

Android Malware Detection using Decision Trees and Random Forest

Submitted in partial fulfillment of the requirements of

Mini Project (CSM501)

for

Third Year of Computer Engineering

By

Bhavesb Dhake 19102A0027

Chaitanya Shetye 19102A0072

Devish Gawas 19102A0024

Pratik Borhade 19102A0049

Under the Guidance of

Prof. Ravindra Sangle

Department of Computer Engineering



Vidyalankar Institute of Technology
Wadala(E), Mumbai-400437

University of Mumbai

2021-22

CERTIFICATE OF APPROVAL

This is to certify that the project entitled

“Android Malware Detection using Decision Trees and Random Forest”

is a bonafide work of

Bhaves Dhake 19102A0027

Chaitanya Shetye 19102A0072

Devish Gawas 19102A0024

Pratik Borhade 19102A0049

submitted to the University of Mumbai in partial fulfillment of

Mini Project (CSM501)

for

Third Year of Computer Engineering

Guide
(Name)

Head of Department

Principal

Mini Project Report Approval

This project report entitled *Android Malware Detection using Decision Trees and Random Forest* by

- 1. Bhavesh Dhake 19102A0027**
- 2. Chaitanya Shetye 19102A0072**
- 3. Devish Gawas 19102A0024**
- 4. Pratik Borhade 19102A0049**

is approved for Mini Project (CSM501) for Third Year of Computer Engineering.

Internal Examiner



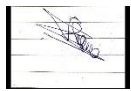

External Examiner

Date: 18/10/2021

Place: Mumbai

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

	Name of student	Roll No.	Signature
1)	Bhavesh Dhake	19102A0027	
2)	Chaitanya Shetye	19102A0072	
3)	Devish Gawas	19102A0024	
4)	Pratik Borhade	19102A0049	

Date: 18/10/2021

Place: Mumbai

Acknowledgements

This Project wouldn't have been possible without the support, assistance, and guidance of a number of people whom we would like to express our gratitude to. First, we would like to convey our gratitude and regards to our mentor **Prof. Ravindra Sangle** for guiding us with his constructive and valuable feedback and for his time and efforts. It was a great privilege to work and study under his guidance.

We would like to extend our heartfelt thanks to our Head of Department, **Dr. Sachin Bojewar** for overseeing this initiative which will in turn provide every Vidyalankar student a distinctive competitive edge over others.

We appreciate everyone who spared time from their busy schedules and participated in the survey. Lastly, we are extremely grateful to all those who have contributed and shared their useful insights throughout the entire process and helped us acquire the right direction during this research project.

Abstract

Mobile malware is so pernicious and on the rise, accordingly having a fast and reliable detection system is necessary for the users. In this research, a new detection and characterization system for detecting meaningful deviations in the network behavior of a smart-phone application is proposed. The main goal of the proposed system is to protect mobile device users and cellular infrastructure companies from malicious applications with just 9 traffic feature measurements. The proposed system is not only able to detect the malicious or masquerading apps, but can also identify them as general malware or specific malware (i.e. adware) on a mobile device. The proposed method showed the average accuracy (91.41%), precision (91.24%), and false positive (0.085) for four classifiers namely; Random Forest (RF), Decision Tree (DT), Random Tree (RT) and Regression (R). We also offer a labeled dataset of mobile malware traffic with 1900 applications includes benign and 12 different families of both adware and general malware.

Table of Contents

Sr No	Description	Page No
1	Introduction	8
2	Problem Definition	9
3	Literature Survey	10-14
4	Dataset	15-16
5	Conclusion	17
6	References	18

1. Introduction

Nowadays with the dramatic growth of available applications in the smart-phones, the users' behavior to interact and utilize the network has changed significantly for access via smart-phones. Besides, mobile networks traffic substantially increased because of many always-connected applications such as social networks apps. According to the Ericsson mobility report, by 2020, the subscription of smartphones will be more than 6 billion and 80 percent of traffic will be generated by mobile networks. Also, more than 96% of global mobile traffic belongs to Android and Apple, which 50% of this traffic is generated by Android smartphones.

