icapie paper.pdf
May 13, 2021
2910 words / 15597 characters

A K

icapie paper.pdf

Sources Overview

10%

OVERALL SIMILARITY

1	www.tandfonline.com INTERNET	1%
2	epdf.pub INTERNET	1%
3	www.hindawi.com INTERNET	<1%
4	jjcit.org INTERNET	<1%
5	"A Hybrid Model for Android Malware Detection", International Journal of Innovative Technology and Exploring Engineering, 2019 CROSSREF	<1%
6	www.intechopen.com INTERNET	<1%
7	www.neti.gatech.edu INTERNET	<1%
8	www.pure.ed.ac.uk INTERNET	<1%
9	"Algorithms and Architectures for Parallel Processing", Springer Science and Business Media LLC, 2018 CROSSREF	<1%
10	Wenjuan Wang, Xuehui Du, Na Wang. "Building a Cloud IDS Using an Efficient FeatureSelection Method and SVM", IEEE Access, 2018 CROSSREF	<1%
11	Yue Gao Tang, Li Miao, Feng Ping Chen. "Peer-Comparison Based Fault Diagnosis for Hadoop Systems", Applied Mechanics and Mate... CROSSREF	<1%
12	deepai.org INTERNET	<1%
13	"Innovations in Smart Cities Applications Edition 3", Springer Science and Business Media LLC, 2020 CROSSREF	<1%
14	"Malware Analysis Using Artificial Intelligence and Deep Learning", Springer Science and Business Media LLC, 2021 CROSSREF	<1%
15	res.mdpi.com INTERNET	<1%
16	www.ijcea.com INTERNET	<1%

Excluded search repositories:

- None

Excluded from Similarity Report:

- Bibliography
- Small Matches (less than 8 words).

Excluded sources:

- None

Mutual Information-Based Feature Ranking Method for Benign and Malicious Android Network Traffic

Kapil Gupta[1,2], Deepti Singh[1] and Ankur Singh[1]

[1] Department of Applied Mathematics, Delhi Technological University, Delhi, India

[2] Corresponding author: Kapil Gupta; kapilgupta1350@gmail.com

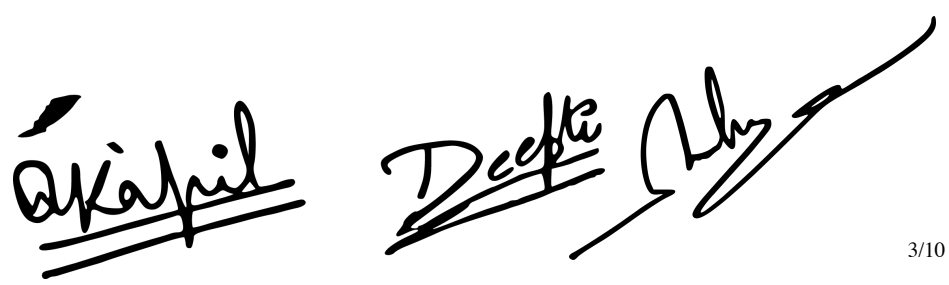
Abstract. Mobile malware can steal private information and cause monetary loss, they can also make the system collapse. The increasing use of Android smartphones has made it a major target of attackers. Around 99% of malware developed targets Android. Despite a large number of studies on the detection of Android malware, most are unable to achieve a high level of accuracy to detect malware. Hence, in this paper, a method to analyze Android traffic features is proposed to select relevant and important features; which in turn can be used by existing models or new models to enhance the detection of Android malware.

Keywords: Mobile Network, Mobile Security, Feature Selection, Malware Detection

1 Introduction

According to recent statistics [1], there are around 5 billion mobile users worldwide out of which 3.8 billion have smartphones, which is almost three-quarters of the number of mobile users. In 2016, there were only 2.5 billion smartphone users. From these figures, we can say that the number of smartphone users increased by 52% from 2016-21. The increasing popularity of smartphones is due to the fact that it provides a simple and convenient route to connect with the internet and individuals. There are various operating systems available in the market for smartphones and one of these is Android. Android OS was first launched in 2008, and as of 2021, it has a market share of 72.2% [2] making it the most popular mobile OS.

The increasing popularity of smartphones has caught the attention of attackers as well. As of 2020, 5.6 million malicious packages have been detected on various mobile devices [3]. One example of such malware is Droiddream [4], which enters the system through the installation of some third-party software available on the Android play store. The malware is capable of downloading several more malicious applications without the knowledge of the user and also grants access to hackers to control the device. The example highlights the vulnerability of the Android security system in dealing with malware. Trojans, keyloggers, spyware, adware are some of the common malware that can infect the device and can steal private information, and cause monetary loss.



