Malware Detection in Android in different application Categories

Nasreen Babar
Dept.of Software Engineering
Mehran University of Engineering & Technology
Jamshoro, Pakistan
17mese05@students.muet.edu.pk

Dr.Sania Bhatti
Dept.of Software Engineering
Mehran University of Engineering & Technology
Jamshoro, Pakistan
sania.bhatti@faculty.muet.edu.pk

Salahuddin Saddar
Dept.of Software Engineering
Mehran University of Engineering & Technology
Jamshoro, Pakistan
salahuddin.saddar@faculty.muet.edu.pk

Uroosa Shaikh
Dept.of Software Engineering
Mehran University of Engineering & Technology
Jamshoro, Pakistan
suroosa@rocketmail.com

Sorath Buriro
Dept.of Software Engineering
Mehran University of Engineering & Technology
Jamshoro, Pakistan
sorathburiro@gmail.com

***Abstract*—In this smart era, mobile phone devices are very common in use. Most people are using it as a daily gadget, doing calculations, saving memos, using notebooks, keeping account details, and even controlling their houses using smart sensors. Man, of this technological era, has earned many inventions and innovations including smartphones. So, they are at high risk by attackers and hackers. They are trying to inject malicious code that the user accepts as permission at the time of installing any application. Detection is necessary to protect the devices from injecting harmful malware. In this paper, we collected applications targeting five different categories i.e. Islamic, Education, Home, Shopping, and Medical. The applications are downloaded from their official marketplace and the permissions are extracted using VirusTotal. The pre-analyzed dataset contains two thousand and five hundred android applications. The malicious and benign applications are identified with the help of permission keywords. Machine Learning Classifiers including Support Vector Machine, Decision Tree, Gradient Boosted Tree, and Random Forest are used to analyze the pre-processed data by the latest version of Rapid Miner 9.8. The classifiers such as the Decision tree and Random Forest give approximate values and the runtime rate is faster than the Gradient Boosted Tree and Support Vector Machine.**

Keywords*—*Android Malware, Malware, Non-Malware, Detection, Detection Techniques, Permissions, Machine Learning Classifiers

I. INTRODUCTION

Smartphones are the most-used devices conducting many tasks from taking selfies, checking the weather to the financial activities and businesses. Smartphones are being used at a large number of involvement of users nowadays. As smartphones are relatively new, so the cyber-attackers can easily catch the smartphones' damages, and viruses, gain access to the emails, contacts, and other activities. Most of the previously developed cyber-attacks on personal computers are repackaged versions that are attacking mobile devices. The smartphones running under the Android operating system are epidemically affected by severe Android malware. Malware is considered as the most threatening task that is affecting the businesses in reaching their goals and stealing classified information. Malware is comprising of two words i.e. Malicious + Software [7]. It is a term used by the attackers to harm the devices' security. Understanding malware is becoming more and more complicated, and the topic of interest in the mobile security industry. The countermeasures increase the complexity to detoxify the applications with cleanser in the technical industry.

A. Malware Detection Background

Commonly the malware detection techniques are of two types. (i) Static detection technique. (ii) Dynamic detection technique.

The static detection technique is an approach of analyzing source code before the execution of the different applications downloaded by end-users [9]. This approach unpacks and disassembles the code to extract some features of the apps to identify malware. Static based techniques are easily automated, flexible, and lightweight. Some well-known static techniques are: Permission-based-technique, Signature-based-technique, etc. [12]

The dynamic detection technique is an approach that identifies malicious reaction after executing the application in a controlled environment. Some dynamic techniques are: Anomaly-based-technique, Taint analysis, Emulation-based-technique, etc. [12]

Some researchers have identified the complexity of modern malware. It is making it very difficult to detect. Agobot [17] is a malware program released in 2002, which attacks, steal

bank accounts details, and propagate itself over a network to avoid its detection. Malware comes in different forms and uses different techniques to comply with their actions, all depending upon the hacker's intent.

