RESEARCH ARTICLE OPEN ACCESS

An Accurate Classifier for Detecting the Ecosystem of Malicious Application on Online Social Network using MADE.

Parivallal K, Praful Kumar .S, Santhosh Raj M

Mrs. G. Pushpa Antanet Sheeba, [Assistant Professor/CSE,GTEC]

1(Computer Science and Engineering, Ganadipathy Tulsi's Jain Engg. College, Kaniyambadi, Vellore- 632102.)
Email: parivallalk7@gmail.com)
2(Computer Science and Engineering, Ganadipathy Tulsi's Jain Engg. College, Kaniyambadi, Vellore- 632102.)
Email: praful62226@gmail.com)
3(Computer Science and Engineering, Ganadipathy Tulsi's Jain Engg. College, Kaniyambadi, Vellore- 632102.)
Email: santhoshr939@gmail.com)

Abstract:

Third-party apps are a major reason for the popularity and addictiveness of Online Social Network. However little is understood about the characteristics of malicious apps and how they operate. Malicious apps are much more likely to share names with other apps, and they typically request fewer permissions than benign apps. We will propose MADE – a Malicious Application Detector and Evaluator that will do two major things. Firstly, we will find the malicious apps often share names with other apps, and they typically request fewer permissions than benign apps. Secondly, leveraging these distinguishing features, we show that MADE can detect malicious apps with certain accuracy. We will also highlight the emergence of app-nets and will keep continue to dig deeper.

Keywords: Facebook apps, malicious, online social networks, spam.

I. INTRODUCTION

Online Social Networks (OSN) enable and encourage third party applications to enhance the user experience on this platform. These enhancements provide interesting and entertaining ways of communicating. Say, Facebook provides developers an API that facilitates app integration into the Facebook user experience. Hacker now have started taking advantages of the popularity of the third party application and malicious apps are providing lucrative business. Currently there are no tool available to advise user about the risks of an applications. As Facebook has dismantled their app rating facility it has become more difficult to know about the application. So here we propose MADE, a suite of efficient classifications techniques for identifying whether an app is malicious or not.

Moreover we will focus on the apps that collude with other application sharing our information and will try to prevent the

II. SOCIAL NETWORKS

Social Networks is an interdisciplinary and international quarterly. It provides a common forum for representatives of anthropology, sociology, history, social psychology, political science, human geography, biology, economics, communications science and other disciplines who share an interest in the study of the empirical structure of **social relations** and associations that may be expressed in **network** form. It publishes both theoretical and substantive papers. Critical reviews of major theoretical or methodological approaches using the notion of networks in the analysis of **social behavior** are also included, as are reviews of recent books dealing with **social networks** and **social structure**.

III. MALICIOUS APPS

The main motivation for detecting malicious apps stems from the suspicion that a significant fraction of malicious posts on OSNs are posted by apps. We

find malicious posts flagged by MyPageKeeper were posted by malicious apps.

Many of malicious apps get at least a hundred thousand clicks on the URLs they post. On quantifying the reach of malicious apps by determining a lower bound on the number of clicks on the links included in malicious posts. We identify all bit.ly URLs in posts made by that application. We focus on bit.ly URLs since bit.ly offers an API for querying the number of clicks received by every bit.ly link; thus, our estimate of the number of clicks received by every application is strictly a lower bound.

Across the posts made by the malicious apps in the D-Sample dataset, we found most were bit.ly URLs. We queried bit.ly for the click count of each URL. h. The application with the highest number of bit.ly clicks in this experiment—the "What is the truth about your past life?" app—receives more clicks then other general apps because of their marketing strategy. Although it would be interesting to find the bit.ly click-through rate per user and per post, we do not have data for the number of users who saw these links

.

IV. EMERGENCE OF APPNETS

Appearance of Appnets: Apps intrigue at large scale. We conduct a forensics investigation on the malicious app ecosystem to identify and quantify the techniques used to promote malicious apps. We find that apps collude and collaborate at a massive scale. Apps promote other apps via posts that point to the "promoted" apps. If we describe the collusion relationship of promoting–promoted apps as a graph, we find 1584 promoter apps that promote 3723 other apps. Furthermore, these apps form large and highly dense connected components, as shown in Fig. 1. Furthermore, hackers use fast-changing indirection: Applications posts have URLs that point to a Web site, and the Web site dynamically redirects too many different apps; we find 103 such URLs that point to 4676 different malicious apps over the course of a month. These observed behaviours indicate well-organized crime: One hacker controls many malicious apps, which we will call an app-net, since they seem a parallel concept to botnets.