1.1 Motivation

According to the article in [5], around 99% of mobile malware developed targets Android devices. The main reasons for the high attacks on Android are the open structure of Android which allows hackers to find and exploit security gaps in the system and the availability of a large number of third-party applications on the system. Because of these system vulnerabilities and threats imposed by malware we need a stealthier mechanism to detect these malware. Several detection methods have been proposed in this area. Some of these methods use static detection, like in [15] [33], in which static features like app permissions, manifest files, and source code are analyzed. However, this type of detection fails when the application is capable of downloading malicious updates in runtime, as these methods do not execute the application. Hence, dynamic detection is preferred over static detection. In dynamic detection, system calls and network traffic features are used. However, some studies suggest that system calls don't give high accuracy in case of Android malware detection. Hence in this paper, we aim to analyze Android traffic features and propose a feature selection method that can be used to develop a promising detection model.

1.2 Contributions

The key contribution of this paper is as follows:

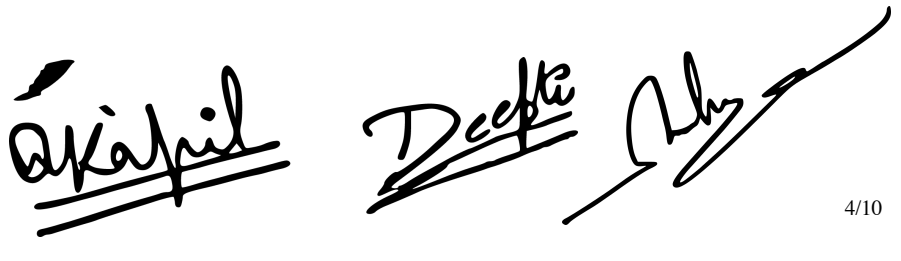
1) We collected the network traffic files for both normal and malicious applications with the help of Wireshark software.
2) After the data collection, we preprocessed the data to obtain 21 different network traffic features.
3) After the preparation of the dataset, we applied a mutual information-based feature ranking algorithm that returns a ranked list of selected features.

1.3 Organization

The remaining content of the paper is structured as follows. Section 2 describes the related work done in this area, and in section 3 we discuss the proposed methodology. Results are discussed in section 4 and conclude with future directions in section 5.

2 Related Work

Several studies have been carried out to analyse Android traffic features which can be used to make a promising detection model. Here, we analyse similar works and studies which have already been carried out / published similar to our proposed work. The authors in [6] used Analysis of Variance (X-ANOVA) to rank features. The authors in [7] and [8] proposed ranking of features based on the most often used APIs. The authors in [9] showed that the distance between a function vector and its associated background vector can be used to rank features. Singh Et Al. [10] using three feature ranking techniques from a collection of 337 attributes. Kavitha Et Al. [11] proposed



security ranking algorithms; unique to all apps. The authors in [12] used ranking of features extracted from Android OS to finally detect Android malware. Yerima Et Al. [13] proposed a multilevel architecture model; using a set of different ranking algorithms at higher level. Gou Et Al. [14] proposed usage of association statistics to rate the danger of an Android app. The authors in [15] and [16] proposed Information Gain for ranking intents and permissions. Fatima Et Al. [17] proposed the usage of Genetic algorithm for ranking and selection of features. The authors in [18] proposed a Linux kernel-based ranking and selection algorithm. The authors in [19] proposed Principal Component Analysis technique for the same. The authors in [20], [21], [22] and [23] proposed various machine learning tools for ranking and selection of features. Qiao Et Al. [24] proposed merging permissions and API features for ranking. Ma Et Al. [25] proposed a knowledge graph-based sensitive feature selection method. Jung Et Al. [26] proposed the usage of popularity and relations between APIs for the purpose of feature selection and ranking. The authors in [29-32] have used network traffic features for Android malware detection.

3 Methodology

In this section, we have discussed the proposed feature selection algorithm. This section can be further divided into three main subsections, 1) Data Collection, 2) Data preprocessing, 3) Feature Selection.

3.1 Data Collection

Data collection is the first step in feature analysis, here we collected network traffic files for normal and malicious applications through a third-party packet capturing application. For malware data, we used some known malicious applications and for the normal dataset, we used some popular mainstream applications available on play store. The traffic files were obtained in pcap form and we extracted the features in a CSV file with the help of Wireshark software. The list of extracted features thus obtained is summarised in Table 1.