B. Different permission types

Permission types can be categorized into two forms. Normal Permissions, Signature Permissions, and Dangerous Permissions. [8]-[10]

- *Normal Permissions:*

Normal permissions, that the system automatically granted by the systems' app permission at the time of installation. Users cannot revoke these permissions. There are very low-risk factors to the user's privacy and device protection.

- *Signature permissions:*

These app permissions are granted by the system at the installation, but only when the app permission is signed by the same certificate as the app attempts to use the grant that defines the permission [5].

- *Dangerous Permissions:*

Dangerous permission wants the data that involves the private information of the user. They are granted only if the user accepts the permission and it starts the installation. Dangerous permissions contain harmful API calls which will drain the user into a big loss.
Such as SEND_SMS, RECEIVE_SMS or CALL_PHONE are the permissions which give access to personal information, bank account details, loggings into the sites and passwords, etc.

I. RELATED WORK

In this section, we discuss some latest work related to malware detection in Android from the literature. Normally, malware drenches into the system's security because of the extensive use of Smartphones nowadays. Malware developers use to target the Android OS because of its great market tendency. The cost-free applications have given popularity to Smartphones' Android operating system, but it also increases the risk of malware-based applications. The malicious-based application affects the systems' security due to the lack of knowledge of the end-user. The end-user is unaware of the things which are being injected. The sensitive information of the user is being stolen by affecting the privacy, it results in stealing passwords, contacts, image files, etc. [6]

In 2020, Xu Jiang et al. [1] proposed a novel method named Fine-Grained Dangerous Permissions (FDP) which detect android Malware apps. The mechanism represents the differences between malware and non-malware applications as gained features of machine learning. They demonstrated the effectiveness of FDP, achieving a true positive rate of 94.5%. Experiencing one thousand and seven hundred benign apps from the Xiaomi market and, one thousand and six hundred malicious apps from malware families.

In another research in 2020, Rishabh Agarwal et al. [2] studied and highlighted the existing detection techniques and analysis approaches used for android malevolent code. Machine learning algorithms are used to analyze malware by doing semantic analysis.

In a research article in 2020 [3], an algorithm of android malware detection is proposed using a hybrid deep-learning model. It combines Deep-Belief Networks (DBN) and Gate-Recurrent-Unit (GRU). Experiments result in comparison with the traditionally customized machine learning algorithms. Android malware detection model based on hybrid deep learning algorithm gives higher detection accuracy.

In another research in 2019, [4] a detection model proposed to pursue and keep track of malware apps behaviors in such devices. An innovative machine learning classifier is employed, and an alarm will be triggered if the android app detects the malicious code. The proposed classification algorithm accomplishes high accuracy, true-positive rates, false-positive rates, etc.

In 2019, the authors [18] proposed a new Android malware detection method based on the correlation relationship of the API calls of applications. At the very first, they split the source code into the abstract API calls transactions and then computed the confidence of the association of interrelated rules between abstracted API calls.

In 2018, [6] the study focuses on discovering and examining Android malware attacks to understand how malware programs misuse the flaws of the system and take advantage of the inadequacy of a set of rules which occur due to the user's lack of knowledge regarding Android applications. Earlier work specifies the issues regarding different kinds of software methods and breaks down the Android Malware.

A survey in the year 2017, [8] describes the different approaches to detecting various kinds of malware. They highlighted their advantages and limitations. They observed that detection of malicious contents using the static approach is less efficient when compared to the dynamic approach which keeps monitoring the apps remotely or otherwise.

Research in 2016, [13] proposed an effective methodology of detecting Android malware using static code analysis-based models. Many studies have been done in the field of Android Malware Detection. As mentioned early, the main malware detection techniques are dynamic and static. CrowDroid is the most popular dynamic technique used to detect malware by monitoring the behavior through analysis

of the application system call. The behavior is analyzed by executing on an emulator or in a controlled environment.

