IJAMML 6:2 (2017) 67-85 June 2017
ISSN: 2394-2258
Available at http://scientificadvances.co.in
DOI: http://dx.doi.org/10.18642/ijamml_7100121838

# ANDROID MALWARE DETECTION BASED ON MACHINE LEARNING ANALYSIS OF TRAFFIC FEATURES

**Fei Wu, Xiangqian Wu and Yuan Pei**

College of Information Science and Engineering, Xinjiang University, Urumqi 830046, P. R. China

## Abstract

Existing traffic flow research methods have some defects and shortcomings, which cannot accurately determine and analysis the traffic flow features. In many cases, the malwares which static analysis methods cannot accurately identify and detect them, such as the high degree of source code confusion or such use of dynamic code loading technology, we extracting the network traffic generated during the Internet connection. Then the information gain algorithm is used to select the discrimination features, after that we improved Naïve Bayes classifier with natural logarithm which could transfer multiplication into addition and Laplace calibration can also correct the result of calculation. Our experiments under ten-fold cross validation method, the results show that improved Naïve Bayes algorithm can both reduce the time complexity and achieve 93% accuracy, compare with the privilege based malware detection method, the improved Naïve Bayes classifier based on traffic features has better classification effect. To some extent, this method also provides a new way to detect android malwares, accurately.

*Keywords*: traffic features, Bayes model, android, malware.

*Corresponding author.
*E-mail address*: 308609778@qq.com (Fei Wu).


2010 Mathematics Subject Classification: 68-02.
Submitted by Bin Guo.
Received June 1, 2017; Revised June 7, 2017

## 1. Introduction

According to the recent report by the third-party authority statistics agency Statistics Portal, android operating system accounts for 86.2% of global market share, which ahead other operating system. So the android operating system can be called the current mainstream mobile operating system and still has great market potential. In addition, all current application software of android operating system almost need to connect to the Internet while they are working. In view of this, the android apps network access and traffic monitoring is becoming particularly important. Especially in android malware detection, network traffic characteristics can be used as an important features to distinguish between normal software and malware [25]. According to the official statistics of third party statistics, in the second quarter of 2016, Internet Security Center intercepted android new malicious process ordered samples above 4 million, the average daily interception of new smart phone malware samples nearly 47 thousand, a total of 61 million people infected with malware. The most statistically malicious behaviours are monitor the users and remote control, the average daily amount of malware infection reached 679 thousand passengers[1].

At present, android malware detection, in general, can be divided into two categories, one is the static analysis method, it detects malwares based on characteristic value. Static detection technology currently used in most security products, the main principle is scanning the target file and the feature values are extracted and matched with the existing features of the virus database, According to the matching algorithm, if the result exceeds the specified threshold value, it is judged to be malicious software after the similarity is calculated. But its disadvantages are also obvious, take up system resources, build a large feature library, and always report false positives; technology has the advantages of simple principle, easy to implement, for malicious software

[1]http://zt.360.cn/1101061855.php?dtid=1101061451&did=3701370185

the higher detection accuracy of low real-time requirement and prevent unknown malicious programs to a certain extent. The other is the dynamic analysis method based on heuristic detection technology, dynamic heuristic refers to the establishment of several feature points in the system to monitor software behaviour, and to determine whether it belongs to malicious software. Dynamic analysis method of resource consuming and high requirement for real-time, behaviours of software operation during the capture and recognition [2]. However, there are relatively few domestic and international researches on the traffic characteristics of android applications. Thereinto, Anshul Arora proposed an android malware identification method based on traffic flow characteristics, but the samples in they used in experiment are fewer, J48 decision tree algorithm can achieve higher accuracy but has certain limitations [4]. Wu et al. proposed a similarity flow about android packaged applications detection technology based on similarity calculation, by calculating the similarity of features to determine the application of the two package, it uses the five tuple method as the unique marking of the data stream marked. This method is mainly used to determine whether there is an advertising through the two package implantation [12]. Wang et. al., have constructed the network signature library for android malicious application by means of different feature, extraction, merging and clustering of malicious HTTP traffic. The validity and accuracy of the signature library have been proved [13].

Through the analysis and study of the above literature, an improved Naïve Bayes algorithm classification model is proposed by us for analyzing the flow characteristics of android application. The algorithm is improved mainly from 3 aspects:

(1) The introduction of natural logarithm multiplier method to modify the operation result underflow, and reduce the time complexity of the algorithm.

