IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.-6, Issue-6, June 2018 E-ISSN: 2347-2693

Pruning and Ranking Based Classifier for Efficient Detection of Android Malware

Ramisetti Uma Maheswari[1], R Raja Sekhar[2]

[1,2] Dept. of CSE, JNTUCEA, Ananthapuramu, Andhra Pradesh, India

Available online at: www.ijcseonline.org

Accepted: 07/Jun/2018, Published: 30/Jun/2018

Abstract—Mobile devices that run Android operating system are widely used. The applications running in Android mobiles can have malicious permissions due to malware. In other words, Android applications might spread malware which can sabotage valuable data. Therefore it is essential to have mechanism to classify malware and benign mobile applications running in Android phones. Since Android mobile applications run in the confines of mobile devices and associated servers, it is very challenging task to detect Android malware. Many solutions came into existence to detect malware applications. Of late Abawajy et al. proposed a technique known as Iterative Classifier Fusion System (ICFS) which employs classifiers iteratively with fusion to generate a final classifier for effective detection of malware. They combined NB tree classifier, Multilayer perception and Lib SVM with polynomial kernel to achieve this. However, the system does not focus on reduction or pruning of Android application permissions so as to build a classifier that reduces time and space complexity. In the proposed system, a methodology is proposed that focuses on reduction or pruning of android application permissions and ranking them in order to build a classifier that reduces time and space complexity. The classifier modelled with best ranked permissions can be representative of all permissions as least significant permissions are pruned to reduce search space. This paper built a prototype application to demonstrate proof of the concept. The experimental results revealed that the proposed system performs better in improving detection accuracy besides precision and recall measures.

Keywords—Malware, malware detection technique, pruning, ranking

I. INTRODUCTION

Android malware has become a potential risk to mobile applications. Due to the increase in the usage of Android mobile applications in the world, the attackers target it to spread malware. Malware is the malicious software that can cause damage to a system of mobile device in terms of removing data or denying a service and so on. Malware can be one of the forms of cyber security threats. It has history of damaging potential applications in the real world. The cyber security threat landscape is increased drastically with the presence of Android malware. The rationale behind this is that people of all walks of life started using Android smart phones for virtually any operation including banking and shopping.

As the mobile smart phones and associated sensors produce huge amount of data known as big data, it became crucial to protect Android applications from malware. Big data has become an important buzzword and there are enterprises that depend on the business intelligence acquired from big data for decision making. In this context, it is important to have an efficient mechanism to detect and prevent Android malware. Many approaches came into exultance as found in the literature. They include signature based approaches [1], commercial malware detection methods [2], detection methods that also support Dalvik byte code transformations [4], data flow path based solutions [5] and [6], characterization of malware through system calls [9] and a hybrid approach that combines both API-calls and permission based approaches. In [12] an iterative approach is followed with multiple classifiers to detect malware. From the literature it is found that the methods are focusing more on accuracy of the solution rather than the computational complexity. In this paper we proposed an approach that focuses more on reducing computational complexity and increasing accuracy of the detection method. Our contributions are as follows.

- We proposed a framework to have a systematic approach in Android malware detection. It is based on the significant permission based permission reduction, pruning and ranking approaches.
- We proposed an algorithm known as Permission Significance-based Pruning for Android Malware Detection (PSP-AMD) to build a classifier that provides accuracy of detection and reduces computational complexity.
- We built a prototype application that demonstrates proof of the concept. The empirical results revealed the utility of the proposed solution which is light weight and focuses not only on the accuracy but also reduction of computational complexity.

Section I contains the introduction, Section II contains the related work, Section III contain the proposed methodology, and Section IV contain the conclusion and future scope.

