (numeric) tcp.srcport
(numeric) protocol        --> 5 (numeric) protocol
(nominal) type            --> 6 missing (type mis-match)


=== Predictions on test data ===

    inst#     actual  predicted error prediction
        1        1:?  2:Malware       1
        2        1:?  2:Malware       1
        3        1:?  2:Malware       0.671
        4        1:?  2:Malware       1
        5        1:?  2:Malware       1
        6        1:?  2:Malware       1
        7        1:?  2:Malware       1
        8        1:?  2:Malware       1
        9        1:?  2:Malware       1
       10        1:?  2:Malware       1
       11        1:?  2:Malware       0.671
       12        1:?  2:Malware       1
       13        1:?  2:Malware       1
       14        1:?  2:Malware       1
       15        1:?  2:Malware       1
       16        1:?  2:Malware       1
       17        1:?  2:Malware       1
       18        1:?  2:Malware       1
       19        1:?  2:Malware       1
       20        1:?  2:Malware       0.671
```

Figure 5. Output prediction of online classification



```
=== Predictions on user test set ===

    inst#     actual  predicted error prediction
        1        1:?  2:Malware       1
        2        1:?  2:Malware       1
        3        1:?  2:Malware       0.671
        4        1:?  2:Malware       1
        5        1:?  2:Malware       1
        6        1:?  2:Malware       1
        7        1:?  2:Malware       1
        8        1:?  2:Malware       1
        9        1:?  2:Malware       1
       10        1:?  2:Malware       1
       11        1:?  2:Malware       0.671
       12        1:?  2:Malware       1
       13        1:?  2:Malware       1
       14        1:?  2:Malware       1
       15        1:?  2:Malware       1
       16        1:?  2:Malware       1
       17        1:?  2:Malware       1
       18        1:?  2:Malware       1
       19        1:?  2:Malware       1
       20        1:?  2:Malware       0.671
```

Figure 6. Output prediction of offline classification

5. CONCLUSION

As conclusion, traffic classification using ML approach with five tuples features and assisted by Snort alert information could provide an efficient classification based on the classification accuracy and training processing time that have been achieved. This classifier is produced within the desired time frame and the outcome is following closely to the expectation. This proposed method is capable to reduce the unclassified traffic network and be a promising way for securing the Internet users. Combining with the real time online classification of unwanted data, our devices and information safety can be sustained. At the end of this project, some recommendations are needed to make this project a better one for the next researcher. To have a more comprehensive result, the training data set can collect more data flows that supported by a variety types of protocol, so that the classifier can be more accurate when it is tested with new dataset. Moreover, experiments on a real network with different types of malicious traffic should be implemented as it would greatly improve this research.

ACKNOWLEDGEMENTS

This research was supported by Grant with cost center no. R.J130000.2651.17J38. We gratitude extends to Universiti Teknologi Malaysia (UTM) for funding this project. UTM also deserves great thanks for giving the resources and supplying the relevant literatures in the PSZ libraries and digital libraries.

REFERENCES

[1] Webroot, “2020 Webroot Threat Report,” *Broomfield, Colorado*, pp. 6-10, 2020. [Online] Available at: https://mypage.webroot.com/rs/557-FSI-195/images/2020 Webroot Threat Report_US_FINAL.pdf

[2] G. Cirillo and R. Passerone, "Packet Length Spectral Analysis for IoT Flow Classification Using Ensemble Learning," *in IEEE Access*, vol. 8, pp. 138616-138641, 2020, doi: 10.1109/ACCESS.2020.3012203.

[3] N. A. Khater and R. E. Overill, "Network Traffic Classification Techniques and Challenges," in *The Tenth International Conference on Digital Information Management (ICDIM 2015)*, 2015, pp. 43-48, doi: 10.1109/ICDIM.2015.7381869.

[4] R. U. Khan, R. Kumar, M. Alazab, and X. Zhang, "A Hybrid Technique to Detect Botnets, Based on P2P Traffic Similarity," *2019 Cybersecurity and Cyberforensics Conference (CCC), Melbourne, Australia,* 2019, pp. 136-142, doi: 10.1109/CCC.2019.00008.

[5] H. A. H. Ibrahim, O. R. Aqeel Al Zuobi, M. A. Al-Namari, G. MohamedAli and A. A. A. Abdalla, "Internet traffic classification using machine learning approach: Datasets validation issues," *2016 Conference of Basic Sciences and Engineering Studies (SGCAC), Khartoum*, 2016, pp. 158-166, doi: 10.1109/SGCAC.2016.7458022.