(2) The Laplace smoothing is introduced to modify the condition that the partition of the feature attribute does not appear to result in the overall operation result of 0.

(3) In the calculation of feature attribute partition refinement. For the discrete value of each division of statistical training samples in each category appear frequency. For continuous variables, we assume they obey Gauss distribution [6], calculate the mean and standard deviation of each category of this feature classification in the training samples, for the estimated value of difference substitution.

The improved algorithm can be used in network traffic classification and analysis of application software generated during runtime, the difference caused by comparing the normal flow of software and malicious software to realize the detection of malicious software. Undoubtedly, we proposed a new idea and method of android malware detection.

This paper focuses on the research of network behaviour of malicious software, such as perform remote control, disclosure of user privacy, and proposes an improved method based on Bayesian classification and analysis for network traffic on android system. Rest of the paper is organized as follows. In Section 2, we detailed describe and demonstration the improved algorithm and its criteria for evaluation. Section 3 show the data collection, experiment and test. Then, we analysis the experiment results and draw the conclusion in Section 4. Finally, we summarize what we have done in our research and related information.

## 2. Modified Bayesian Classifier Model

In this paper, we propose a method based on improved Naive Bayes [7, 11] to classify the network traffic of android system, and try to use it to detect malicious software. The first application of android produced during operation flow as features, using the information gain algorithm [14-16] to optimize feature, which could improve the operation efficiency, at the some time, the natural logarithm and Laplace smoothing method were introduced to improve Naive Bayesian algorithm.

### 2.1. Algorithm overview

Naive Bayesian algorithm [10, 17] is a combination of distributed function classification and attribute, to estimate the classification of samples through the joint probability calculation sample. Under normal circumstances: Assume that the sample space of random experiment $E$ is $S$, $A$ is an event and $P(A) > 0,$ here is the formula:

$$P(B_i|A) = \frac{P(B_i)P(A|B_i)}{\sum_{j=1}^{n} p(B_j)P(A|B_j)}, \; i = 1, 2, \ldots, n, \tag{1}$$

$P(B_i)$ is called prior probability, which can be obtained from the training data. $P(B_i|A)$ is called posterior probability, which means the probability of correcting the result. The application of Naive Bayesian to identify and classify network traffic, first need to collect traffic flow in a period of time, then divided into continuously varying lengths of time data flows, each data flow corresponding to one dimension. The classification mark of flow $F$ in the set $\{C_1, C_2, \ldots, C_m\}$. Assuming a random flow $F$, $F$ belongs to the class of conditional probability formula can be expressed as:

$$P(C_k|F) = \frac{P(F|C_k)P(C_k)}{P(F)}, \tag{2}$$

$P(C_k)$ is a priori probability, which can be obtained by calculating the ratio in the whole data flow. $P(F)$ is a constant, which represents the marginal probability of the data flow $F$. $P(F|C_k)$ is the conditional probability that the data flow $F$ belongs to the class $C_k$, which can be obtained from the training data. The features of the data flow $F$ can be represented by eigenvectors like $(f_1, f_2, \ldots, f_n)^T$, then $P(F|C_k)$ can be represented:

$$P(F|C_k) = P(f_1, f_2, \ldots, f_n|C_k). \tag{3}$$

The Naive Bayes model assumes that all variables are useful for

classification, since all attributes have been exported:

$$P(f_1, f_2, \ldots, f_n|C_k) = \prod_{j=1}^{n} P(f_j|C_k). \tag{4}$$

Taking formula (3), (4) into the formula (2), we will get follow formula:

$$P(C_k|f_1, f_2, \ldots, f_n) = \frac{P(C_k)\prod_{j=1}^{n} P(f_j|C_k)}{\sum_{k=1}^{m} P(C_k)\prod_{j=1}^{n} P(f_j|C_k)}. \tag{5}$$

$X$ is defined as a random variable in the input space, $Y$ is defined as a random variable in the output space. The Naive Bayes classifier can be expressed as:

$$y = \arg\max_{C_k} \frac{P(Y = C_k)\prod_{j=1}^{n} P(X_j = f_j|Y = C_k)}{\sum_{k=1}^{m} P(Y = C_k)\prod_{j=1}^{n} P(X_j = f_j|Y = C_k)}.$$