Several apps are available on the dataset provided on the http://amd.arguslab.org website. This dataset link contains more than twenty-four thousand app-related data. It has defined 135 types of malware, but the dataset does not explicitly define the categories of analyzed apps. [14]

However, we collected our data of two thousand and five hundred samples (five nominated categories) from different sources.

II. METHODOLOGY

In this section, there are four phases. The first phase includes the downloading and installation of categorical apps. The apps are downloaded from a well-known website APKPure.com [16]. It provides a huge number of applications verified by the official marketplace Google Play. In the second phase, the manifest file is extracted. The third phase covers the detection of malware and benign apps using an online tool VirusTotal [15]. The fourth phase analyzes the apps containing severe and harmful malware using machine learning algorithms.

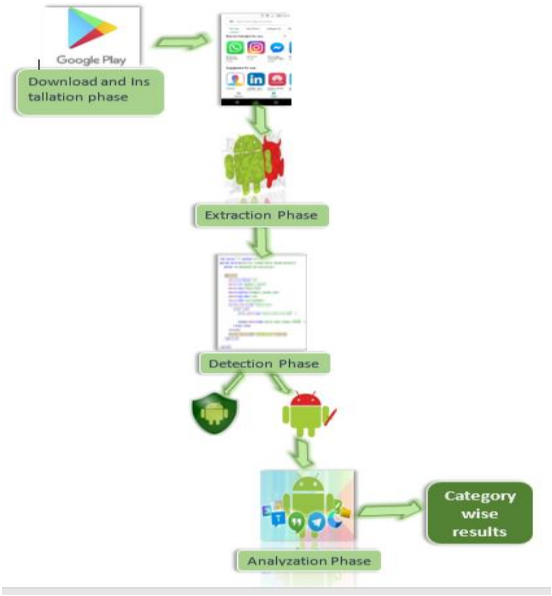


Fig.1 Phases of methodology

III. IMPLEMENTATION

This section depicts the methods carried out during thorough experiments. The process is comprising of the collection and analysis of malware and non-malware applications of different categories available on the official marketplace i.e. Google Play Store. The detailed process contains four phases. The details of each phase are discussed as follows.

A. Pre-processed data analytics of android apps:

The dataset is achieved by collecting random applications containing both malicious and non-malicious content. The pre-processed data is collected from an online tool i.e. APKPure. APKPure.com is a website providing smartphone software downloads. It is founded in the year 2014 by the APKPure Team and has grown into one of the leading websites in the smartphone software industry. APKPure is not affiliated with Google.com, Google Play, or Android anyway. These. apk files are downloaded relevantly to achieve the ratio of malicious and benign applications of different categories under the umbrella of Google Play Store provided in every smartphone and gadget. The five targeted categories of the Play Store are Islamic, Education, Medical, Shopping, and Home. Five hundred .apk files of each category are downloaded and extracted. In a meanwhile, the total number of downloaded .apk files is two thousand five hundred. The .apk files are then uploaded to the VirusTotal one by one to get the desired permission keywords. Each application was scanned and filtered out as benign or malicious. The results were recorded based on hazardous and non-hazardous permission. Table 1 shows the permission keywords of the benign apps and table 2 shows the permission keywords of the malicious apps.

Table 1 Benign apps' permission keywords

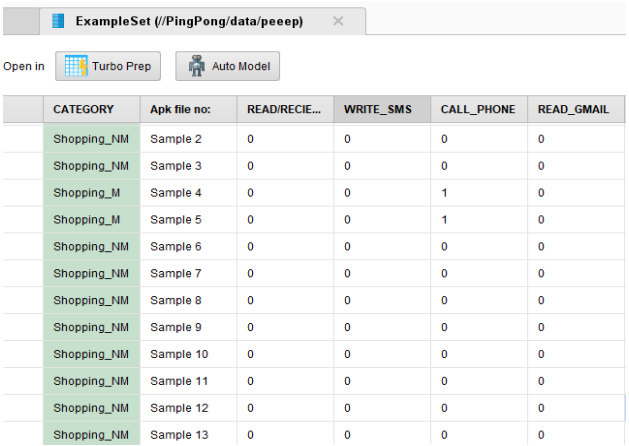
App Name	Category	Permission Keyword
Sample-1	Islamic	internet, read_external_storage, access_wifi_location, wake_lock access_fine_location, foreground_service access_network_state
Sample-2	Shopping	(internet, read_external_storage, access_wifi_location, wake_lock access_fine_location, access_network_state
Sample-3	Medical	(internet, read_external_storage, access_wifi_location, wake_lock access_fine_location, access_network_state
Sample-4	Home	(internet, read_external_storage, access_wifi_location, wake_lock access_fine_location, access_network_state
Sample-5	Education	(internet, read_external_storage, access_wifi_location, wake_lock access_fine_location, access_network_state)