Table 1. List of extracted features

Notation	Extracted Features
f_1	Total Packet transferred
f_2	Total Bytes transferred
f_3	Packets Sent
f_4	Bytes sent
f_5	Packets received
f_6	Bytes received
f_7	Rel Start

f_8	Duration
f_9	Bits sent per second
f_{10}	Bits received per second
f_{11}	Bytes transferred per second
f_{12}	Packets transmitted per second
f_{13}	Packets sent per second
f_{14}	Packets received per second
f_{15}	Average packet size
f_{16}	Average packet size sent
f_{17}	Average packet size Received
f_{18}	Ratio of incoming to outgoing packets
f_{19}	Ratio of incoming to outgoing Bytes
f_{20}	Average time between a packet sent
f_{21}	Average time between a packet received

3.2 Data Preprocessing

In this step, we normalize the dataset using equation (1) so as to scale all the data on one common scale of 0-1.

$$Z = (X - \mu)/\sigma \qquad (1)$$

Here μ represents the mean of the data, and σ represents the standard deviation of the data. So for each feature, we calculated the mean and standard deviation and then transformed each data point into its normalized value.

3.3 Feature Ranking

Here we have used a feature ranking algorithm based on mutual information to rank and select relevant features that can be used for malware detection. We use mutual information as it's a good estimator to determine the dependency of features. We used a library function included in [27] to calculate mutual information, the function uses the k nearest neighbor algorithm based on entropy to estimate mutual information. For feature selection, we employ the algorithm proposed by Ambusaidi et al. [28]. In this algorithm, we select and rank features based on their GMI score which is calculated using equation (2).

$$G_{MI}(f_i) = MI(C, f_i) - MR \qquad (2)$$

Here S is the subset of selected features f_s and f_i is the feature from our original feature set. MI(C, Fi) represents the mutual information carried by feature f_i around the target class variable C. Here C can have two values 0 and 1, the former indicating

normal data point and the latter indicating malware. MR or mutual redundancy can be calculated using equation (3)

$$MR(fi) = \frac{1}{|S|}\sum_{fs}\frac{\mathrm{MI(fi,fs)}}{\mathrm{MI(C,fi)}} \qquad (3)$$

The G_{MI} values can be either positive or negative, a positive value indicates that the feature is relevant and important while the latter indicates that the feature fails to provide any important information for classification.

To obtain the set of selected features, we first started with an empty set S, which will contain the selected ranked features and our original feature set F as input. We calculated the mutual information about C for each feature in F and picked the feature carrying maximum mutual information and inserted it into the set S and removed it from F. We Calculated G_{MI} scores using equation (2) for all features present in the feature set and picked and removed the feature having the highest G_{MI} score and inserted in S only if the G_{MI} score was greater than zero. The above step was repeated till set F became empty. We finally sorted the features in S based on their G_{MI} values.

4 Results

We used the feature selection method discussed in the previous section and obtained a set of 18 features which are listed in Table 2. The features are ranked on the basis of their GMI values in ascending order.

Table 2. List of selected features ranked on the basis of their G_{MI} values in ascending order

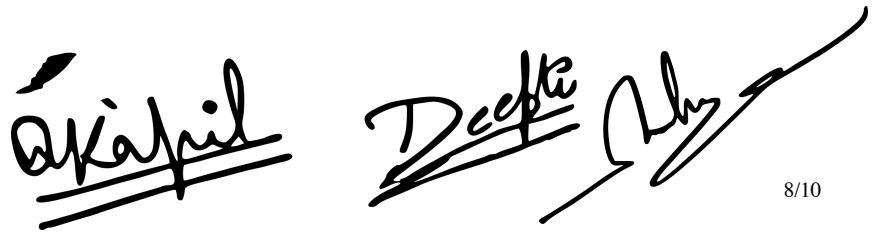
Selected features
$f_5, f_3, f_{21}, f_{16}, f_{17}, f_{18}, f_{19}, f_1, f_4, f_{15}, f_2, f_{14}, f_{13}, f_8, f_6,, f_7, f_9, f_{20}$