II. RELATED WORK

This section provides review of literature on the malware detection methods and related works. As the Android mobile platform became popular, adversaries are targeting spreading of malware through Android mobile apps. There is a good survey on the current methods to detect malware in Android applications is found in [1]. There are signature based methods that are used to make use of malware signatures for detection. Signature based approaches are more prevalent among solutions available. Zhou et al. [2] studied commercial malware detection systems that are popular. Their studies revealed the fact that the detection rate of the method is between 20.2% and 78.6%. Similar kind of work is made in [3] where experimental results are provided for man popular anti-malware approaches associated with cloud. For many modern computers, the previous solutions were found inadequate. The work is to know whether the current anti-malware detection methods can handle Dalvik byte code transformations. Their experiments proved that there was further research needed to define methods to handle obfuscation. In [4] an advanced detection method that is behaviour-based is presented. It could prevent the vulnerability known as system-call injection. Asymptotic equi-partition property is used by their method in order to extract important call sequences to detect malware. A framework for automated analysis for detection of malware in Android applications is proposed. The framework identified malicious behaviours automatically by simulating intent broadcasts and user-interface events. Both static and dynamic analyses are combined in [5] while [6] make use of data flow path to distinguish benign apps from malicious ones. They made experiments on a large dataset and found that there was classification accuracy of 96% with benign apps and 98% with malware apps. In [7] a new approach is proposed to make use of system call in order to detect malware by characterizing malware behaviour. In [8] a static approach is proposed based on the API-call based and permission-based approaches. It uses a multi-classifier system and follows a collaborative approach based on probability theory that combines decisions of multiple classifiers. There are many approaches that exist in the literature. There are ensemble- classifiers that utilize approaches. In [9] pruning ensemble classifiers are studied. A multi-level system is proposed for detection of Android malware while focuses on an iterative multi-tier ensemble classifiers to do the same. In [10] another multi-classifier system is built with high accuracy. These solutions have used multi-classifier systems to increase accuracy in the detection of malware when compared with the solutions that used single-classifiers. The problem with these systems is that they are very huge and cannot be directly used for smart phone applications. They need more processing power and storage capacities besides causing much communication overhead as explored in [9]. These approaches focused on detection accuracy but they did not consider computational cost. There are some features of malware that are generally used to characterise them. In [11] multiple features are used to detect malware. They used feature selection algorithms to do so. There are some limitations in the feature selection methods too as they give emphasis on the algorithm that is specific. In [12] an iterative classifier fusion method is employed where multiple classifiers are involved. It is found to be complex and it can be optimized further. In this paper we proposed a light weight approach known as permission reduction, pruning and ranking based classifier for efficient detection of Android malware.

III. PROPOSED METHODOLOGY

In the proposed system, an alternative methodology is proposed. This consists of reduction or pruning of Android application permissions and ranking them in order to build a classifier that reduces time and space complexity. The classifier modelled with best ranked permissions can be representative of all permissions as least significant permissions are pruned to reduce search space. Thus the proposed system is expected to have better performance besides minimizing overhead.

METHODOLOGY

A dataset of around 5000 malware Android apps are collected. There are around 135 permissions that can be used by any Android app. The 135 permissions are taken as reference list of permissions. An excerpt from the list of Android permissions is given in Listing 1

```
<Uses-permission android: name="android .Permission .INTERNET"/>
<Uses-permission android: name="android .Permission. READ_PHONE_STATE"/>
<Uses-permission android: name="android .Permission. READ_LOGS"/>
<Uses-permission android: name="android .Permission .VIBRATE"/>
<Uses permission android: name="android .Permission.
RECEIVE_BOOT_COMPLETED"/>
<Uses-permission android: name="android .Permission .WAKE_LOCK"/>
<Uses-permission android: name="android .Permission. ACCESS_NETWORK_STATE"/>
```

Listing 1: Permissions used by one of the malicious app

Each dataset contains a list of permissions used. Once the dataset is loaded, the proposed system takes all malicious app names and corresponding permissions. Permission reduction, pruning and ranking based approach are used to build a classifier. Initially 135 total permissions are available. These permissions are subjected to pruning. Any permission which has least usage in the malicious apps is removed from the list. That way some of the permissions are removed from the master list of permissions. Then association rule mining is made on the malicious permissions of all apps. When there are associations (multiple permissions repeatedly occurring in apps), one of them will be treated as representative for all permissions in

the given association while others are pruned from the master list of permissions.

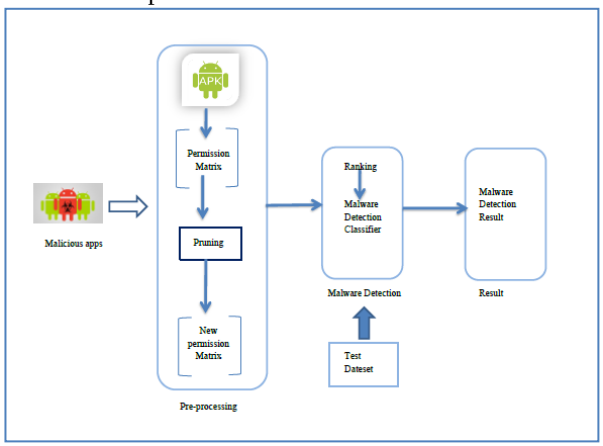

Figure 1: Overview of the proposed methodology.