Where,

android.permission.FOREGROUND_SERVICE is a service that automatically accepts the app request. If the request is not accepted by foreground services, it throws a security exception.

android.permission.READ_EXTERNAL_STORAGE allows an application from external storage. Android. permission.ACCESS_NETWORK_STATE allows network information etc.

Table 2 Malicious apps' permission keywords



	CATEGORY	Apk file no:	READ/RECIE...	WRITE_SMS	CALL_PHONE	READ_GMAIL
	Shopping_NM	Sample 2	0	0	0	0
	Shopping_NM	Sample 3	0	0	0	0
	Shopping_M	Sample 4	0	0	1	0
	Shopping_M	Sample 5	0	0	1	0
	Shopping_NM	Sample 6	0	0	0	0
	Shopping_NM	Sample 7	0	0	0	0
	Shopping_NM	Sample 8	0	0	0	0
	Shopping_NM	Sample 9	0	0	0	0
	Shopping_NM	Sample 10	0	0	0	0
	Shopping_NM	Sample 11	0	0	0	0
	Shopping_NM	Sample 12	0	0	0	0
	Shopping_NM	Sample 13	0	0	0	0

App Name	Category	Permission Keyword
Sample-1	Shopping	(read_SMS, write_SMS, get_accounts, read_contacts)
Sample-2	Shopping	(receive_SMS, get_accounts, access_fine_location)
Sample-3	Education	(call_phone, disable_keyguard, get_accounts)
Sample-4	Education	(read_gmail, disable_keyguard, get_accounts)
Sample-6	Medical	(call_phone, get_accounts, change_WiFi_setting access_fine_location)
Sample-7	Home	(disable_keyguard, access_fine_location change_wifi_settings)
Sample-8	Islamic	(read_phone_state, get_accounts, read_gmail)

However, android. permission.SEND_SMS allows an application to send SMS.android.

permission.WRITE_CONTACTS allows the application to write contacts etc.

B. Pre-processed Dataset of Android Apps:

The dataset is a mixture of both malware and non-malware applications. We collected 2500 applications (2364 benign and 136 malicious) and 15 permission keywords for each application. Table 3 depicts some permission for malware and non-malware apps.

Table 3 Some permissions collected in 1's and 0's

Apk file no	READ/ RECIEVE_ SMS	WRITE_S MS	CALL_PH ONE	READ_G MAIL	DISABLE_ KEYGUA RD	GET_ACC OUNTS	READ/ WRITE_C ONTACTS	DOWNLO AD-WITHOU T _NOTIFIC ATION	READ_PH ONE _STATE	CLASS	CATEGORY
Sample 1	0	0	0	0	0	0	0	0	0	NM	Islamic
Sample 2	0	0	0	0	0	0	0	0	0	NM	Islamic
Sample 3	0	0	0	0	0	0	0	0	1	NM	Education
Sample 4	0	0	1	0	0	1	1	1	0	M	Education
Sample 5	0	0	1	0	0	1	1	1	0	M	Shopping
Sample 6	0	0	0	0	0	0	0	0	0	NM	Home

(NM=Non-malware; M=Malware)

C. Machine Learning Classifiers:

Fig.2 Data loading process of different classifiers

The pre-processed data set is loaded by creating local repositoryin RapidMiner on personal computer. The data is then processed to get the desired results using four well-known classifiers.