### 2.2. Improved algorithm

Naive Bayesian traditional algorithms have some drawbacks and due to the complexity of traffic characteristics cannot be used directly, through the improved algorithm not only improve the efficiency but also can be applied to traffic classification algorithm is as follows:

**Figure 1.** The improved algorithms.

**Algorithm 1.** Improved Naïve Bayesian algorithms

**Input:** Given a training set $T$ and $T = \{(X_1, Y_1), (X_2, Y_2), \ldots, (X_N, Y_N)\}$ and $X_i = (x_i^{(1)}, x_i^{(2)}, \ldots, x_i^{(n)})^T$, the $Y_i$ is the label of $X_i$, $x_i^{(j)}$ means $i$-th sample $j$-th features $x_i^{(j)} \in \{a_{j1}, a_{j2}, \ldots, a_{js}\}$, $a_{jm}$ means $m$-th values which $j$-th features could get in its maximum span, thereinto $j = 1, 2, \ldots, n$, $m = j_1, j_2, \ldots, j_s$, and the $y_i \in \{C_1, C_2, \ldots, C_K\}$; training set $T' = \{X_1', X_2', \ldots, X_l'\}$.

**Onput:** The classification of $T'$

(1) preprocessing:

(i) Calculate the information gain values for each feature and rank them in descending order.

$$gain\left(x^{(j)}\right) = info(C_k) - info_{x^{(j)}}(C_k).$$

(ii) If the characteristic item does not appear, numerical correction using Laplace calibration.

$$\varphi_j = \frac{\sum_{i=1}^{N} I\left(x^{(j)} = a_{ji},\ y_i = C_k\right) + \mu}{\sum_{i=1}^{N} I(y_i = C_k) + m^{\mu}}.$$

(iii) In multiplication, the natural logarithm method is adopted to correct the result.

(2) Calculate the prior probability and conditional probability

$$P(Y = C_K) = \ln \frac{\prod_{i=1}^{N} I(y_i = C_K)}{N},\ k = 1, 2, \ldots, K.$$

(i) When the characteristic attributes are discrete random variables.

$$P\left(X^{(j)} = a_{ji} | Y = C_k\right) = \frac{\sum_{i=1}^{N} I\left(x^{(j)} = a_{ji},\ y_i = C_k\right)}{\sum_{i=1}^{N} I(y_i = C_K)}.$$

(ii) When the characteristic attributes are continuous random variables, the Gauss distribution is introduced and thus computed estimated value $g\left(a_{ji},\ \eta_{C_k},\ \sigma_{C_k}\right)$.

$$P\left(X^{(j)} = a_{ji} | Y = C_k\right) = \frac{1}{\sqrt{2\pi\sigma_{C_k}^2}} e^{\frac{(\nu-\mu_c)^2}{2\sigma_{C_k}^2}}.$$

(3) Simply, for a newly given instance $X_1' = (x_1^{(1)}, x_1^{(2)}, \ldots, x_1^{(n)})^T$, compute following:

$$\ln P(Y = C_k)\prod_{j=1}^{n} P(X^{(j)} = x^{(j)}|Y = C_k),\ k = 1, 2, \ldots, K.$$

(4) Determine the classification of instances $X_i'$, computing the maximum probability of each category.

$$y = \arg\max_{C_k} \ln P(Y = C_k)\prod_{j=1}^{n} P(X^{(j)} = x^{(j)}|Y = C_k).$$

## 2.3. Algorithm evaluation

The evaluation criteria of the Naive Bayes classifier using the receiver operating characteristic curve (ROC), accuracy (ACC), and area under the curve (AUC).

The accuracy is the standard to reflect the performance of the classifier, and its definition is simple, the positive samples were predicted as positive samples and negative samples were predicted as negative samples, and the ACC = ((TP + TN)) / ((P + N)). True positive (TP) means positive samples were predicted to be positive samples by model, true negative (TN) means positive samples predicted to be negative samples by model, false positive (FP) means negative samples predicted to be positive samples by model, false negative (FN) means negative samples predicted to be negative samples by model.

ROC curve [21, 22] is the standard to reflect the quality of the classifier, the true positive rate (TPR) is Y axis, the false positive rate (FPR) is X axis, if the curve is closer to the upper left corner, the better the classifier performance.