Several issues have risen in smart-phones development. Firstly, with the rapid development of applications on the Android smart-phone due to market demand, an outbreak trend of malicious software has appeared. Secondly, Android is an open-source operating system which makes it vulnerable to all sorts of malicious attacks. Furthermore, most of the malwares in smart-phones are connecting to the servers for sending and receiving information for attacks. These abundance malware applications and attacks caused serious damage, such as affecting normal usage, monitoring users' activities, and stealing users' private information on smart-phones. Hence, detecting malware apps before damage occurs is crucial.

Most of the malware detection techniques which have been proposed can be categorized into static and dynamic groups. Finding malicious characteristics or bad code segments in an app without running the app is called static detection. Preparing an isolated environment such as an emulator or any devices for running the app and monitoring the app's dynamic behavior is named dynamic detection. This research focuses on the common dynamic behavior of malware and aims to use the generated traffic of malicious apps installed on the real Android device for malware detection and labeling adware apps among others.

2. Problem Definition

The main contribution of this research is to propose an Android Malware detection model based on a new network traffic feature set to expedite the efficiency of traffic classifier. For this purpose, nine new flow-based network traffic features presents for characterizing the benign, adware and apps. It means, not only can the proposed method detect the unknown Malware, but also it can label the type of Malware.

The main contribution of the proposed model is to detect Android malware and label them as specific malware (i.e. adware) or general malware. This operation, which is based on the detection of the network traffic deviation on mobile devices has been divided into three phases (Figure 1). The first phase focuses on the gathering of benign, general malware, and adware apps for generating the training and testing dataset. Executing these apps on the smart phones, capturing the generated traffic, and labeling them creates the dataset.

The second phase consists of measuring all features including all previous work defined features and our new proposed features. Also, this phase trains the machine learning model with 80% of dataset (Training part) and suggests the most suitable feature set for the detection and labeling process. The last phase computes the final selected feature set for labeling the new apps. This phase uses the 20% of dataset which has been separated from the whole dataset for testing and evaluating the proposed algorithm and feature set.

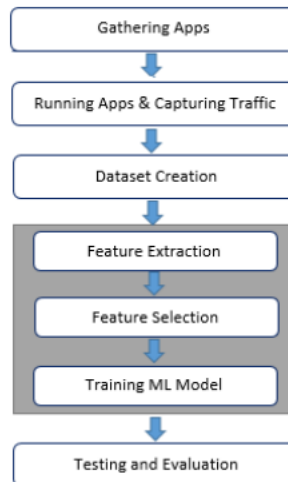


Figure 1: Procedure of Proposed Model

3. Literature Survey

A number of studies have been conducted on area of Android malware detection and characterization from the network traffic perspective. One of the first attempts was presented by Iland et al. in 201, where the authors presented a lightweight approach of detecting Android malware and violations of user privacy through the network traffic analysis. The authors conducted a series of controlled experiments. They first created the virtualized Android devices with a clean app (i.e. e-commerce app) installed from a third party market, and some dummy user data such as contact data, account passwords, browser history, and credit card information and infected them with 18 malware samples; then they analyzed the collected network traces to find information leakage behavior and identify connection attempts to command and control (C&C) servers; finally, they tested two approaches (i.e. IP/DNS blacklisting, string matching) for detecting the malicious behavior of Android malware and defined four features (HTTP header flag, HTTP-GET request, content and pattern of POST request, well-structured identifiers in POST request). However, the proposed approaches have several limitations. First, the blacklisting technique heavily relies on static malware behavior which requires frequent updating over time. Also, more complex attacks such as fast-flux and botnet are more difficult to effectively detect. The content matching technique infers information from plain-data transmission. But, this technique cannot be applied in the encrypted traffic.