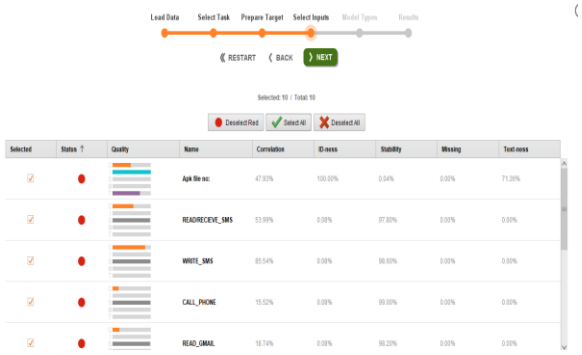


Fig.3 Data analyzation of classifiers

The data analyzation of applying different classifiers such as, Decision Tree, Random Forest, Gradient Boosted Trees and Support Vector machine on acquired dataset. The outcome is shown in Table 4 below.

IV. RESULTS

In this section, the results are regulated by using Machine Learning Classifiers provided by the latest version RapidMiner 9.8. We have used four well-known classifiers which are Random Forest, Decision Tree, Gradient Boosted Tree, and Support Vector Machine.

A. Processed Dataset of Android Applications:

The input vectors are monitored for training the dataset collected from different sources which are shown in figure 4. The dataset represents the application names and their permission keywords in columns. The value either 1 or 0 indicates the presence or absence of the permissions corresponding to the app name, respectively. We used supervised learning to train a dataset of 2500 applications by labeling _NM for non-malware and _M for malware nominating different categories. An example is given in Table 3 above.

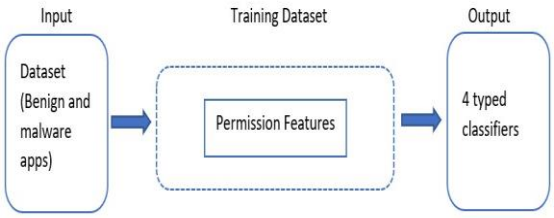


Fig.4 Training dataset under process

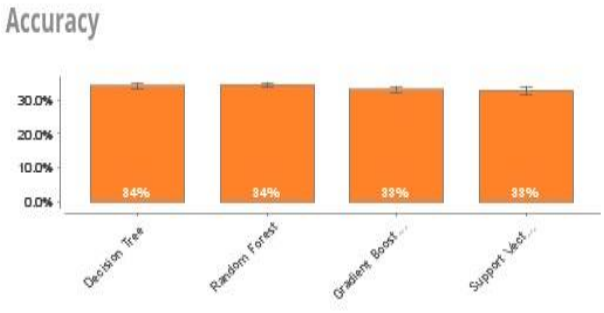


Fig.5 Accuracy of benign and malicious apps

As fig.5 illustrates the accuracy ratio of benign and malicious applications. Gradient Boosted Tree and Support VM classifier show a low accuracy ratio in comparison with the Random Forest and Decision tree.

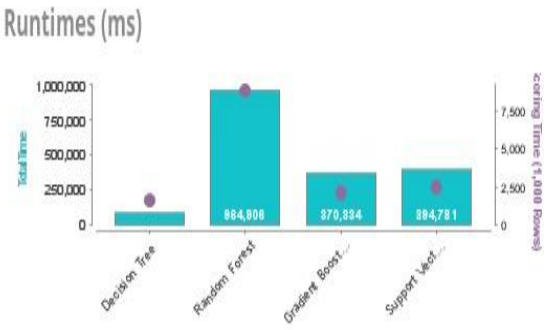


Fig.6 Runtime calculations of apps

Table 4 Comparison of different classifiers and their accuracy

Model	Accuracy
Decision Tree	34.00%
Random Forest	34.50%
Gradient Boosted Tree	32.90%
Support Vector Machine	32.90%

As the results given in table 4, the Decision Tree provides high accuracy in a very short time in comparison to other classifiers. The acquired %age demonstrates that we have very short problematic data in our dataset collection.

