SEARCH FRAUD RANK AND MALWARE DETECTION IN GOOGLE PLAY

Chigurla Yogender [1], Assistant Prof. Mr. E Srikanth Reddy[2]

1. **M.Tech Scholar, Department of CSE, Vaageswari College of Engineering, Karimnagar, Telangana, India**
   Email:Yogender.chigurla@outlook.com.
   Mobile: +91 9000472787
2. **Associate Professor, Department of CSE , Vaageswari College of Engineering, Karimnagar, Telangana, India-**
   Email: srikanthmanchikatla24@gmail.com,
   Mobile: +91 9849439438

ABSTRACT

In this paper, we propose Fraudulent behaviors in Google Play, the most popular Android app market, fuel search rank abuse and malwareproliferation. To identify malware, previous work has focused on app executable and permission analysis. In this paper, we introduceFairPlay, a novel system that discovers and leverages traces left behind by fraudsters, to detect both malware and apps subjected tosearch rank fraud. FairPlay correlates review activities and uniquely combines detected review relations with linguistic and behavioralsignals gleaned from Google Play app data (87K apps, 2.9M reviews, and 2.4M reviewers, collected over half a year), in order toidentify suspicious apps. FairPlay achieves over 95% accuracy in classifying gold standard datasets of malware, fraudulent andlegitimate apps. We show that 75% of the identified malware apps engage in search rank fraud. FairPlay discovers hundreds offraudulent apps that currently evade Google Bouncer's detection technology. FairPlay also helped the discovery of more than 1,000reviews, 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.

I.INTRODUCTION:

The commercial success of Android app markets such as Google Play [1] has made them a lucrative medium for committing fraud and malice. Some fraudulent developers deceptively boost the search ranks and popularity of their apps (e.g., through fake reviews and bogus installation counts) [2], while malicious developers use app markets as a launch pad for their malware [3, 4, 5, 6]. Existing mobile malware detection solutions have limitations. For instance, while Google Play uses the Bouncer system [7] to remove malware, out of the 7, 756 Google Play apps we analyzed using VirusTotal [8], 12% (948) were flagged by at least one anti-virus tool and 2% (150) were identified as malware by at least 10 tools (see Figure 3a). Previous work has focused on dynamic analysis of app executables [9, 10, 11] as well as static analysis of code and permissions [12, 13, 14]. However, recent Android malware analysis revealed that malware evolves quickly to bypass anti-virus tools [15].

In this paper, we seek to identify both malware and search rank fraud targets in Google Play. This combination is not arbitrary: we posit that malicious developers resort to search rank fraud to boost the

impact of their malware. Unlike existing solutions, we build this work on our observation that fraudulent and malicious behaviors leave behind telltale signs on app markets. We uncover these nefarious acts by picking out such trails. For instance, the high cost of setting up valid Google Play accounts forces fraudsters to reuse their accounts across review writing jobs, making them likely to review more apps in common than regular users. Resource constraints can compel fraudsters to post reviews within short time intervals. Legitimate users affected by malware may report unpleasant experiences in their reviews. Ramps in the number of "dangerous" permissions requested by apps may indicate benign to malware (Jekyll-Hyde) transitions.

II.EXISTING SYSTEM:

- Google Play uses the Bouncer system to remove malware. However, out of the 7, 756 Google Play apps we analyzed using Virus Total, 12% (948) were flagged by at least one anti-virus tool and 2% (150) were identified as malware by at least 10 tools.
- Sarma et al. use risk signals extracted from app permissions, e.g., rare critical permissions (RCP) and rare pairs of critical permissions (RPCP), to train SVM and inform users of the risks vs. benefits tradeoffs of apps.
- Peng et al. propose a score to measure the risk of apps, based on probabilistic generative models such as Naive Bayes.
- Yerima et al. also use features extracted from app permissions, API calls and commands extracted from the app executables.

III.PROPOSED SYSTEM:

We propose FairPlay, a system that leverages to efficiently detect Google Play fraud and malware. Our major contributions are:

To detect fraud and malware, we propose and generate relational, behavioral and linguistic features, that we use to train supervised learning algorithms

We formulate the notion of *co-review graphs* to model reviewing relations between users.

We develop PCF, an efficient algorithm to identify temporally constrained, co-review pseudo-cliques — formed by reviewers with substantially overlapping co-reviewing activities across short time windows.

We use temporal dimensions of review post times to identify suspicious review spikes received by apps; we show that to compensate for a negative review, for an app that has rating R, a fraudster needs to post at least positive reviews. We also identify apps with "unbalanced" review, rating and install counts, as well as apps with permission request ramps.

We use linguistic and behavioral information to (i) detect genuine reviews from which we then (ii) extract user-identified fraud and malware indicators.

