International Journal of Latest Trends in Engineering and Technology
Vol.(11)Issue(2), pp.071-077
DOI: http://dx.doi.org/10.21172/1.112.11
e-ISSN:2278-621X

DETECTION OF MALWARE APPS AND RANKING IN PLAYSTORE

Dr. P.Maragathavalli[1], P.R.Chandru[2]

Abstract- Google Play which is the most popular Android app market in Fraudulent behaviours is the fuel search rank abuse and malware proliferation. But Google Play App recommendation is not efficient in recent days. Genuine apps do not have a proper Ranking in playstore.This is all due to the Fraudulent developers who frequently exploit crowdsourcing sites. This behaviour is called as search rank fraud. The proposed system deals with slight variation apps which means some fraudulent developer may release hundreds or even thousands of nearly identical apps with the different set of permissions (may be created using app maker tools) that provide the same functionality with slight variation and the non-programmers and non-developers may also create apps by app maker tools. This project uses k-means technique and Kernel weight method which involves comparison of permission weights of the apps, block the malware apps and thus improve ranking of genuine apps.
Keywords – Malware, Search Rank, Playstore, Genuine Apps, Fraud Detection.

1. INTRODUCTION

Fraudulent developers frequently exploit crowdsourcing sites (e.g., Freelancer,Fiverr, BestAppPromotion) to hire teams of willing workers to commit fraud collectively, emulating realistic, spontaneous activities from unrelated people (i.e. Crowdturfing). We call this behaviour "search rank fraud". For instance, PlayStore uses the Bouncer system to remove malware.This work is built on the observation that fraudulent and malicious behaviour's leave behind tell-tale signs on app markets. They uncover these nefarious acts by picking out such trails. For instance, the high cost of setting up valid Playstore accounts forces fraudsters to reuse their accounts across review writing jobs, making them likely to review more apps in common than regular users. The proposed system deals about slight variation apps which means some fraudulent developer may release hundreds or even thousands of nearly identical apps with the different set of permissions (may be created using app maker tools) that provide the same functionality with slight variation and the non-programmers and non-developers may also create apps by app maker tools. The list of items in Google Playstore is shown in Figure 1.

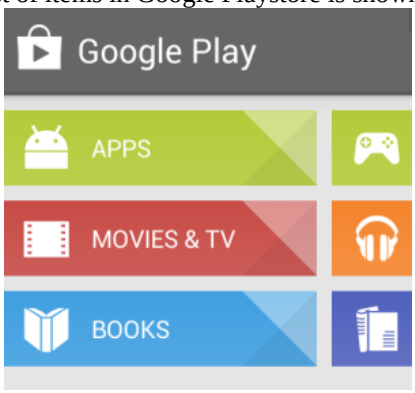


Figure 1. Google PlaystoreSample

To detect fraud and malware, the input parameters considered were mobile apps, permissions, and ratings. In this proposed work the permissions of app for each category is clustered by K-means method from the data set and after this each category is assigned to set of relevant permissions. Each permission is given some weight which is then verified (kernel weight method) with the apps which were newly uploaded by the developer. After verifying, the apps which are uploaded with irrelevant permissions were blocked based on their permission weights and thus increasing ranks of genuine apps. Ranking of apps is done by the install counts and also only for active apps. After blocking of apps, it is no more visible to users to download.

[1] Assistant Professor/IT, Pondicherry Engineering College
[2] Student Member/IT, Pondicherry Engineering College

2. EXISTING SYSTEM

The existing system relevant to Google Playstore and malware apps detection are studied and major works are given below:

Paper 1: Search Rank Fraud and Malware Detection in Google Play

In this paper, they identified malware apps by using app executable and permission analysis. They introduced FairPlay, a novel system that discovers and leverages traces left behind by fraudsters, to detect both malware and apps subjected to search rank fraud. FairPlay correlates review activities and uniquely combines detected review relations with linguistic and behavioural signals from Google Play app data in order to identify suspicious apps. FairPlay has been used to achieve over 95 percent accuracy in classifying gold standard datasets of malware, fraudulent and legitimate apps. They show that 75 percent of the identified malware apps engage in search rank fraud. FairPlay has discovered hundreds of fraudulent apps that currently evade Google Bouncer's detection technology. In addition to that FairPlay helped the discovery of more than 1,000 reviews, reported for 193 apps, that reveal a new type of "coercive" review campaign: users are harassed into writing positive reviews, and install and review other apps.