Another detection technique using a sensor application was presented by Kuhnel & Meyer in 2012. They analyzed over 30 malware families targeting Android platform, which are divided into four categories: premium calls, SMS only, HTTP only, and Remote Access Tool (RAT). They also utilized the user space layer of Android Architecture by adding a filtering component, which is responsible for network analysis and can be controlled via a sensor app. The sensor app is also capable of the following tasks: inform a user about suspicious traffic, list all events from a local database, send blocked SMS, and change blocking preferences. With 95% of detection rate, the authors claimed that it is sufficient to detect mobile malware by simply filtering the ingoing and outgoing traffic.

In 2013, Tenenboim et al. proposed a novel network based behavioral analysis for detecting a new group of malware with self-updating capabilities which have been found in the Google Android marketplace. Researches show that this group of malware are not detectable by using available static and dynamic analysis methods or any signature-based approaches. They defined a specific pattern for the generated traffic of each application on the device as a representative model. They used the machine learning to find the deviations of the pattern from normal applications' patterns.

They also have been used nine features such as number of total concurrent connections of the application, number of sent and received TCP or UDP packets, number of sent and received TCP or UDP payload bytes and number of sent and received TCP segments which they defined in their previous publication with five fixed time intervals for calculating the various aggregation

functions such as average, standard deviation, minimum, and maximum. For 15 selected applications, they analyzed the original benign and repackaged version which has been created by injecting malware code of five real applications and also ten self-written Trojan malware. Their results showed that in most of the applications the threat was detected in the first five minutes after the infection occurred.

In the same year, Dai et al. presented a new automatic network profile generator for detecting Android applications in HTTP. They mentioned that regarding the huge number of applications that are appearing every day based on HTTP/HTTPS, traditional method of traffic classification are no longer useful for traffic analysis. In this project, they first ran thousands Android apps in an emulator and collected the network traces. Then for extracting apps' fingerprints, they proposed and developed a light-weight technique that can break the request to "method", "query" that can be split to key-values and "page" that can be broken into "page components" and "filename". There are two limitations of the proposed technique. First the system needs a user seed path when login is involved and secondly it cannot detect apps which have no distinct network behavior and use the same service.

Arora et al. proposed an Android malware detection method using its network traffic analysis. They used an Android emulator on a host with public IP and captured the traffic of thirteen malware to create their dataset. Based on previous work, they have chosen sixteen traffic features such as average packet size, average duration of the flow, average number of bytes sent per flow. Then they using machine learning algorithms with their dataset and finalized seven features as the final feature set such as Average number of bytes received per second, average number of packets sent per flow, average number of packets received per flow. In the experiment and analysis section they categorized their dataset to three risky levels of Malware, namely high, medium and low. Their classifier correctly predicted 45 of 48 samples which is around 95% accuracy. The small size and lack of diversity of the Malware in the dataset, could be the major deficiencies of this research.

Similarly, Shabtai et al. stated that the best detection technique for malware is using the application's network traffic patterns. They proposed a specific model based on some features such as sent/receive data in byte, network state, send/receive mode, total time in force/background and minutes since last active/modified time along with related aggregation functions such as minimum, maximum and standard deviation to represent a specific traffic pattern for each application. They, also used semi-supervised machine learning methods for defining the normal behavior and calculating the deviation for detecting the abnormal activities. For evaluation, they designed and implemented their method on Android devices with five real malware and ten in-house malware. For benign purpose, the original application were executed and for the malicious purpose, the repackaged versions injected with malware code were operated. The evaluation results indicated that certain categories of applications can be detected by specific traffic patterns. Also, they realized that malware with self-updating ability have various and diagnosable traffic patterns for a few minutes after starting execution.

Li et al. proposed a network traffic monitoring system for detecting Android malware in order to improve the Android terminal defense ability against malicious attacks, and Advanced Persistent Threat (APT) attacks. The system consists of four components: traffic monitoring, traffic anomaly recognition, response processing, and cloud storage. Overall, the system will perform the following steps: (1) parses the protocol of data packets, (2) extracts the feature data (ID of the process, the start of the network connection time, end of the network connection time, upward flow, downward flow, packet source IP address, packet destination IP address, protocol type, packet source port, destination port number), (3) uses an SVM classification algorithm for data classification, (4) determines whether the network traffic is abnormal, (5) locates the application that produced abnormal through the correlation analysis. The experiment results show that the monitoring system can effectively detect the Android malware with fewer false positives.