AUC is an auxiliary parameter to determine the performance of the classifier, that is, when the ROC curve cannot accurately and clearly reflect the classification effect of multiple classifiers, then the AUC value comparison, the greater the performance of the classifier.

## 3. Experiment and Test

### 3.1. Sample selection

Each android sample traffic as the characteristic attribute, the improved Naive Bayes algorithm is used to classify the traffic to find the difference between the flow characteristics of the malware and the normal software. The normal software come from Google play covers all application categories, the malware come from Android Malware Genome Project [3], https://virusshare.com [24], include 27 malware families that connects the Internet during the runtime and generates data traffic. However, it is proved that only 13 of them are effective, as Table 1 (Appendix). The reason for this may be that the remote server has been shut down or used for other purposes [5].

### 3.2. Data collection

In an Internet connected host running android simulator, and the virtual device on the IP is set to the same network to ensure the application of the simulator can access the network, after the installation of a malicious software in the simulator, and to keep it running for 24 hours, as shown in Figure 2. When the malicious software running in the simulator, in ADB mode and the call to tcpdump [18, 19] capture network traffic, script file as shown in Figure 3 the exported pcap file [20] using Wireshark analysis software, the data collection process is normal too.

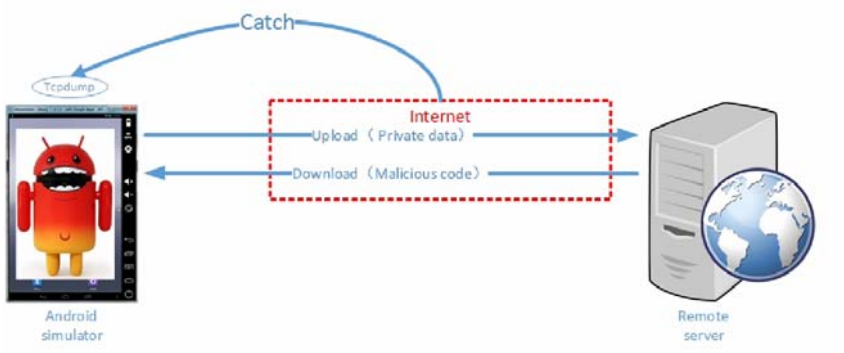


**Figure 2.** Experimental deployment settings.

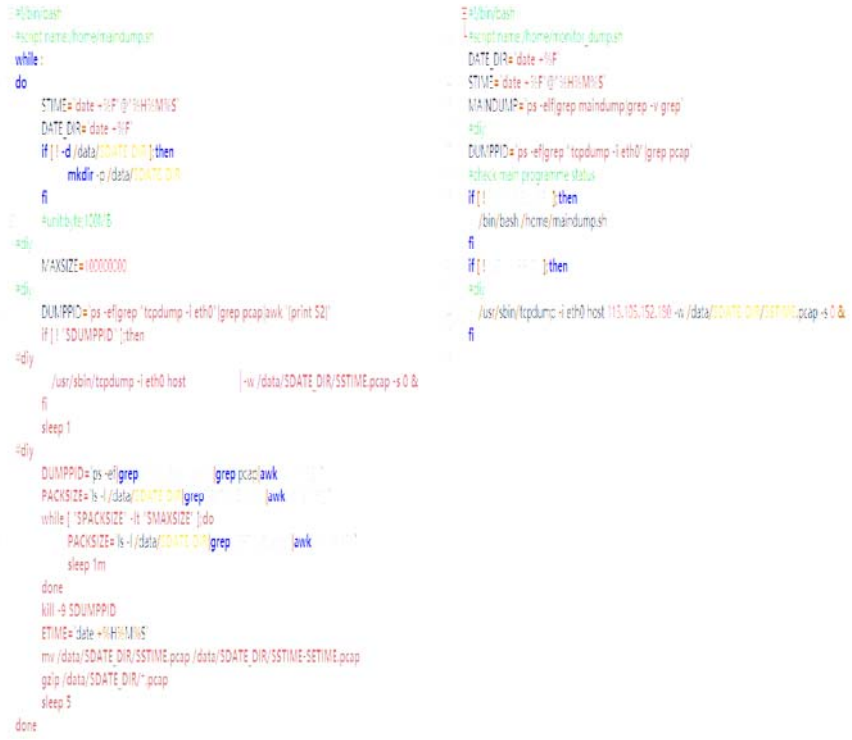

**Figure 3.** Script file.

### 3.3. Feature selection

We choose information gain (IG) feature selection algorithm [8], the calculation of each feature in the classifier in the information gain, the bigger the IG value is more important. The characteristics of flow characteristics of IG value of the 20 features selected in descending order as shown in Table 2 (Appendix).