In this paper have introduced FairPlay, a system to detect both fraudulent and malware Google Play apps. Our experiments on a newly contributed longitudinal app dataset, have shown that a high percentage of malware is involved in search rank fraud; both are accurately identified by FairPlay. In addition, we showed FairPlay's ability to discover hundreds of apps that evade Google Play's detection technology, including a new type of coercive fraud attack.

Limitations: To increase the accuracy of FairPlay by detecting Slight VariationApps.

Paper 2: Android Malware Detection using Permission Analysis

Android applications are widely used by millions of users to perform many different activities. However, many applications have been reported to be malware performing activities not matching with their expected behaviour's (e.g., sending SMS message to premium numbers). The existing relevant approaches that identify these applications (malware detection technique) suffer from performance issues where the occurrence of false negatives (FN) remain high. These approaches are not scalable and provides little flexibility to query based on a set of suspicious permission to identify a set of highly relevant known anomalous applications to examine further. This paper proposes a new approach to reduce the number of apps that must be sandboxed in order to determine if they are malicious. We first examine the source of applications to identify list of permissions and find a set of highly close and relevant applications from a given set. The identified most relevant application category's permission is then checked to find if there is significant overlapping or not to identify an application as suspected anomalous. We apply Latent Semantic Indexing (LSI) to identify malware application. Their initial evaluation results suggest that the proposed approach can identify malware applications accurately.

This work proposed a new approach to detect anomalous Android applications by identifying the most relevant category of permissions to match from and then confirming behaviour's in an emulator environment. The relevant category is identified using Latent Semantic Index (LSI) analysis. The approach has the potential to discover new anomalous applications. We evaluate the approach using two open source malware repositories. The initial results show that our approach can detect malware applications. Our future work includes evaluating more applications and query types, and apply the approach to address other security vulnerabilities.

Limitations: Evaluation of more applications and query types, and apply the approach to address other security vulnerabilities.

Paper 3: A Longitudinal Study of Google Play

The difficulty of large-scale monitoring of app markets affects our understanding of their dynamics. This is particularly true for dimensions such as app update frequency, control and pricing, the impact of developer actions on app popularity, as well as coveted membership in top app lists. In this paper, they performed a detailed temporal analysis on two datasets one consisting of 160000 apps and the other with 87223 newly released apps. They have monitored and collected data about these apps over more than six months. Their results show that a high number of these apps have not been updated over the monitoring interval. They observed that infrequently updated apps significantly impact the median app price. There is no correlation between app price and the download count. They show that apps that attain higher ranks have better stability in top app lists. They show that app market analytics can help detect emerging threat vectors, and identify search rank fraud and even malware. Also, they discussed the research implications of app market analytics on improving developer and user experiences.

This paper studies temporal patterns in Google Play, an influential app market. They usedcollected data from more than 160000 apps daily over a six-month period, to examine market trends, application characteristics, and developer behaviour in real-world market settings. Their work provides insights into the impact of developer levers (e.g., price, permissions requested, and update frequency) on app popularity. They have givenfuture directions for integratinganalytics insights into developer and user experiences. They introduced novel attack vectors on app markets and discussed future detection directions.

Limitations: Changing of app permissions to confirm their statistical significance in detecting search rank fraud and malware.