5 Conclusion and Future Work

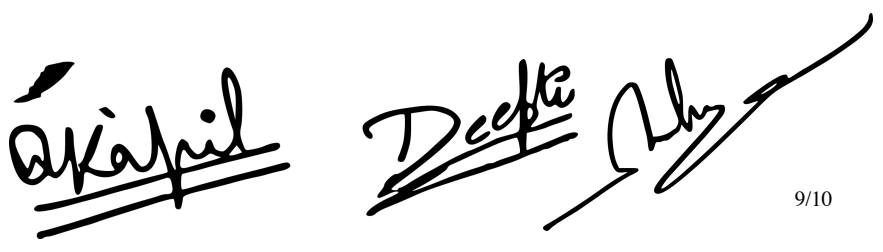
In this paper, we first collected data for malware and normal Android samples and normalized it. Then, we proposed a feature selection algorithm based on mutual information that selects and ranks the relevant features required for malware detection. The results with mutual information concept helped us to identify the top ranked features out of a set of all 21 features. In our future work, we will use this set of selected features to propose a network intrusion detection that can detect Android malware dynamically.

6 References

1. How Many People Have Smartphones Worldwide (May 2021) (Source:https://www.bankmycell.com/blog/how-many-phones-are-in-the-world)
2. Mobile Operating System Market Share Worldwide | StatCounter Global Stats (Source: https://gs.statcounter.com/os-market-share/mobile/worldwide)
3. Mobile malware evolution 2020 | Securelist (Source: https://securelist.com/mobile-malware-evolution-2020/101029/)
4. What is Droiddream? | Webopedia (Source: https://www.webopedia.com/definitions/droiddream/)
5. Another Reason 99% of Mobile Malware Targets Androids - F-Secure Blog (Source: https://blog.f-secure.com/another-reason-99-percent-of-mobile-malware-targets-androids/)
6. R. Raphael, Vinod P. and B. Omman, "X-ANOVA ranked features for Android malware analysis," 2014 Annual IEEE India Conference (INDICON), 2014, pp. 1-6, doi: 10.1109/INDICON.2014.7030646.
7. J. Jung, K. Lim, B. Kim, S. -j. Cho, S. Han and K. Suh, "Detecting Malicious Android Apps using the Popularity and Relations of APIs," 2019 IEEE Second International Conference on Artificial Intelligence and Knowledge Engineering (AIKE), 2019, pp. 309-312, doi: 10.1109/AIKE.2019.00062.
8. J. Jung et al., "Android Malware Detection Based on Useful API Calls and Machine Learning," 2018 IEEE First International Conference on Artificial Intelligence and Knowledge Engineering (AIKE), 2018, pp. 175-178, doi: 10.1109/AIKE.2018.00041.
9. M. Yousefi-Azar, L. Hamey, V. Varadharajan and S. Chen, "Byte2vec: Malware Representation and Feature Selection for Android," in The Computer Journal, vol. 63, no. 1, pp. 1125-1138, Jan. 2020, doi: 10.1093/comjnl/bxz121.
10. L. Singh and M. Hofmann, "Dynamic behavior analysis of android applications for malware detection," 2017 International Conference on Intelligent Communication and Computational Techniques (ICCT), 2017, pp. 1-7, doi: 10.1109/INTELCCT.2017.8324010.
11. K. Kavitha, P. Salini and V. Ilamathy, "Exploring the malicious android applications and reducing risk using static analysis," 2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT), 2016, pp. 1316-1319, doi: 10.1109/ICEEOT.2016.7754896.
12. S. Aonzo, A. Merlo, M. Migliardi, L. Oneto and F. Palmieri, "Low-Resource Footprint, Data-Driven Malware Detection on Android," in IEEE Transactions on Sustainable Computing, vol. 5, no. 2, pp. 213-222, 1 April-June 2020, doi: 10.1109/TSUSC.2017.2774184.
13. S. Y. Yerima and S. Sezer, "DroidFusion: A Novel Multilevel Classifier Fusion Approach for Android Malware Detection," in IEEE Transactions on Cybernetics, vol. 49, no. 2, pp. 453-466, Feb. 2019, doi: 10.1109/TCYB.2017.2777960.
14. C. Guo, J. Xu, L. Liu and S. Xu, "Using association statistics to rank risk of Android application," 2015 IEEE International Conference on Computer and