### 3.4. Classification model

The classification model is divided into 3 stages, as shown in Figure 4. The first stage is the preparation stage of the experiment data, a total of 702 samples were collected, of which 344 malware samples, 358 normal samples. The whole experiment using ten-fold cross validation method, and the experiments were repeated averaging. The process of data

collection is described in 3.2, and the IG feature selection algorithm is used to extract the feature set $\alpha$, as classifier input. The second stage is the core part of classifier training phase, training samples are calculated in the prior probability and conditional probability, the results will directly determine the classifier training process, we properly adjusting the sample distribution, to get the best results. The third stage classifier testing stage, the core part is to classify the test samples and to evaluate the effect of the classifier. In the improved Bayesian model classification method, it is necessary to calculate the prior probability of the sample in the category division and the conditional probability of each feature of the sample in each category. In the probability calculation, with the multiplier, taking the natural logarithm and the Laplasse calibration, it can make the result more accurate and distinguish the malicious software and the normal software more effectively. Finally, the prediction results are compared with the real categories, and the classification results of the classifier are evaluated objectively by using ACC, ROC, and AUC.

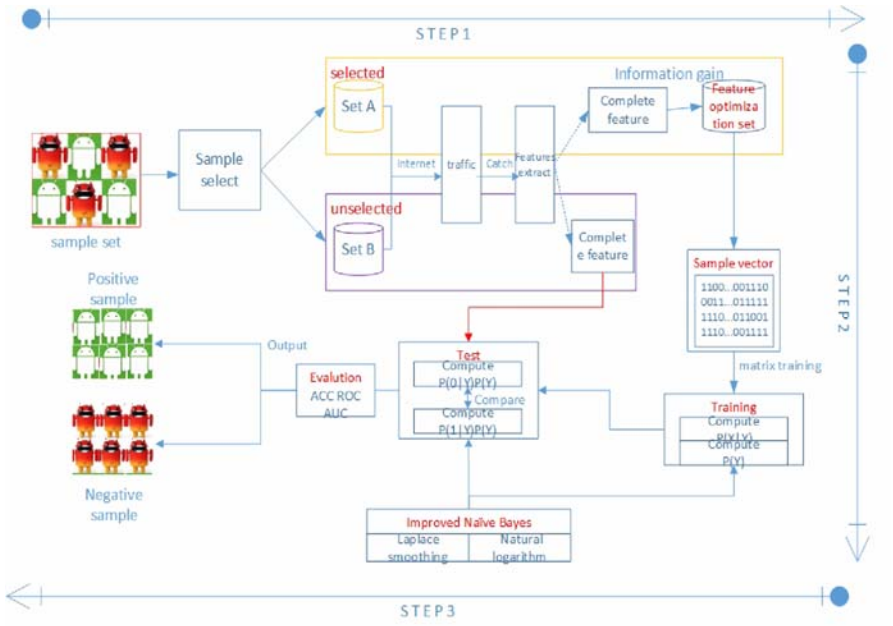


**Figure 4.** Improved naive bayes model.

## 4. Result Analysis and Conclusion

In the experiment, the author found that the malware is receiving and sending data periodically, such as AnserverBot every two hours to access a remote server once every 5 seconds, Plankton to the remote server to send a data to the remote server and receive instructions. In a total of 20 traffic characteristics, there are 6 features performance greater differentiation in malware and normal software. They are malicious domain, the receive / send byte ratio, the number of bytes received per second on average, the average number of sent packets containing data streams, each received data stream contains the average number of packets the average number of bytes received, each data stream, as shown in Table 3 (Appendix).

The IG algorithm is used to select the features of larger information gain as the input of the classifier. The ROC, AUC comparison and accuracy of different features are shown in Figure 5 and Figure 6.