Entropy

To measure the purity of the dataset, entropy is calculated by using its formula is given in (1)

$$\text{Entropy} = \sum_{i=1}^{n} - p(s_i)\log_2 p(s_i) \quad (1)$$

$$\text{Entropy} = -\frac{2364}{2500} log_2 \frac{2364}{2500} - \frac{136}{2500} log_2 \frac{136}{2500} + \cdots$$

Entropy = 0.3

The less value of entropy shows an unbalanced dataset. That is why the classifiers applied in this work are achieving fewer accuracies.

V. CONCLUSION

In this paper, we have targeted the categories of android applications. There are hundreds of thousands of datasets available on the internet but not explicitly hitting the categorial part. We constructedown dataset.We aim to identify which category of Play Store contains more malware and which category contains the least malicious apps.One of the future work directions is the applicability of different techniques to make data balanced to improve the accuracy of the classifiers. Another extension of this work is to apply cross-validation techniques.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their insightful feedback on this work.

REFERENCES

[1] Jiang, X., Mao, B., Guan, J., & Huang, X. (2020). Android malware detection using fine-grained features. *Scientific Programming*, *2020*.

[2] Agrawal, Rishab, et al. "Android Malware Detection Using Machine Learning." *2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE)*. IEEE, 2020.

[3] Lu, Tianliang, et al. "Android Malware Detection Based on a Hybrid Deep Learning Model." *Security and Communication Networks* 2020 (2020).

[4] Zhou, Qingguo, et al. "A novel approach for mobile malware classification and detection in Android systems." *Multimedia Tools and Applications* 78.3 (2019): 3529-3552.

[5] Shahriar, Hossain, Mahbubul Islam, and Victor Clincy. "Android malware detection using permission analysis." *SoutheastCon 2017*. IEEE, 2017.

[6] Li, Jin, et al. "Significant permission identification for machine-learning-based android malware detection." *IEEE Transactions on Industrial Informatics* 14.7 (2018): 3216-3225.

[7] Ali, Kashif, Saleem, Muhammad and Ali, Baqar, "Detection and Prevention of Malware in Android Operating System." 1st International Conference on Computational Sciences and Technologies, INCCST, 2019.

[8] Zachariah, Raima, et al. "Android malware detection a survey." *2017 IEEE international conference on circuits and systems (ICCS)*. IEEE, 2017.

[9] Abubaker, Howida, Siti Mariyam Shamsuddin, and Aida Ali. "Analytics on malicious android applications." *International Journal of Advances in Soft Computing & Its Applications* 10.1, 2018.

[10] Yan, Ping, and Zheng Yan. "A survey of dynamic mobile malware detection." *Software Quality Journal* 26.3 (2018): 891-919.

[11] Bai, Chongyang, et al. "DBank: Predictive Behavioral Analysis of Recent Android Banking Trojans." *IEEE Transactions on Dependable and Secure Computing* (2019).

[12] Murtaz, Muhammad, et al. "A framework for Android Malware detection and classification." *2018 IEEE 5th International Conference on Engineering Technologies and Applied Sciences (ICETAS)*. IEEE, 2018.

[13] Mostafa, Ahmad H., Marwa MA Elfattah, and Aliaa AA Youssif. "An intelligent methodology for malware detection in android smartphones based on static analysis." *International Journal of Communications* 10 (2016).

[14] http://amd.arguslab.org/behaviors

[15] https://www.virustotal.com/gui/

[16] https://apkpure.com/app

[17] Ayed, Ahmed Ben. *Android Security: Permission Request Analysis Using Self-Organizing Maps in Android Malware Applications*. Diss. Colorado Technical University, 2017.

[18] Zhang, Hanqing, et al. "An efficient Android malware detection system based on method-level behavioral semantic analysis." *IEEE Access* 7 (2019): 69246-69256.