Paper 4:Identifying Indicators of Fake Reviews Based on Spammer's Behaviour Features
In the enterprise marketing process, on-line data plays more and more important role. As a type of data, spam (or fake) reviews of the products, however, have been seriously affecting the reliability of both decision making and data analysis of the enterprise. To detect spam reviews, the paper presents a set of opinion spam detection's identification indicators based on behaviour features of the spammer. Two algorithms are then proposed to recognize similar reviews and relevant reviews from all reviews. Compared to the traditional algorithm, our review identification algorithm achieves shorter execution time. More importantly, the proposed algorithm for recognizing relevant reviews can be used to analyse the relevancy between the review content and the given review topic by using the automatic word segmentation technique. The Experimental results show that the number of fake reviews by our algorithms is higher than that of the traditional algorithm. Moreover, we found that about 46% of the mobile phone reviews on the Amazon website were irrelevant to the product's topic, and 54.7% of the reviews were similar to other reviews.
Fake reviewers have some common behaviour features. Thus, based on these features, the paper constructs eight identification indicators to detect spam reviews of the product on e-commerce platforms. To implement the proposed identification indicators, we present two algorithms to recognize both similar reviews and relevant reviews, respectively. Then, we apply our algorithms to 214,005 mobile phone reviews that are crawled from Amazon.com. The experimental results show that our algorithms have higher efficiency to identify fake reviews than the traditional algorithm. In the future, we will improve our relevancy algorithm to recognize more non-relevant reviews.
Limitations: To improve relevancy algorithm to recognize more non-relevant reviews.

Paper 5: PUMA: Permission Usage to Detect Malware in Android
The usage of mobile devices has increased in our lives providing almost the same functionality as a PC. Android operating system with more number of applications is increasing today. Google has its own Android applications which are offered to users are easily prone and misuse. Malware writers usually insert malicious applications into this market. In this paper, theyintroduced a new method for detecting malicious Android applications through machinelearning techniques by analysing the extracted permissions from the application itself.
Permissions are the most recognisable security feature in Android. User must accept them in order to install the application. In order to validate their method, they collected 239 malware samples of Android applications. Then, they extracted the features for each application and trained the models, evaluating each configuration using the Area Under ROC Curve (AUC). They obtained a 0.92 of AUC using the Random Forest classifier as a result. Nevertheless, there are several considerations regarding the viability of our approach. Furthermore, despite the high detection rate, the obtained result has a high false positive rate. Consequently, this method can be used as a first step before other more extensive analysis, such as a dynamic analysis.
Limitations: Forensics tools for Android applications should be developed in order to obtain new features.
From the existing papers, the limitations are analysed and listed. Finally, detection of malware apps and ranking by using appropriate techniques have been taken as a proposal.

3. PROPOSED SYSTEM

In this system, it consists of user interface, google play admin and developer interface to detect malware apps based on permission clustering and verifying permission weights and giving rank to apps. This system deals about slight variation apps which means some fraudulent developer may release hundreds or even thousands of nearly identical apps with the different set of permissions (may be created using app maker tools) that provide the same functionality with slight variation and the non-programmers and non-developers may also create apps by app maker tools. In this proposed work the permissions of app for each category is clustered by K-means method from the data set and after this each category is assigned to set of relevant permissions. Each permission is given some weight which is then verified (kernel weight method) with the apps which were newly uploaded by the developer f(i,j) = WkKk(ai, aj).
After verifying, the apps which are uploaded with irrelevant permissions were blocked based on their permission weights and thus increasing ranks of genuine apps. Ranking of apps is done by the install counts and also only for active apps. After blocking of apps, it is no more visible to users to download. k-means algorithm is used to classify the large data set. Initially the work of k-means algorithm is to classify the cluster data by comparing the similarities between them. In this case k-means is designed in a way to cluster the data set whose values are equal. Initially all permissions considered are given a value and stored in an array. While class a permission for a category the k-means algorithm checks the data set for equal values and cluster then in a group and these are the deserved permissions for a app in a category. The proposed system architecture with developer, admin, user and their relationships is shown in Figure 2.

