Finding Phising Rank in Google Play

M.Kishore
Student, Department Of Mca, Rcr Institutions Of Management & Technology, Tirupati, Andhra Pradesh, India

ABSTRACT

The commercial success of Android app markets such as Google Play and the incentive model they offer to popular apps, make them appealing targets for false and malicious behaviors. Some fraudulent developers deceptively boost the search rank and popularity of their apps (e.g., through fake reviews and bogus installation counts), while malicious developers use app markets as a launch pad for their malware. Proliferation. To identify malware, previous work has focused on app executable and permission analysis. In this paper, we introduce 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 behavioral signals gleaned from Google Play app data (87K apps, 2.9M reviews, and 2.4M reviewers, collected over half a year), in order to identify suspicious apps. FairPlay achieves over 95% accuracy in classifying gold standard datasets of malware, fraudulent and legitimate apps. We show that 75% of the identified malware apps engage in search rank fraud. FairPlay discovers hundreds of fraudulent apps that currently evade Google Bouncer's detection technology. FairPlay also 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.

I. INTRODUCTION

The commercial success of Android app markets such as Google Play and the incentive model they offer to popular apps, make them appealing targets for fraudulent and malicious behaviors. Some fraudulent developers deceptively boost the search rank and popularity of their apps (e.g., through fake reviews and bogus installation counts), while malicious developers use app markets as a launch pad for their malware. The motivation for such behaviors is impact: app popularity surges translate into financial benefits and expedited malware proliferation. 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 behavior “search rank fraud”. In addition, the efforts of Android markets to identify and remove malware are not always successful. For instance, Google Play uses the Bouncer system to remove malware. However, out of the 7,756 Google Play apps we analyzed using VirusTotal, 12% (948) were flagged by at least one anti-virus tool and 2% (150) were identified as malware by at least 10 tools. Previous mobile malware detection work has focused on dynamic analysis of app executables as well as static analysis of code and permissions. However, recent Android malware analysis revealed that malware evolves quickly to bypass anti-virus tools.

II. EXISTING SYSTEM

The efforts of Android markets to identify and remove malware are not always successful. For instance, Google Play uses the Bouncer system to
remove malware. Previous mobile malware detection work has focused on dynamic analysis of app executables as well as static analysis of code and permissions. However, recent Android malware analysis revealed that malware evolves quickly to bypass anti-virus tools.

Disadvantages:

✓ Can’t detect genuine reviews.
✓ Can’t identify fraud users and malware indicators.
✓ Time taking process with executing app and analysis of code permission methods.

III. PROPOSED SYSTEM

In this, we introduce 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 syntactical and behavioral signals gleaned from Google Play app data, in order to identify doubtful apps. FairPlay achieves over 95% accuracy in classifying gold standard datasets of malware, fraudulent and rightful apps. We show that 75% of the identified malware apps engage in search rank fraud. FairPlay discovers hundreds of fraudulent apps that currently evade Google Bouncer’s detection technology. FairPlay also helped the discovery of more than 1,000 reviews, reported for 193 apps that reveal a new type of forceful review operation. Malicious 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.

Advantages:

✓ Can detect genuine reviews
✓ Can identify fraud users and malware indicators.
✓ Identifies forceful reviews operation.

Algorithm:

Algorithm 1 PCF algorithm pseudo-code.

Input: days, an array of daily reviews, and \( \theta \), the weighted threshold density
Output: allClique, set of all detected pseudo-cliques

1. for \( d := 0 \) \( d < \text{days.size()} \); \( d++ \)
   2. Graph \( PC := \text{new Graph}() \);
   3. bestNearClique(\( PC, \text{days}[d] \));
   4. \( c := 1; n := PC.size(); \)
   5. for \( nd := d+1; d < \text{days.size()} \& c = 1; d++ \)
   6. bestNearClique(\( PC, \text{days}[nd] \));
   7. \( c := (PC.size() > n); \)
   8. if (\( PC.size() > 2 \))
     9. allClique := allClique.add(\( PC; fi \ endfor \)
   10. return
   11. function bestNearClique(\( \text{Graph PC, Set revs} \))
   12. if (\( PC.size() = 0 \))
     13. for root := 0; root < revs.size(); root++
     14. Graph \( \text{candClique} := \text{new Graph}() \);
     15. \( \text{candClique}.addNode(\text{revs[root]}, \text{getUser}()); \)
     16. do \( \text{candNode} := \text{getMaxDensityGain}(\text{revs}); \)
     17. if (\( \text{density(candClique} \cup \{\text{candNode}\}) \geq \theta \))
       \( \text{candClique}.addNode(\text{candNode}); \)
     18. \( \text{c} := (\text{PC.size()} > \theta); \)
     19. while (\( \text{candNode} != \text{null} \))
     20. if (\( \text{candClique}.density() > \text{maxRho} \))
       \( \text{maxRho} := \text{candClique}.density(); \)
     21. \( \text{PC} := \text{candClique}; \)
     22. if (\( \text{PC.size()} > 0 \))
       do \( \text{candNode} := \text{getMaxDensityGain}(\text{revs}); \)
     23. else if (\( \text{PC.size()} > 0 \))
     24. do \( \text{candNode} := \text{getMaxDensityGain}(\text{revs}); \)
     25. if (\( \text{density(candClique} \cup \{\text{candNode}\}) \geq \theta \))
       \( \text{PC}.addNode(\text{candNode}); \)
     26. \( \text{while (candNode} != \text{null} \));
     27. return

The Pseudo Clique Finder (PCF) algorithm:

We propose PCF (Pseudo Clique Finder), an algorithm that exploits the observation that fraudsters hired to review an app are likely to post those reviews within relatively short time intervals (e.g., days). PCF (see Algorithm 1), takes as input the set of the reviews of an app, organized by days, and a threshold value \( \theta \). PCF outputs a set of identified pseudo-cliques with \( p \geq \theta \), that were formed during contiguous time frames. For each day when the app has received a review (line 1), PCF finds the day’s most promising pseudo-clique (lines...
3 and 12 − 22): start with each review, and then greedily add other reviews to a candidate pseudo-clique; keep the pseudo clique (of the day) with the highest density. With that “working-progress” pseudo-clique, move on to the next day (line 5): greedily add other reviews while the weighted density of the new pseudo-clique equals or exceeds _ (lines 6 and 23 − 27). When no new nodes have been added to the work-in-progress pseudo-clique (line 8), we add the pseudo clique to the output (line 9), then move to the next day (line 1). The greedy choice (getMaxDensityGain, not depicted in Algorithm 1) picks the review not yet in the work-in-progress pseudo-clique, whose writer has written the most apps in common with reviewers already in the pseudo clique.

IV. CONCLUSION

We 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.

V. REFERENCES