Another analysis focusing on the network-based malware detection system was conducted by Carrasquillo et al. in 2014. The authors combined both the signature-based traffic analysis and the network flows analysis of a virtual private network (VPN) targeting the mobile platform. The goal is to provide alarms and visualization analytics of anomalous behavior to both mobile users and the network administrators. The framework of the detection system uses Snort (for the signature-based) and NetFlow by Cisco (for network flows analysis), which have the following features: Internet Protocol address (IP), the destination IP, the source port, destination port, the sum of the payload size of the packets, and timestamp.

Chen et al. believed the main reasons that malware detection research in Android devices still remained theoretical are because of the lack of systematic analysis of traffic features and a large-scale repository of malware. Thus, for capturing the malware generated traffic data in real internet, they proposed and designed a behavior monitoring scheme for Android malware traffic and applied 24 different families of Android malwares. Regarding their analysis of the major compositions of the first 5 minutes of generated traffic, more than 99% of traffic were HTTP and DNS.

They used some common network features such as DNS query, HTTP packet length, HTTP request, and Ad traffic features in the analysis process and discovered that HTTP request can detect malware 40.89% and DNS query can detect 69.55%. Also, they illustrated that more than 70% of malwares started to generate the malicious traffic in the first 5 minutes. As Ad and malware are major HTTP traffic components, so Ad traffic can significantly affect malware detection process.

Zaman et al. in 2015 demonstrated a work-in progress detection method that was effective against malware communicating with the remote server C&C servers, which is based on the logs of all remote locations. The authors divided their detection procedure into two steps (1) creating the URLs table, and (2) matching the URLs with blacklists. They conducted four main tasks in the first procedure of creating the URLs table: (1) packet dumping - recording

incoming and outgoing network packets, (2) netstat logging - checking port numbers, (3) extracting necessary information from packet dump - filtering out all other packets from the packet dump, (4) Aggregating packet dump and netstat logs - creating a time-sequenced log of applications. They also discussed their direction of research toward developing an intelligent malware detection model.

Wang et al. presented an effective malware identification and classification method called TrafficAV by combining machine learning algorithm and traffic analysis. In order to get minimum resources on mobile devices without affecting the user experience, the authors performed all data analysis and malware detection on the server side by mirroring the network traffic generated by mobile app from the wireless access point to the server for data analysis. The authors highlighted that with machine learning algorithm (C4.5 decision tree) the detection rates of TCP flow and HTTP models reach 98.16% and 99.65% while the false positive rates are 5.14% and 1.84%.

A recent paper in 2017 presented an algorithm to prioritize network traffic features with minimize number of features to be analyzed for better detection accuracy and processing time. The results showed that 9 features out of 22 are sufficient to give maximum detection accuracy (85% up to 100%). Moreover, the authors claimed that these 9 features also save considerable amount of time for training and testing the dataset. The training time of 300 applications is reduced from 11.7 seconds to 5.8 seconds and testing time of 230 applications is reduced from 25.1 seconds to 17.3 seconds.

In addition to Android studies, we also reviewed some previous work on computer network traffic analysis. In 2005, Karagiannis et al. presented BLINC, a multilevel traffic classification of host behavior at the transport layer. This paper was the first attempt in shifting from characterizing flows by application to associating hosts with applications at three different levels: (1) social level- capture source and destination IP addresses, (2) network level - analyze characteristics of the source and destination IP address, and the source port., and (3) the application level - capture additional flow information, such as the transport protocol or the average packet size. A range of application types was studied in this work, including network management traffic, mail, chat, media streaming, web, p2p, data transfer, and gaming. In order to respect the privacy, the authors 235 operated the study in the dark (having no access to packet payload, no knowledge of port numbers, and no additional information other than what current flow collectors provide). They represented these patterns by using graphs, which called graphlets and matched them with the library of host behavior under examination. The results showed that BLINC is able to classify 80%- 90% of the traffic with more than 95% accuracy.