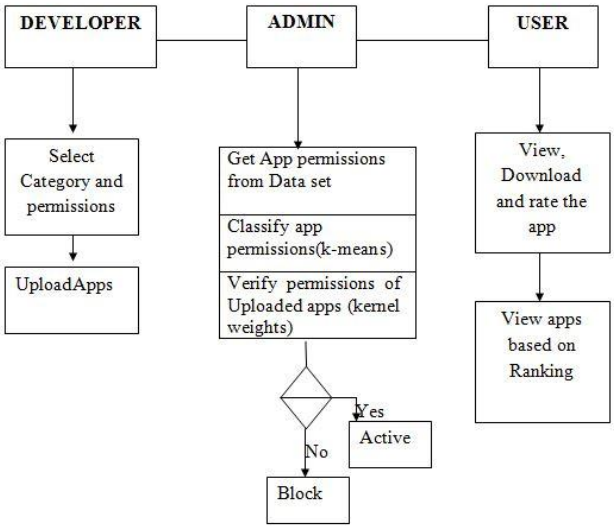


Figure 2. Proposed System Architecture

Further while using kernel weightmethod, it calculates the weight of the permission of the uploaded app and compare them with the weight of permission already classified using k-means. If the weight is not equal the app is not allowed to upload and is blocked. And so if the weight is equal the app is uploaded and it is viewed on user interface for downloading.
Ranking of apps is done on the user side by the download count of the apps. The app is given a first rank based on the download count. Such a way it gives user a clear information regarding the genuine apps and the rating interface is given for user rating.

Input Data Representation:
Mobile Apps
The number of apps uploaded by the developer for user download. It is checked and verified using the algorithms implemented.
Permissions
The permissions are classified using k-means method from a large data set uploaded. The permissions are classified for each category and a weight is calculated for them.
Ratings
Rating interface is developed at user side for ranking apps. This gives a chance for user to give their opinion regarding the apps.

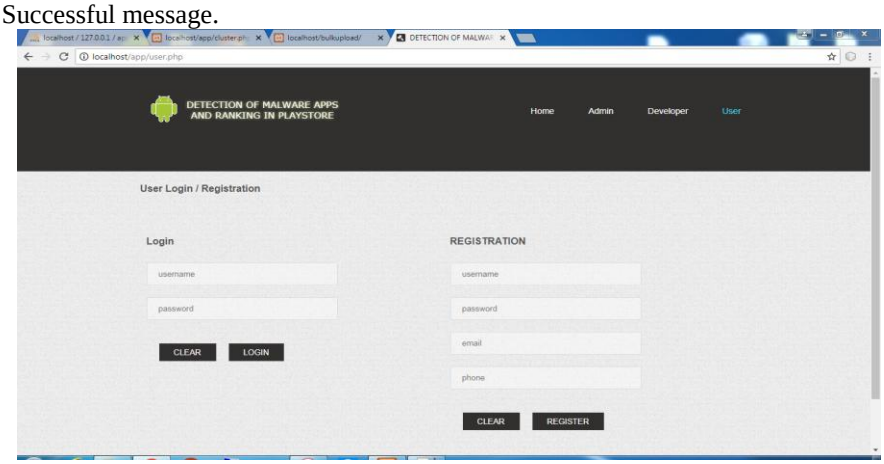
Techniques Used for Detecting Malware Apps:
K-means technique
Slight variation app detection Technique (Kernel weight method)
In this technique permission of apps are compared and those apps which are not equal weight are blocked.

Output Parameters:
Malware Count
The number of malware apps blocked by google play admin.

Rank Fraud Count
The number of fraud apps with rank zero (assigned by admin).