Afterwards, the ranking of the remaining permissions in the master list is made based on the permissions present in all malicious apps. The best ranked permissions are retained in the master list while the poorly ranked ones are removed. Then a classifier is built to model malicious behaviour of Android apps. This classifier is used to test new apps to know whether they are malicious or genuine. The pseudo code of the proposed algorithm known as Permission Significance-based Pruning for Android Malware Detection (PSP-AMD) is as shown below

Algorithm: PSP-AMD
Inputs: Malware apps M, master list of permissions MP
Output: Classifier for malware detection

1 Initialize malicious application permissions vector AP
2 Initialize map for holding apps and list of permissions MAP
3 For each malware app m from M
4 Extract permissions from m into AP
5 Add app name and AP to MAP
6 End For
7 For each permission p from MP
8 Analyze MAP for the presence of p
9 Remove p from MP if it has negligible frequency
10 End for
11 Perform association rule mining on permissions of malicious apps (MAP)
12 Prune representative permissions from MP
13 For each permission p from MP
14 Perform ranking for p in the MAP
15 Prune the permission p if its ranking is negligible
16 End for
17 Build a classifier using MP which contains pruned list of master list of permissions
18 Classifier is applied to new app to know whether it is malicious

The algorithm is built based on the proposed architecture as shown in Figure 1.

The algorithm is meant for building a classifier based on the significant permission selection pruning of unwanted permissions besides ranking. Two matrices representing malware apps (M) and benign apps (B) are used. The difference between them is computed as follows. The threshold value for the difference is set to∈.

$$\frac{\text{Size } (M_j) - \text{ size } (B_j)\ |}{\min\left(\text{Size } \left(M_j\right),\ \text{size } \left(B_j\right)\right)} < \in$$

If the difference is very smaller than the threshold, it is important to implement the pruning method as follows.

$$R(P_j) = \frac{\sum_i M_{ij} - \sum_i B_{ij}}{\sum_i M_{ij} + \sum_i B_{ij}}$$

Then the balancing of matrices in spite of changes in size is done using the following equation.

$$SB\left(P_j\right) = \frac{\sum_i B_{ij}}{\text{size } \left(B_j\right)} * \text{ Size } \left(M_j\right)$$

Here the SB (P_j) denotes the support of jet permission pruning is implemented as the modified equation shown below.

$$R(P_j) = \frac{\sum_i M_{ij} - S_B\left(P_j\right)}{\sum_i M_{ij} + S_B\left(P_j\right)}$$

The algorithm is implemented with a prototype application shows in the following section. The algorithm results showed that the proposed methodology is effective as it considering the significant permission based pruning and ranking to build a classifier. The proposed classifier is found to be computationally effective.

PROTOTYPE AND EXPERIMENTAL RESULTS

We built a prototype application to demonstrate proof of the concept. The application is built using Java platform. Java Swing API is used to have intuitive user interface while the IO mechanisms are used to deal with file handling. The detection of malware is preceded by the classifier building with proposed pruning approach.

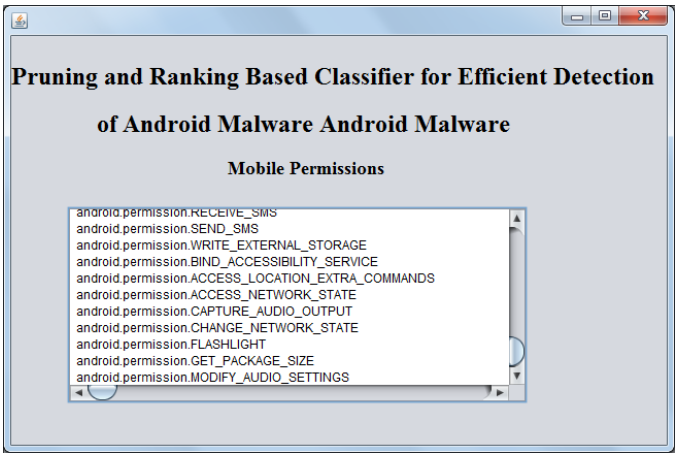

Figure 2: Shows Android mobile app permissions.

As can be shown in Figure 2, there are around 135 Android permissions taken as initial input. Afterwards based on the significance in detecting malware, they are pruned further.