[6] V. Labayen, E. Magaña, D. Morató, and M. Izal, "Online classification of user activities using machine learning on network traffic," *Computer Networks*, vol. 181, 2020, doi: 10.1016/j.comnet.2020.107557.

[7] Y. Dhote, S. Agrawal and A. J. Deen, "A Survey on Feature Selection Techniques for Internet Traffic Classification," *2015 International Conference on Computational Intelligence and Communication Networks (CICN),* Jabalpur, 2015, pp. 1375-1380, doi: 10.1109/CICN.2015.267.

[8] O. Salman, I. H. Elhajj, A. Kayssi, A. Chehab, "A Review on Machine Learning Based Approaches for Internet Traffic Classification," *Annals of Telecommunications - Annales des télécommunications*, vol. 75, no. 5, 2020, doi: 10.1007/s12243-020-00770-7.

[9] K. Takyi, A. Bagga and P. Goopta, "Clustering Techniques for Traffic Classification: A Comprehensive Review," *2018 7th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), Noida, India,* 2018, pp. 224-230, doi: 10.1109/ICRITO.2018.8748772.

[10] M. Jerbi, Z. C. Dagdia, S. Bechikh, and L. B. Said, "On the use of artificial malicious patterns for android malware detection," *Computers & Security*, 2020, pp. 315-320, doi: 10.1016/j.cose.2020.101743.

[11] K. S. Shim, Y. H. Goo, D. Lee, and M. S. Kim, "Automatic Payload Signature Generation for Accurate Identification of Internet Applications and Application Services," *KSII Transaction on Internet and Information Systems,* vol. 12, no. 04, pp. 1572-1593, Apr. 2018.

[12] A. Finamore, M. Mellia, and M. Meo, "Mining Unclassified Traffic using Automatic Clustering Techniques," *Proc. 2011 TMA International Workshop on Traffic Monitoring and Analysis*, 2011, pp. 150-163.

[13] P. Palumbo, L. Sayfullina, D. Komashinskiy, E. Eirola, and J. Karhunen, "A Pragmatic Android Malware Detection Procedure," *Computers and Security*, vol. 70, pp. 689-701, 2017.

[14] H. S. Galal, "Behavior-based Features Model for Malware Detection*," J Comput Virol Hacking Tech*, vol. 12, pp. 59-67, 2016.

[15] D. Bekerman, B. Shapira, L. Rokach, and A. Bar, "Unknown Malware Detection using Network Traffic Classification," *2015 IEEE Conference on Communications and Network Security (CNS),* 2015, pp. 134-142.

[16] N. Wu, Y. Qian, and G. Chen, "A Novel Approach to Trojan Horse Detection by Process Tracing," *In Proceedings of the 2006 IEEE International Conference on Networking, Sensing and Control (ICNSC'06), Ft. Lauderdale, Florida,* 2016, pp. 721-726, doi: 10.1109/ICNSC.2006.1673235.

[17] J. Zhang, C. Chen, Y. Xiang, W. Zhou, and A. V. Vasilakos, "An Effective Network Traffic Classification Method with Unknown Flow Detection," *IEEE Transactions on Network and Service Management*, vol. 10, no. 2, pp. 133-147, June 2013, doi: 10.1109/TNSM.2013.022713.120250.

[18] M. F. Umer, M. Sher, and Y. Bi, "Flow-based Intrusion Detection: Techniques and Challenges,*" Computers & Security,* vol. 70, pp. 238-254, 2017, doi: 10.1016/j.cose.2017.05.009.

[19] J. Brownlee, "How to Run Your First Classifier in Weka," 2014. [Online] Available at: https://machinelearningmastery.com/how-to-run-your-first-classifier-in-weka/

[20] E. Kidando, R. Moses, T. Sando, and E. E. Ozguven, "Assessment of factors associated with travel time reliability and prediction: an empirical analysis using probabilistic reasoning approach," *Transportation Planning and Technology,* vol. 42, pp. 309-323, 2019, doi: 10.1080/03081060.2019.1600239.

[21] 2017-08-01-Rig EK from the HookAds campaign sends Dreambot. [Online]. Available at: https://www.malware-traffic-analysis.net/2017/08/01/index.html, 2017.

[22] 2018-08-07-Hookads Rig EK pushes AZORult, AZORult pushes SmokeLoader. [Online]. Available at: https://www.malware-traffic-analysis.net/2018/08/07/index.html, 2018.