Later in 2008, Nguyen & Armitage conducted a survey to analyze techniques for Internet traffic classification using machine learning during the period of 2004 to early 2007. They reviewed 18 significant works and discussed the following: (1) the importance of IP traffic classification in operational networks., (2) the limitations of traditional portand payload-based

classification., (3)the metrics for assessing classification accuracy., and (4) the key requirements for the employment of ML-based classifiers in operational IP networks. The authors highlighted that most of the machine learning algorithms for offline analysis, such as AutoClass, Expectation Maximisation, Decision Tree, NaiveBayes etc. has demonstrated high accuracy (up to 99%). They also suggested that the promising results of machine learning based IP traffic classification can open many new avenues for related research areas, such as the application of ML in intrusion detections, anomaly detection, routing traffic, real time monitoring and management, and grey net or darknet networks.

Recently in 2016, Bartos et al. presented a robust representation with a supervised method for classifying malware behaviors. Instead of classifying flows individually, the authors grouped flows into bags, where each bag contains flows that are related to each other. They also developed a novel method that combines the process of learning the representation with the process of learning the classifier. They deployed the system on large corporate networks and evaluated them on real HTTP network traffic with more than 43k malicious samples and more than 15M samples overall. The system detected 2,090 new and unseen variants of malware samples with 90% precision (9 of 10 alerts were malicious).

In summary, most of the related work from 2010 to early 2017 have been selected and evaluated in order to select a collection of commonly used features that could yield high malware detection accuracy based on network traffic.

4. Dataset

One of the contributions of this project is a labeled Android network traffic dataset that we found for our experiment. To generate traffic that represents the real world, real smart phones were used (NEXUS 5) instead of emulators or Android Virtual Device (AVD) to ensure that the dataset is rich enough in quality and quantity. Table I shows the yearly breakdown of the collected dataset.

No	Creation Date (Year)	Benign	Malware
1	2008	18	0
2	2009	9	0
3	2010	2	1
4	2011	6	8
5	2012	7	2
6	2013	5	10
7	2014	59	27
8	2015	236	29
9	2016	765	20
10	Altered	393	303
Total		1500	400

Table I

A. Benign Dataset

1527 benign apps were collected from Google play market in 2015 and 2016. These apps were collected based on their popularity (i.e., top free popular and top free new) for each category available on the market. Out of 1527 apps, 27 of them were removed (leaving 1500 apps in total) as they were flagged as suspicious or adware by more than two Anti-Virus (AV) products in Virustotal web service.

B. Malware Dataset

According to the recent Symantecs Internet Security Threat Report (SISTR), the normal distribution of benign and malware apps in the real world is 80% to 20%, so we selected 400 malware apps versus 1500 Benign apps to create a balanced dataset. We have collected these 400 malware from two categories, i.e. adware (250) and general malware (150). The first category, which is an adware consisting the following popular families:

- Airpush: Designed to deliver unsolicited advertisements to the user's systems for information stealing.

- Dowgin: Designed as an advertisement library that can also steal the user's information.
- Kemoge: Designed to take over a user's Android device. This adware is a hybrid of botnet and disguises itself as popular apps via repackaging.
- Mobidash: Designed to display ads and to compromise user's personal information.
- Shuanet: Similar to Kemoge, Shuanet also is designed to take over a user's device. Lookout, an Antivirus 237 company claimed that the authors of Kemoge and Shuanet used the same pieces of code to build their versions of the auto-rooting adware with 71 percent to 82 percent code similarity.