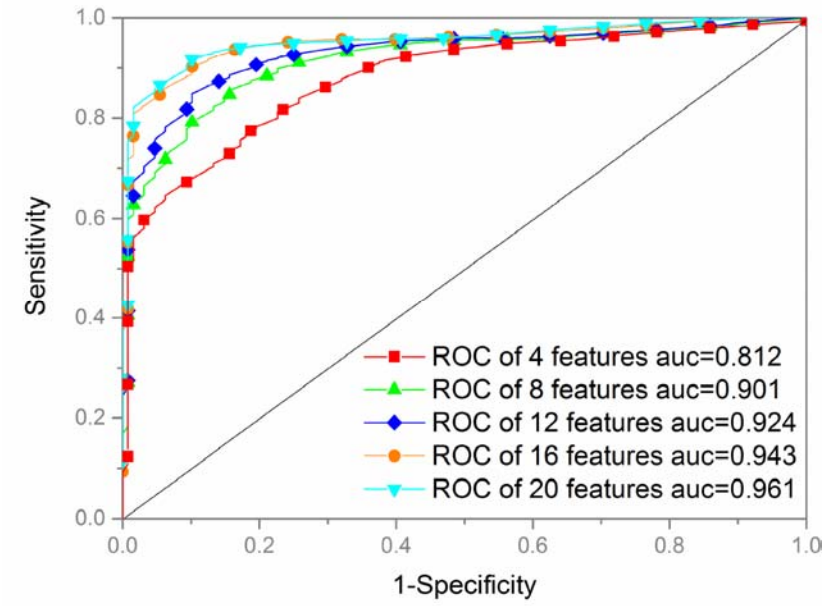


**Figure 5.** ROC curves of different features.

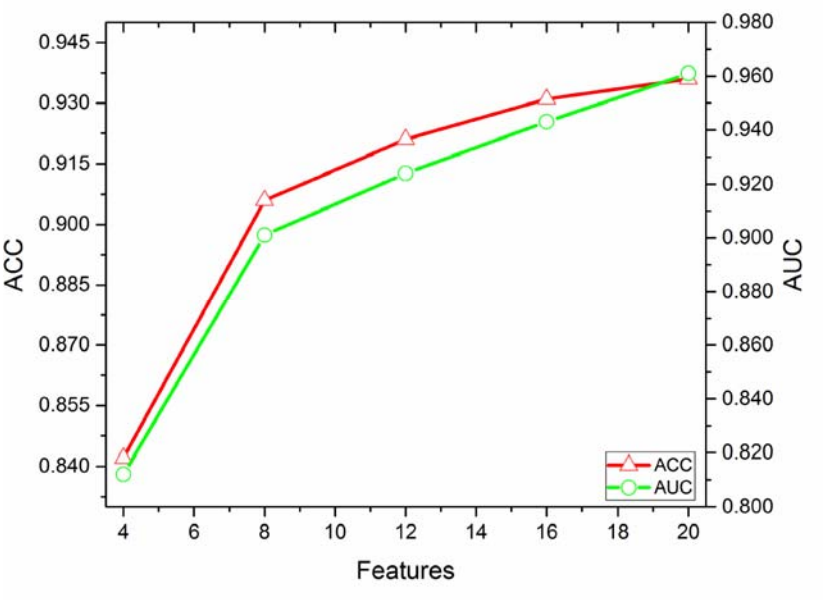


**Figure 6.** The accuracy of different features.

The detail experimental data of our method are shown in Table 4.

**Table 4.** Accuracy and AUC

| Traffic | features | 4 | 8 | 12 | 16 | 20 |
|---|---|---|---|---|---|---|
| Features | ACC | 0.842 | 0.906 | 0.921 | 0.931 | 0.936 |
| | AUC | 0.812 | 0.901 | 0.924 | 0.943 | 0.961 |

The following conclusions can be drawn from the above experimental results and comparisons:

(1) Based on the analysis of traffic characteristics in malware detection devices powered by android operating system is feasible, and our method can achieve high accuracy.

(2) In information theory [23], the expected information is little, the information gain is greater, the higher the purity. It can be seen from Figure 5, when the feature number reaches a certain amount, the accuracy rate of AUC and tends to be stable, that features selected by the

IG algorithm can achieve the purpose of denoising in a certain extent, abandon the characteristics which impact the classification results or have negative effect, can achieve better classification results.

(3) Our method has some limitations, APP which to be detected need to connect to the Internet and collect network traffic information for a period of time.