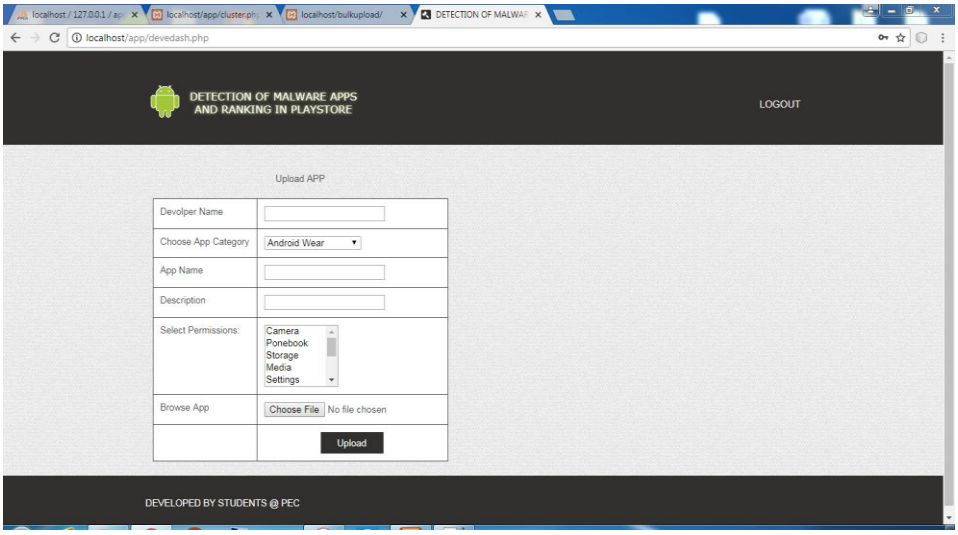
4. EXPERIMENTATION
There are three provisions given to the registered user. They can view, upload and download apps based on the level of privileged user. Based on the ratings, the system categorizes the user and rank them. The malware apps are found out based on the frequency of the accessed apps and the access privilege like either block or active mode is given to the appropriate end user. The following are the major steps involved in implementation process:
User registration
Upload/download
Rating

App categorization either malware or not
Ranking
Register and Login
In this module, app developer and app users register with Google Play admin with username, password, name, mobile number and address.After the registration, they access their accounts.
Input: username, password, name, mobile number and address.
Output: Registration Successful message.

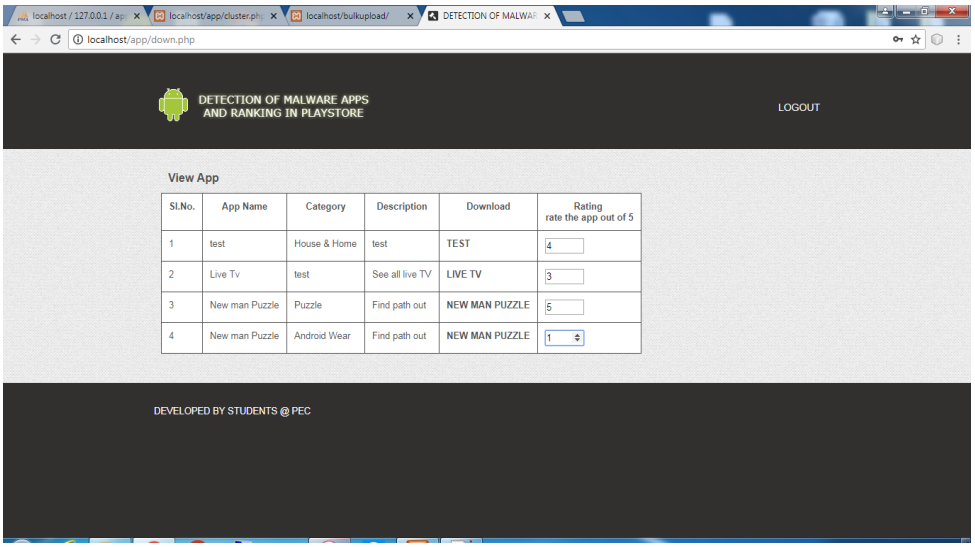


Upload Apps
In this module, developer uploads his apps. The developer chooses the category and the permissions interface is opened. The developer needs to select an appropriate permission for the apps. Then further he needs to upload the apps. The apps with proper permissions are only uploaded and the fraud apps are blocked and user is blocked from uploading apps.

Input: App, App Icon, App Metadata, Permission
Output: Uploaded Success Message.