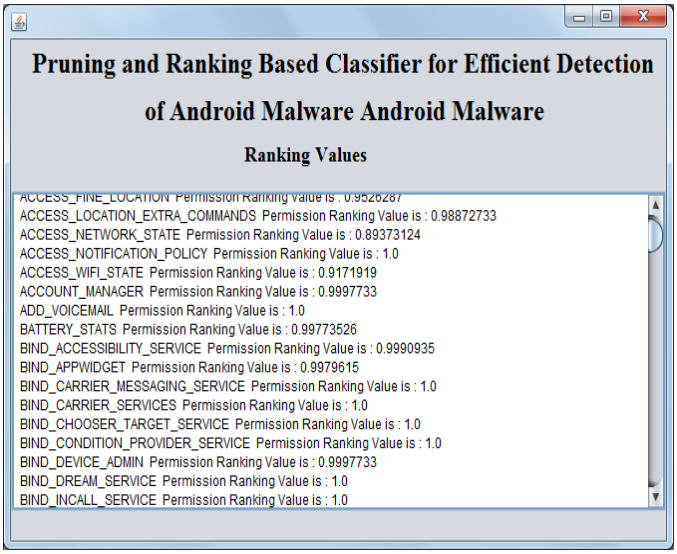

Figure 3: After completion of pruning the remaining permission are given ranking.

As shown in Figure 3, it is evident that every permission is given ranking. The permissions that remained after pruning process are considered to give ranking.

EVALUATION

We evaluated the proposed methodology with an empirical study. The detection accuracy of different algorithms is presented in Table 1.

Table1: Detection accuracy comparison.

Algorithms	**Detection Accuracy**
Lib SVM	0.6
J48	0.8
ICFS	0.93
PSP-AMD	0.97

As can be seen in Table 1, the detection accuracy of the algorithm is compared. The proposed algorithm exhibited 0.97 Accuracy in detection of malware. It is comparatively better performance when ICFS, J48 and Lib SVM are considered

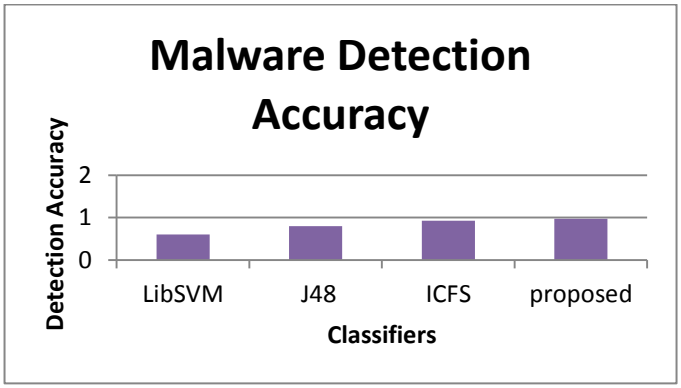

Figure 4: Malware detection accuracy.

As presented in Figure 3, it is evident that there are many classifiers compared with the proposed one. The Lib SVM showed lest accuracy while proposed method showed highest accuracy.

Table2: Evaluation of the proposed algorithm.

No. of Features	Precision	Recall
5	91.29%	83.90%
10	90.21%	90.24%
15	90.21%	91.21%
20	90.47%	91.65%
25	90.64%	91.77%
30	91.27%	90.58%
35	91.83%	90.05%
40	96.28%	86.19%
45	96.28%	85.94%
50	96.34%	85.82%
55	96.35%	85.80%
135	98.81%	83.73%

The proposed algorithm is evaluated with measures like precision and recall. A shown in Table 2, the precision and recall values are presented against number of features considered.

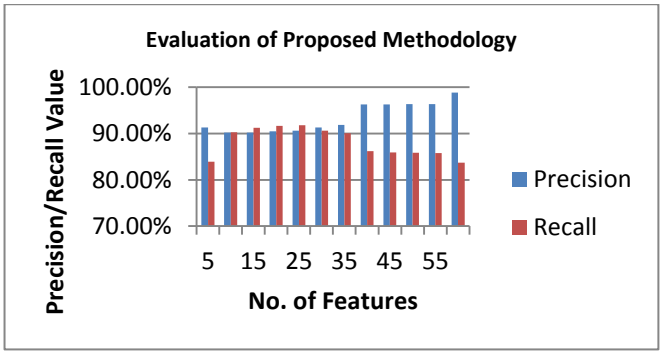

Figure 5: Precision and recall of the proposed algorithm against number of features.

As can be seen in Figure 4, the number of features is presented in horizontal axis. The values are taken from 5 to 135 incremental by 5 gradually. The precision and recall values showed in vertical axis are showing the performance of the proposed method. The precision and recall will have trade-offs. It does mean that when precision is decreasing recall increase and vice versa.