In this paper, according to the traffic characteristics of normal app and malicious app, then information gain algorithm and the improved Naive Bayes algorithm are used to establish the android malware classifier based on traffic characteristics. The experimental result prove that it has better performance. Our work provides new ideas and methods for the static detection cannot accurately identify and detect. Subsequent research work can be combined with static detection, the detection rate of dynamic code loading and source code highly obfuscated malware. In the feature, we will increase the malicious application sample database, and try to find more discriminative features.

## Acknowledgements

This research has been fully supported by the National Natural Science Fund of China (61303231). We are equally thankful to my institute Xinjiang University and my colleagues for their invaluable inputs, comments, and suggestions to improve this paper.

## References

[1] Guojun Peng, Jingwen Li, Runkang Sun and Yunchang Xiao, Android malware detection research and development, J. Wuhan Univ. (Nat. Sci. Ed.) (2015), 21-33.

[2] X. Jiang and Y. Zhou, Dissecting android malware characterization and evolution, IEEE Symposium on Symposium Security & Privacy, IEEE (2012), 95-109.

[3] A. Arora, S. Garg and S. K. Peddoju, Malware detection using network traffic analysis in android based mobile devices, Eighth International Conference on Next Generation Mobile Apps Services and Technologies (2014), 66-71.

[4] X. Jiang and Y. Zhou, A Survey of Android Malware, Springer, New York, (2013), 3-20.

[5] V. Y. Zaburdaev, Gaussian Signals, Correlation Matrices and Sample Path Properties, A First Course in Statistics for Signal Analysis, Birkhäuser, Boston, (2006), 159-174.

[6] Yanping Xu, Chunhua Wu, Meijia Hou, Kangfeng Zheng and Shan Yao, Android malware detection technology based on improved Naïve Bayesian, Journal of Beijing University of Posts and Telecommunications (2016), 43-47.

[7] A. Feizollah, N. B. Anuar, R. Salleh et al., A review on feature selection in mobile malware detection, Digital Investigation the International Journal of Digital Forensics & Incident Response 13(C) (2015), 22-37.

[8] P. Harrington, Machine Learning in Action, Manning Publications Co., 2012.

[9] Hui Wang, Hongyu Chen and Shufen Liu, Intrusion detection system based on improved Naïve Bayesian algorithm, Computer Science 41(4) (2014), 111-115.

[10] Hang Li, Statistical Learning Method, Tsinghua University Press, 2012.

[11] M. Chandramohan and H. B. K. Tan, Detection of mobile malware in the wild, Computer 45(9) (2012), 65-71.

[12] Fangping Wu, Dafang Zhang, Xin Su and Xia-an Bi, Based-on network traffic's similarity to detect android repackaged application, Journal of Chinese Computer Systems 36(5) (2015), 954-958.

[13] Xiaofei Wang, Research on Android Malicious Application Communication Mechanism and Traffic Feature Extraction Based on HTTP, Hunan University, 2014.

[14] C. Lee and G. G. Lee, Information gain and divergence-based feature selection for machine learning-based text categorization⋆, Information Processing & Management 42(1) (2006), 155-165.

[15] A. G. Karegowda, A. S. Manjunath and M. A. Jayaram, Comparative study of attribute selection using gain ratio and correlation based feature selection, International Journal of Information Technology and Knowledge Management 2(2) (2010), 271-277.

[16] H. Liu and L. Yu, Toward integrating feature selection algorithms for classification and clustering, IEEE Transactions on Knowledge and Data Engineering 17(4) (2005), 491-502.

[17] I. Rish, An empirical study of the Naïve Bayes classifier, IJCAI 2001 workshop on empirical methods in artificial intelligence, IBM, New York, 3(22) (2001), 41-46.

[18] F. Fuentes and D. C. Kar, Ethereal vs. Tcpdump: A comparative study on packet sniffing tools for educational purpose, Journal of Computing Sciences in Colleges 20(4) (2005), 169-176.

[19] V. Jacobson, C. Leres and S. McCanne, The tcpdump manual page, Lawrence Berkeley Laboratory, Berkeley, CA, (1989), 143.

[20] V. Jacobson, C. Leres and S. McCanne, pcap-Packet Capture Library, UNIX Man Page, 2001.

[21] J. Davis and M. Goadrich, The relationship between Precision-Recall and ROC curves, Proceedings of the 23rd International Conference on Machine Learning, ACM (2006), 233-240.