[23] PCAP files from the US National CyberWatch Mid-Atlantic Collegiate Cyber Defense Competition (MACCDC), [Online]. Available at: https://www.netresec.com/?page=MACCDC, 2012.

[24] CapLoader, "Handles Big Data PCAP files," [Online]. Available at: https://www.netresec.com/?page=CapLoader.

[25] H. Alaidaros and M. Mahmuddin, "Flow-Based Approach on Bro Intrusion Detection," *Journal of Telecommunication, Electronic and Computer Engineering*, vol. 9, pp. 139-145, 2017.

[26] I. R. Pérez I *et al.*, "A Bayesian Network approach to study the relationships between several neuromuscular performance measures and dynamic postural control in futsal players," *PLoS ONE*, vol. 14, no. 7, 2019, doi: 10.1371/journal.pone.0220065.

[27] B. Anderson and D. Mcgrew, "Machine Learning for Encrypted Malware Traffic Classification: Accounting for Noisy Labels and Non-Stationarity," *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD 17*, 2017, pp. 1723-1732.

[28] A. Wibawa, A. C. Kurniawan, D. M. P. Murti, and R. P. Adiperkasa, "Naïve Bayes Classifier for Journal Quartile Classification," *International Journal of Recent Contributions from Engineering, Science & IT (iJES)*, vol. 7, no. 2, pp. 91-99, 2019, doi: 10.3991/ijes.v7i2.10659.

[29] A. K. Mishra and B. K. Ratha, "Study of Random Tree and Random Forest Data Mining Algorithms for Microarray Data Analysis," *International Journal on Advanced Electrical and Computer Engineering (IJAECE),* vol. 3, no. 4, pp. 5-7, 2016.

[30] P. Mittal and N. S. Gill, "A Comparative Analysis Of Classification Techniques On Medical Data Sets," *IJRET: International Journal of Research in Engineering and Technology*, vol. 3, no. 6 pp. 454-460, 2014, doi: 10.5120/17314-7433.

[31] G. Kaur and A. Chhabra, "Improved J48 Classification Algorithm for the Prediction of Diabetes," *International Journal of Computer Applications*, vol. 98, no. 22, pp. 13-17, 2014, doi: 10.5120/17314-7433.

[32] J. Brownlee, "How to Estimate a Baseline Performance for Your Machine Learning Models in Weka," 2019. [Online]. Available at: https://machinelearningmastery.com/estimate-baseline-performance-machine-learning-models-weka/

[33] S. Sathyadevan and R. R. Nair, "Comparative Analysis of Decision Tree Algorithms: ID3, C4.5 and Random Forest. Computational Intelligence in Data Mining - Volume 1 Smart Innovation," *Systems and Technologies*, pp. 549-562, 2014, doi: 10.1007/978-81-322-2205-7_51.

[34] T. P. Fowdur, B. N. Baulum, and Y. Beeharry, "Performance analysis of network traffic capture tools and machine learning algorithms for the classification of applications, states and anomalies," *International Journal of Information Technology,* vol. 12, pp. 805-824, 2020, doi: 10.1007/s41870-020-00458-0.

[35] J. Brownlee, "A Gentle Introduction to k-fold Cross-Validation," 2020. [Online]. Available at: https://machinelearningmastery.com/k-fold-cross-validation/

[36] B. Daniel, "Cross Validation," *Encyclopedia of Bioinformatics and Computational Biology,* vol. 1, pp. 542-545, 2018, doi: 10.1016/B978-0-12-809633-8.20349-X.

BIOGRAPHIES OF AUTHORS

Ying Yenn Chan received the Bachelor's Degree in Electrical and Electronic Engineering from University Teknologi Malaysia, Malaysia in 2020.

Ismahani Bt Ismail received the Ph.D degree in Electrical and Electronic Engineering from University Teknologi Malaysia, Malaysia in 2013. She is currently a senior lecturer in University Teknologi Malaysia, Malaysia under Department of Electrical Electronic and Computer Engineering. She works in network algorithmics, digital system and FPGA implementation.

Ban Mohammed Khammas received the Ph.D degree in Electrical and Electronic Engineering from University Teknologi Malaysia, Malaysia in 2017. She is currently a lecturer in Collage of Information Engineering, Al-Nahrain University, Iraq under Department of Network Engineering. She works in neural network and FPGA implementation