The following families have been chosen for the apps:

- AVpass: Designed to be distributed in the guise of a Clock app.
- FakeAV: Designed as a scam that tricks user to purchase a full version of the software in order to re-mediate non-existing infections.
- FakeFlash/FakePlayer: Designed as a fake Flash app in order to direct users to a website (after successfully installed).
- GGtracker: Designed for SMS fraud (sends SMS messages to a premium-rate number) and information stealing.
- Penetho: Designed as a fake service (hacktool for Android devices that can be used to crack the WiFi password). The malware is also able to infect the user's computer via infected email attachment, fake updates, external media and infected documents.

C. Capturing Traffic from Real Smartphone

After defining group of apps, they proceeded with the apps installation on the real smartphone, which is on NEXUS 5. They first configured the NEXUS 5 by syncing the already installed apps such as Gmail, Facebook, and YouTube with the user profile named iscx. We created this iscx profile to make sure that the smartphone was connected with registered email account. They then connected the smartphone to the computer by turning on the setting of USB debugging and developer access. Then they ran a script to install the apps in bulk. They first installed the benign apps in groups of 20 and start running all apps and have some interactions in the following hour. Then the apps were removed and new group of apps were installed. All the traffic was captured in the access point. We only captured the traffic which was coming from the smartphone. They followed the same procedure for installing the general malware and adware groups. However, in this case, the smartphone was reset and reconfigured after each installation to make sure that the smartphone was clean and not infected with the malware (before installing new malware apps).

5. Snippets of the Model

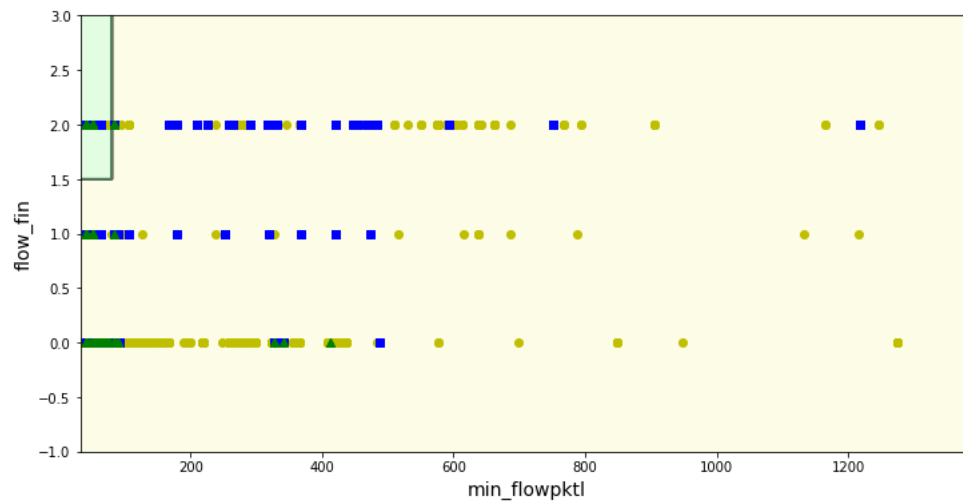
Accuracy of the Training Data Set

```
f1_score WITHOUT preparation: 0.9593915432471419  
f1_score WITH preparation: 0.9593876451031004
```