Communications (ICCC), 2015, pp. 6-10, doi: 10.1109/CompComm.2015.7387530.
15. K. Khariwal, J. Singh and A. Arora, "IPDroid: Android Malware Detection using Intents and Permissions," 2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4), 2020, pp. 197-202, doi: 10.1109/WorldS450073.2020.9210414.
16. V. Kouliaridis, G. Kambourakis and T. Peng, "Feature Importance in Android Malware Detection," 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), 2020, pp. 1449-1454, doi: 10.1109/TrustCom50675.2020.00195.
17. A. Fatima, R. Maurya, M. K. Dutta, R. Burget and J. Masek, "Android Malware Detection Using Genetic Algorithm based Optimized Feature Selection and Machine Learning," 2019 42nd International Conference on Telecommunications and Signal Processing (TSP), 2019, pp. 220-223, doi: 10.1109/TSP.2019.8769039.
18. A. Sangal and H. K. Verma, "A Static Feature Selection-based Android Malware Detection Using Machine Learning Techniques," 2020 International Conference on Smart Electronics and Communication (ICOSEC), 2020, pp. 48-51, doi: 10.1109/ICOSEC49089.2020.9215355.
19. L. D. Coronado-De-Alba, A. Rodríguez-Mota and P. J. Escamilla-Ambrosio, "Feature selection and ensemble of classifiers for Android malware detection," 2016 8th IEEE Latin-American Conference on Communications (LATINCOM), 2016, pp. 1-6, doi: 10.1109/LATINCOM.2016.7811605.
20. A. Muñoz, I. Martín, A. Guzmán and J. A. Hernández, "Android malware detection from Google Play meta-data: Selection of important features," 2015 IEEE Conference on Communications and Network Security (CNS), 2015, pp. 701-702, doi: 10.1109/CNS.2015.7346893.
21. K. Zhao, D. Zhang, X. Su and W. Li, "Fest: A feature extraction and selection tool for Android malware detection," 2015 IEEE Symposium on Computers and Communication (ISCC), 2015, pp. 714-720, doi: 10.1109/ISCC.2015.7405598.
22. Varsha M V, Vinod P and Dhanya K A, "Heterogeneous feature space for Android malware detection," 2015 Eighth International Conference on Contemporary Computing (IC3), 2015, pp. 383-388, doi: 10.1109/IC3.2015.7346711.
23. S. J. K., S. Chakravarty and R. K. Varma P., "Feature Selection and Evaluation of Permission-based Android Malware Detection," 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184), 2020, pp. 795-799, doi: 10.1109/ICOEI48184.2020.9142929.
24. M. Qiao, A. H. Sung and Q. Liu, "Merging Permission and API Features for Android Malware Detection," 2016 5th IIAI International Congress on Advanced Applied Informatics (IIAI-AAI), 2016, pp. 566-571, doi: 10.1109/IIAI-AAI.2016.237.
25. D. Ma, Y. Bai, Z. Xing, L. Sun and X. Li, "A Knowledge Graph-based Sensitive Feature Selection for Android Malware Classification," 2020 27th



Asia-Pacific Software Engineering Conference (APSEC), 2020, pp. 188-197, doi: 10.1109/APSEC51365.2020.00027.

26. J. Jung, K. Lim, B. Kim, S. -j. Cho, S. Han and K. Suh, "Detecting Malicious Android Apps using the Popularity and Relations of APIs," 2019 IEEE Second International Conference on Artificial Intelligence and Knowledge Engineering (AIKE), 2019, pp. 309-312, doi: 10.1109/AIKE.2019.00062.
27. Scikit-learn: Machine Learning in Python, Pedregosa *et al*., JMLR 12, pp. 2825-2830, 2011.
28. M. A. Ambusaidi, X. He, P. Nanda and Z. Tan, "Building an Intrusion Detection System Using a Filter-Based Feature Selection Algorithm," in *IEEE Transactions on Computers*, vol. 65, no. 10, pp. 2986-2998, 1 Oct. 2016, doi: 10.1109/TC.2016.2519914.
29. A. Arora, and S. Peddoju, ”NTPDroid: A Hybrid Android Malware Detector Using Network Traffic and System Permissions”, 17th IEEE TrustCom, 2018.
30. A. Arora, S. Peddoju, V. Chauhan, and A. Chaudhary, ”Hybrid Android Malware Detection by Combining Supervised and Unsupervised Learn- ing”, 24th ACM MobiCom, 2018.
31. A. Arora, S. Garg, and S.Peddoju,”Malware detection using network traffic analysis in android based mobile devices”, 8th IEEE NGMAST,2014.
32. A. Arora, and S. Peddoju, ”Minimizing Network Traffic Features for Android Mobile Malware Detection”, 18th ACM ICDCN, 2017.
33. A. Arora, S. K. Peddoju and M. Conti, "PermPair: Android Malware Detection Using Permission Pairs," IEEE Transactions on Information Forensics and Security, vol. 15, pp. 1968-1982, 2020.