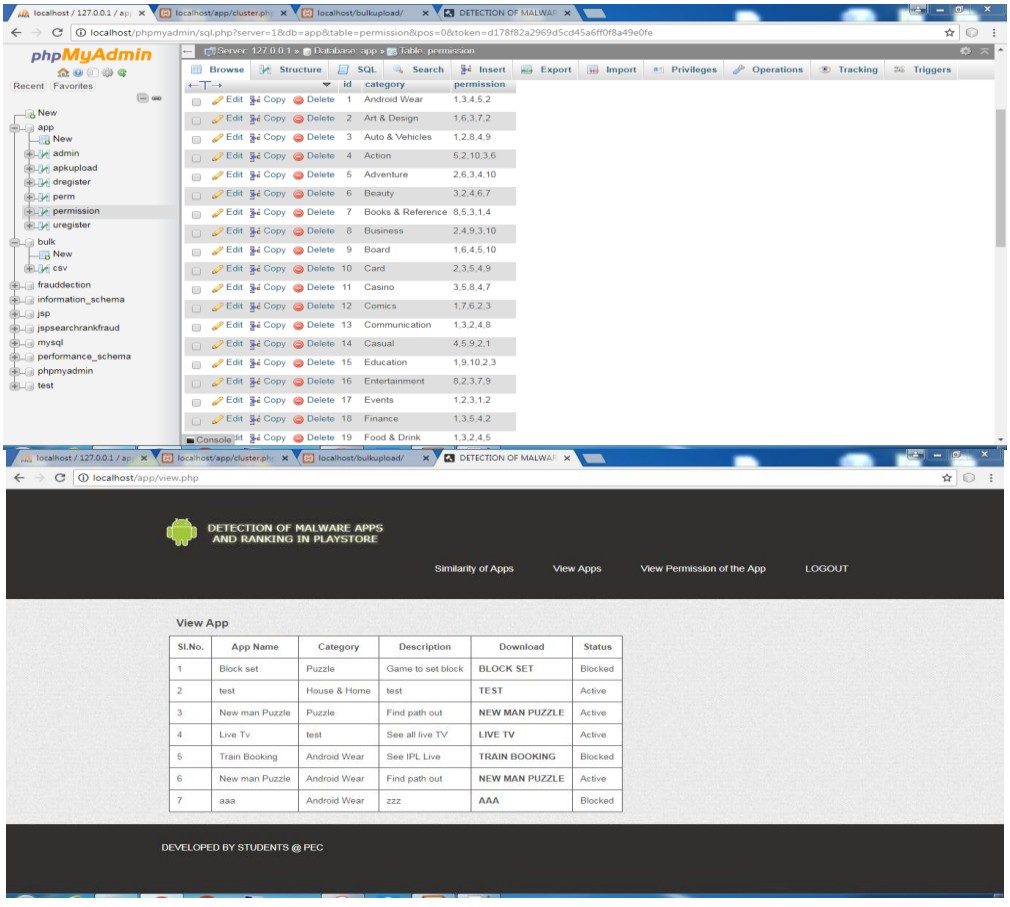


Download & Give Reviews and Ratings
In this module, app users download any desirable apps and install his Smartphone.
After installation, he uses this app and gives the reviews and ratings about that particular app.
Input:App, App Metadata
Output:Downloaded Successful, Give Reviews and Ratings