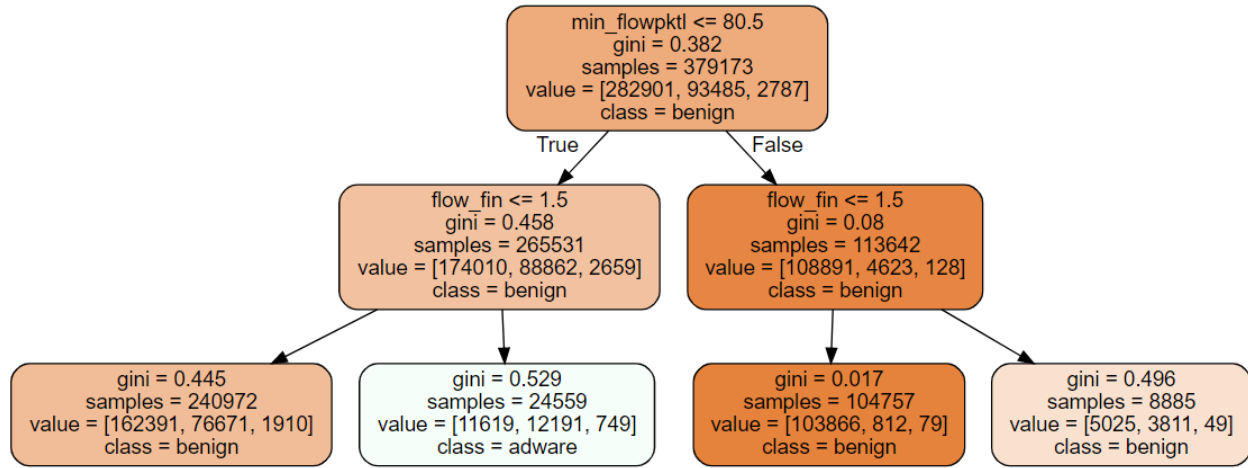
Accuracy of the Validation Data Set

```
f1_score WITHOUT preparation: 0.9335628011437564  
f1_score WITH preparation: 0.7938732691267106
```

Graphical Representation of the Constructed Decision Limit



Tree Diagram



6. Conclusion

In this research, we proposed a mobile malware detection model based on 9 traffic features to expedite the efficiency of traffic classifier. Moreover, the model uses classification methods including flow-based, packet-based and time-based features to characterize malware families. The analysis shows the proposed feature set has more than 93% accuracy in the detection and 92% success probability on characterization with less than 0.08 percent false positive rate on average, which is adequately good and necessary for real world malware detection systems. In future work, we plan to include other features such as system and host information together with the traffic features for the malware characterization and propose a complete detection system for smart-phones.

6. References

- I. Danny Iland, Alexander Pucher and Timm Schauble, Detecting Android malware on network level, University of California, Santa Barbara, vol 12, 2011
- II. Tenenboim-Chekina, L and Barad, O and Shabtai, A and Mimran, D and Rokach, L and Shapira, B and Elovici, Y, Detecting application update attack on mobile devices through network feature, 2013 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), p91-92, 2013
- III. Z. Chen and H. Han and Q. Yan and B. Yang and L. Peng and L. Zhang and J. Li, A First Look at Android malware Traffic in First Few Minutes, Trustcom/BigDataSE/ISPA, Vol 1, p206-213, 2015 IEEE
- IV. Marian Kuhnle and Ulrike Meyer, Detecting malware initiating traffic on mobile device, RWTHAACHEN university, 2012
- V. J Chekina, Lena and Mimran, Dudu and Rokach, Lior and Elovici, Yuval and Shapira, Bracha, Detection of deviations in mobile applications network behavior, arXiv preprint arXiv:1208.0564, 2012
- VI. Shabtai, Asaf and Tenenboim-Chekina, Lena and Mimran, Dudu and Rokach, Lior and Shapira, Bracha and Elovici, Yuval, Mobile malware detection through analysis of deviations in application network behavior, Computers & Security journal Elsevier, vol 43, p1-18, 2014
- VII. J A. Arora and 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
- VIII. Nguyen, T. T., and Armitage, G. (2008). A survey of techniques for internet traffic classification using machine learning. IEEE Communications Surveys & Tutorials, 10(4), 56-76.
- IX. Arora, Anshul, and Sateesh K. Peddoju. "Minimizing Network Traffic Features for Android Mobile Malware Detection." Proceedings of the 18th International Conference on Distributed Computing and Networking. ACM, 2017.
- X. Wang, Shanshan, et al. TrafficAV: An effective and explainable detection of mobile malware behavior using network traffic. Quality of Service (IWQoS), 2016 IEEE/ACM 24th International Symposium on. IEEE, 2016.
- XI. M. S. Alam and S. T. Vuong, "Random Forest Classification for Detecting Android malware," 2013 IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing, Beijing, 2013, pp. 663-669.