IV.IMPLEMENTATION:

Implementation is the stage of the project when the theoretical design is turned out into a working system. Thus it can be considered to be the most critical stage in achieving a successful new system and in giving the user, confidence that the new system will work and be effective.

The implementation stage involves careful planning, investigation of the existing system and it's constraints on implementation, designing of methods to achieve changeover and evaluation of changeover methods.

MODULE DESCRIPTION:

1. FairPlay Overview
2. The Co-Review Graph (CoReG) Module
3. Reviewer Feedback (RF) Module.

1. FairPlay Overview.

FairPlay organizes the analysis of longitudinal app data into the following 4 modules, illustrated in Figure 1. The Co-Review Graph (CoReG) module identifies apps reviewed in a contiguous time window by groups of users with signifi- cantly overlapping review histories. The Review Feedback (RF) module exploits feedback left by genuine reviewers, while the Inter Review Relation (IRR) module leverages relations between reviews, ratings and install counts. The Jekyll-Hyde (JH) module monitors app permissions, with a focus on dangerous ones, to identify apps that convert from benign to malware. Each module produces several features that are used to train an app classifier. FairPlay also uses general

2. The Co-Review Graph (CoReG) Module.

Let the co-review graph of an app, see Figure 2, be a graph where nodes correspond to users who reviewed the app, and undirected edges have a weight that indicates the number of apps reviewed in common by the edge's endpoint users. We seek to identify cliques in the coreview graph. Figure 5a shows the co-review clique of one of the seed fraud apps (see § 3.2). To address the problem's NP-hardness, we exploit two observations. First, fraudsters hired to review an app are likely to post those reviews within relatively short time intervals (e.g., days). Second, perfect cliques are not necessary. Instead, we relax this requirement to identify "pseudo cliques", or groups of highly but not necessarily completely connected nodes. Specifically, we use the weighted density definition of Uno [23]: given a co-review graph, its weighted density $\rho = P\ e \in E\ w(e)\ (\ n\ 2)$, where E denotes the graph's edges and n its number of nodes (reviews). We are interested then in subgraphs of the co-review graph whose weighted density exceeds a threshold value θ.

3. Reviewer Feedback (RF) Module.

Reviews written by genuine users of malware and fraudulent apps may describe negative experiences. The RF module exploits this observation through a two step approach: (i) detect and filter out fraudulent reviews, then (ii) identify malware and fraud indicative feedback from the remaining reviews. Step RF.1: Fraudulent review filter. We posit that users that have higher expertise on apps they review, have written fewer reviews for apps developed by the same developer, have reviewed more paid apps, are more likely to be genuine. We exploit this conjecture to use supervised learning algorithms trained on the following features, defined for a review R written by user U for an app A: • Reviewer based features. The expertise of U for app A, defined as the number of reviews U wrote for apps that are "similar" to A, as listed by Google Play (see § 2). The bias of U towards A: the number of reviews written by U for other apps developed by A's developer. In addition, we extract the total money paid by U on apps it has reviewed, the number of apps that U has liked, and the number of Google+ followers of U

V.CONCLUSION:

In this paper, we introduced a novel two-factor data security protection mechanism for cloud storage system, in which a data sender is allowed to encrypt the data with knowledge of the identity of a receiver only, while the receiver is required to use both his/her secret key and a security device to gain access to the data. Our solution not only enhances the confidentiality of the data, but also offers the revocability of the device so that once the device is revoked, the corresponding ciphertext will be updated automatically by the cloud server without any notice of the data owner. Furthermore, we presented the security proof and efficiency analysis for our system.

VI.REFERENCES

[1] Google Play. https://play.google.com/.

[2] Ezra Siegel. Fake Reviews in Google Play and Apple App Store. Appentive, 2014.

[3] Zach Miners. Report: Malware-infected Android apps spike in the Google Play store. PCWorld, 2014.

[4] Stephanie Mlot. Top Android App a Scam, Pulled From Google Play. PCMag, 2014.

[5] Daniel Roberts. How to spot fake apps on the Google Play store. Fortune, 2015.

[6] Andy Greenberg. Malware Apps Spoof Android Market To Infect Phones. Forbes Security, 2014.

[7] Jon Oberheide and Charlie Miller. Dissecting the Android Bouncer. SummerCon2012, New York, 2012.

[8] VirusTotal - Free Online Virus, Malware and URL Scanner. https://www.virustotal.com/, Last accessed on May 2015.

[9] Iker Burguera, Urko Zurutuza, and Simin NadjmTehrani. Crowdroid: Behavior-Based Malware Detection System for Android. In Proceedings of ACM SPSM, pages 15–26. ACM, 2011.

[10] Asaf Shabtai, Uri Kanonov, Yuval Elovici, Chanan Glezer, and Yael Weiss. Andromaly: a Behavioral Malware Detection Framework for Android Devices. Intelligent Information Systems, 38(1):161–190, 2012