Classification of Apps Permission by App Category and Blocking Malware Apps

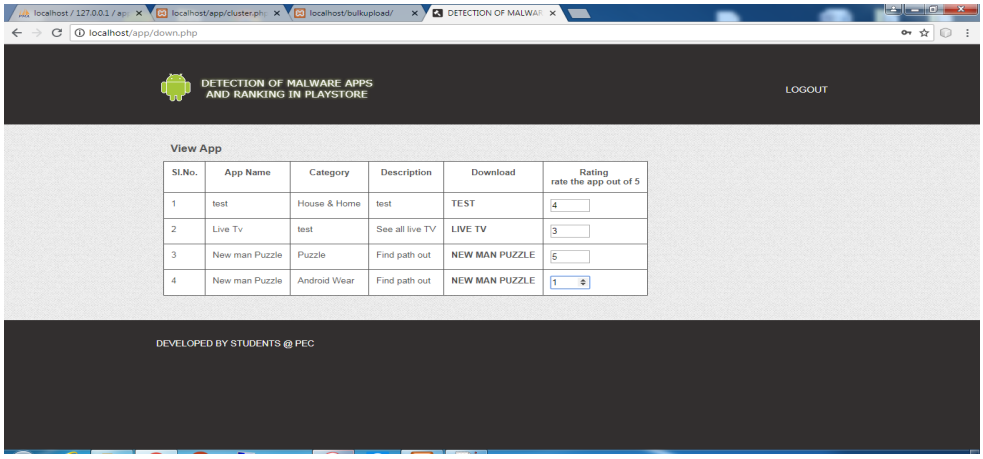
In this module, the permissions for the app is classified based on their categories. Here k-means clustering algorithm classifies the permissions from the given data set. Each permissionhas a weight and it's calculated by admin. After developer uploads his app with certain permissions the weights are compared to check whether the uploaded apps are genuine apps. The fraud apps will be blocked and the genuine apps will be displayed in the user interface for download.



In this module, the apps with irrelevant permissions are compared and blocked and developers are also blocked from uploading apps. The rank of the apps is made to zero.

Ranking of Apps

In this module user can view the apps which are active and can give ratings to each app.The blocked apps will not be visible to the users. Ranking of the apps is done based on install counts and the app which is installed more times will be displayed first to download.



5. CONCLUSION

This paper proposed a technique to compare weight of the apps permission for detecting fraud apps and making their rank zero and further blocking the developer from uploading malware apps. Further, the significant improvement inthe detection of malware apps and rank genuine apps with different access permissions. We demonstrated the effectiveness of the proposed technique using AppPermissions dataset which is verified using weightage parameter. A significant improvement in security by blocking the malware apps in the initial stage of uploading and ranking is done effectively by using download count.

6. REFERENCES

[1] Mahmudur Rahman, Mizanur Rahman, Bogdan Carbunar, and DuenHorng Chau, "Search Rank Fraud and Malware Detection in Google Play", IEEE Transactions on Knowledge and Data Engineering, Vol. 29, No. 6, pp. 1329–1342, Jun 2017.

[2] Z. Miners, "Report: Malware-infected Android apps spike in the Google Playstore", PC World, Feb 2016.

[3] J. Sahs and L. Khan, "A Machine Learning Approach to Android Malware Detection", in Proceedings of International Conference on European Intelligence and Security Informatics, pp. 141–147, Aug 2012.

[4] H. Peng, et al., "Using Probabilistic Generative Models for Ranking Risks of Android Apps", in Proceedings of ACM Conference on Computer and CommunicationsSecurity, pp. 241–252, Oct 2012.

[5] A. Tamersoy, K. Roundy, and D. H. Chau, "Guilt by Association: Large Scale Malware Detection by Mining File-Relation Graphs", in Proceedings of 20th ACM SIGKDD International Conference on Knowledge Discovery Data Mining, pp. 1524–1533, Aug 2014.

[6] S. Yerima, S.Sezer, and I. Muttik, "Android Malware Detection using Parallel Machine Learning Classifiers", in Proceedings of NGMAST, pp. 37–42, Sep 2014.

[7] D.H. Chau, C.Nachenberg, J. Wilhelm, A. Wright, and C. Faloutsos, "Polonium: Tera-Scale Graph Mining and Inference for Malware Detection", in Proceedings of SIAM International Conference on Data Mining, pp. 131–142, Apr 2011.

[8] Google Play. [Online]. Available: https://play.google.com.

[9] Fraud detection in social networks. Available: https://users.cs.fiu.edu/carbunar/caspr.lab/socialfraud.html.