[22] J. Fan, S. Upadhye and A. Worster, Understanding receiver operating characteristic (ROC) curves, CJEM 8(01) (2006), 19-20.

[23] D. J. C. MacKay, Information Theory, Inference and Learning Algorithms, Cambridge University Press, 2003.

[24] VirusShare, Virusshare.com-because sharing is caring, http://virusshare.com/, 2013. Accessed Dec. 10, 2013.

[25] W. Enck, D. Octeau, P. McDaniel et al., A study of android application security, USENIX Security Symposium 2 (2011), 2.

■

## Appendix

The Tables 1, 2, and 3 are large tables, so we put them in Appendix.

**Table 1.** Malware families and their function

| Malware | Features | Malware | Features |
|---|---|---|---|
| AnserverBot | High degree of confusion of the source code, dynamic loading malicious code | BaseBridge | Send and read text messages and make security software stop working |
| BgServ | Collect personal information, such as IMEI, mobile phone number, etc. | DroidDream | Collect and upload IMEI, IMSI, phone number, etc., similar to BgServ |
| DroidDreamLight | DroidDream variants that enhance the concealment of malicious behaviour | DroidKungFu1 | Collect and upload IMEI, phone model, android version number, etc. |
| DroidKungFu2 | Variant of DroidKung, added remote command and control server | DroidKungFu3 | A high degree of confusion of the source code, and hide and encrypt all the native binaries associated with malware |
| GoldDream | Collect and upload IMEI and IMSI to the specified server | jSMSHider | Collect and upload IMSI and phone numbers to the specified server |
| Pjapps | Collect and upload IMEI, device number, SIM card serial number to the specified server | Plankton | Infect other benign APP while uploading device ID and access permission information |
| RogueLemon | Send text messages without the user's knowledge and can receive remote server control commands | | |

**Table 2.** IG values of network traffic characteristics

| Features | IG | Features | IG |
|---|---|---|---|
| Malicious domain | 0.306 | Packet encryption percent (%) | 0.177 |
| Average send bytes per second | 0.298 | Number of open ports | 0.165 |
| Receive / send byte ratio | 0.292 | Adjacent packet time interval(receive) | 0.152 |
| Average data packet in a data stream | 0.287 | Average number of bytes per second | 0.141 |
| Average data packets in a data stream | 0.269 | Number of connections / target IP number ratio | 0.132 |
| Average number of bytes received | 0.255 | Average duration of data flow | 0.116 |
| Average number of bytes sent data stream | 0.226 | Adjacent packet time interval(send) | 0.102 |
| Average packet size | 0.202 | Average number of packets per second | 0.081 |
| Receive / send packet ratio | 0.195 | Average number of packets received per second | 0.063 |
| The size of data stream header | 0.182 | Flow composition (TCP,%) | 0.045 |

**Table 3.** Feature complete statistics (*means obviously distinguishing features)

| Features | Malware | Normal | Features | Malware | Normal |
|---|---|---|---|---|---|
| * Receive and send ratio (Bytes) | [0.15,3.66] | [5.5,8.1] | Transmission time interval | [1,120] | [0.5,1] |
| * Average number of bytes received per second | [35,1200] | [12000,18000] | Average duration of data flow (S) | [0.5,60] | [5,15] |
| * Average number of packets in a data stream | [4,7] | [21,55] | The average number of bytes sent to the data stream | [500,1800] | [2500,4200] |
| * Average packet size in data stream | [2,6] | [27,60] | Number of connections / target IP number ratio | [1,8] | [2,5] |
| * Average number of bytes received data stream | [800,5000] | [24000,100000] | Receive and send ratio (Packets) | [0.76,1] | [1,6] |
| Receiving time interval | [1,116] | [0.5,30] | Average number of bytes per second | [15,4560] | [5,2500] |
| Average packet size (Bytes) | [86,360] | [220,900] | Flow composition (TCP,%) | [86.26,91.23] | [90.12,92.25] |
| Average number of packets per second | [1,24] | [12,19] | Average number of packets received per second | [1, 24] | [16, 25] |
| The size of data stream header (Byte) | [8,60] | [8,36] | Number of open ports | [1,12] | [1,8] |
| * Malicious domain | [2,5] | [0,0] | Packet encryption percent (%) | [0,0.15] | [0.25,0.90] |