IV. CONCLUSION AND FUTURE SCOPE

Android malware became potential risk to smart phone applications. The rationale behind this is the unprecedented popularity of Android platform for mobile phones. In this paper we studied different Android malware detection approaches in the literature and found the need for an approach that is cost effective besides increasing accuracy of prediction. We proposed a framework for building a classifier that takes care of malware detection. The proposed approach is based on the permissions of Android mobile apps. We proposed an algorithm known as Permission Significance-based Pruning for Android Malware Detection (PSP-AMD) that identifies significance of permissions based on the given dataset and perform pruning and ranking in order to build a final model that can be used to detect Android malware. Experiments are made with malware dataset collected from Virus Total. We built a prototype application to demonstrate proof of the concept. The empirical results revealed that the proposed solution is effective in detection of Android malware. In future we investigate further in the permission pruning and ranking to optimize our solution. We also find it interested to work with other datasets to generalize our findings.

REFERENCES

[1] P. Faruki, A. Bharmal, V. Laxmi, V. Ganmoor, M. S. Gaur, M. Conti, and M. Rajarajan, "Android security: A survey of issues, malware penetration, and defences," IEEE Communications Surveys and Tutorials, vol. 17, pp. 998–1022, 2015.

[2] Y. Zhou and X. Jiang, "Dissecting Android malware: Characterization and evolution," in Proceedings of the 33rd IEEE Symposium on Security and Privacy, San Francisco, CA, pp. 95–109, 2012.

[3] J. Walls and K.-K. R. Choo, "A review of free cloud-based antimalware apps for Android," in Proceedings of 2015 14th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, Trust Com 2015, vol. 1, pp. 1053–1058, 2015.

[4] S. Naval, V. Laxmi, M. Rajarajan, M. S. Gaur, and M. Conti, "Employing program semantics for malware detection," IEEE Transactions on Information Forensics and Security, vol. 10, no. 12, pp. 2591–2604, 2015.

[5] P. Faruki, S. Bhandari, V. Laxmi, M. Gaur, and M. Conti, "Droid analyst : Synergic app framework for static and dynamic app analysis," Studies in Computational Intelligence, vol. 621, pp. 519–552, 2015.

[6] L. Sinha, S. Bhandari, P. Faruki, M. S. Gaur, V. Laxmi, and M. Conti, "Flow Mine : Android app analysis via data flow," in Proceeding of the 13th IEEE Annual Consumer Communications and Networking Conference, CCNC 2016, pp. 435–441, 2016.

[7] J. Abawajy, M. Chowdhury, and A. Kelarev, "Hybrid Consensus Pruning of Ensemble Classifiers for Big Data Malware Detection," IEEE Transaction on Cloud Com put 3(2):111, 2017.

[8] S. Sheen, R. Anitha, and V. Natarajan, "Android based malware detection using a multi feature collaborative decision fusion approach," Neuro computing, vol. 151, pp. 905–912, 2015.

[9] S. Naval, V. Laxmi, M. S. Gaur, S. Raja, M. Rajarajan, and M. Conti, "Environment-reactive malware behaviour: Detection and categorization," in Data Privacy Management, Autonomous Spontaneous Security, and Security Assurance, ser. LNCS, vol. 8872, pp. 167–182, 2015.

[10] F. Daryabar, A. Dehghantanha, F. Norouzi, and F. Mahmoodi, "Analysis of virtual honey net and vlan-based virtual networks," in International Symposium on Humanities, Science and Engineering Research, SHUSER 2011, pp. 73–77, 2011.

[11] K. Zhao, D. Zhang, X. Su, and W. Li, "Fest: A feature extraction and selection tool for Android malware detection," in 20th IEEE Symposium on Computers and Communication, ISCC 2015, pp. 714–720, 2015

[12] J. Abawajy, A. Kelarev "Iterative Classifier Fusion System for the Detection of Android Malware". IEEE Transactions on Big Data, Vol. 5, No. 4, p1-12, 2017.

Authors Profile

Ms Ramisetti Uma Maheswari recieved the Bachelor of Computer Science engineering degree from Sri Padmavati Mahila Visvavidyalayam, tirupati, andhra pradesh, india in 2016.She is currently pursuing Master of Computer Science engineering from JNTUCEA, Ananthapuramu in year 2018.

Dr R Raja Sekhar received the Ph.D. degree from JNTU College of Engineering, Ananthapuramu, Andhra Pradesh, India in 2018. He is currently an Associate Professor with the Computer science and Engineering, JNTU College of Engineering. His current research interests Mobile Ad-hock Networks, Compilers and algorithms include digital multimedia forensics.