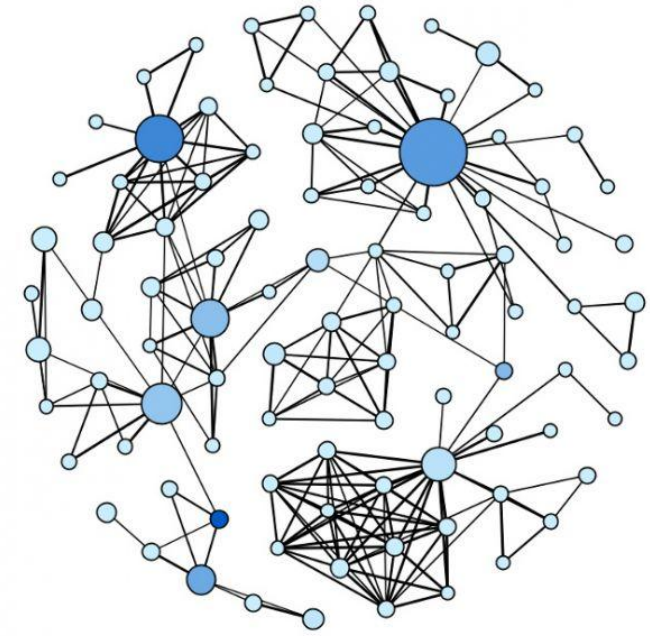

Fig. 1. Emergence and Colluding of App-nets.

Impersonation of Malicious Hackers. Some of the third party apps will impersonate themselves as trusted app and the post were made by them.

V. MY PAGEKEEPER

There are tons of malicious post and feeds circulated throughout the internet and used by the module. MyPageKeeper was used to crawl through the each and every single post made by the user and resulted in list of Benign and Malicious Apps. MyPageKeeper consists of six functional modules.

a. User authorization module. User's authorization is acquired to check the wall and feeds. Once a user installs the MyPageKeeper application, we obtain info that will help to access the needed information. For alerting the user, we also request permission to access the user's email address and to post on the user's wall and news feed.

b. Crawling module. As MyPageKeeper collects information on feeds, we will target the post containing URLs. Apart from the URL, It also contains information on user id, their ratings.

c. Feature extraction module. To classify a post, My-PageKeeper it will assess every single embedded URL. Our key lies in classifying post based on likes and comments. Moreover, we will check spread of the post.

d. Classification module. It uses techniques based on Support Vector Machines, but also utilizes several local and external whitelists and blacklists that help speed up the process and increase the overall accuracy. The classification module receives a URL and the related social context features extracted in the previous step if a URL is classified as fake posts, all posts containing the URL are labelled as such.

e. Notification module. The notification module notifies all users who have fake posts in their wall or news feed. we will make our system to remove the malicious post automatically, but this can create liabilities in the case of false positives.

f. User feedback module. Finally, to improve My-PageKeeper's ability to detect fake posts, we leverage our user community. Here they declare why they actually consider it to be fake.

VI. MALICIOUS APPLICATION DETECTION AND EVALUATOR

We develop MADE, a synthesized based suite that focuses on profiling, understanding and quantifying of parasitic ecosystem of apps. It will be totally based on attributes of the application that are being scanned using MADE. Here Feature Extractor will extract the data from Apps Data Base. The attributes are compared based on machine learning using support vector machines once they are classified malicious out of Blacklist. Benign apps comes under white list and Malicious under blacklist. With the help of blacklisted apps and their information we would be able to stop the emergence of App-nets.

ADVANTAGES

- Account Privacy and security will be increased by reducing the menace of hackers and securing the profiles.
- Breaking the cycle for app propagation.
- Enforcing stricter app authentication before posting.
- It will forbid cross promotion among application.

ARCHITECTURE

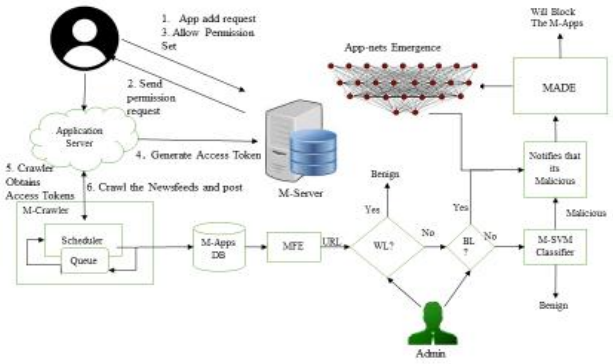

Fig. 2. Architecture for MADE

VII. PROFILING MALICIOUS AND BENIGN APPS

We will make sure that all the malicious and benign apps are classified based on aggregation based feature and on demand features using heuristics and Damerau–Levenshtein algorithm.

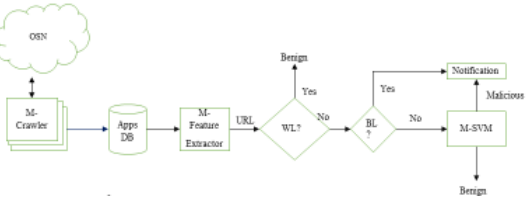

Fig. 3. Profiling Malign and Benign application

In fig. 4. We can see that whenever user login into any social network platform it will activate the web crawler that will actually go through the feeds and post made that are on our wall as well as on our recent feeds. Once crawling is done all the data will be sent to the application database where feature extractor will extract the links that are linked with the applications. Then the link will be compared with the white lists and black list to determine their usage as malicious or benign.

VIII. DAMERAU-LEVENSHTEIN ALGORITHM

The **Damerau–Levenshtein distance** (named after Frederick J. Damerau and Vladimir I. Levenshtein) is a distance (string metric) between two strings, i.e., finite sequence of symbols, given by counting the minimum number of operations needed to transform one string into the other, where an operation is defined as an insertion, deletion, or substitution of a single character, or a transposition of two **adjacent** characters. In his seminal paper, Damerau not only distinguished these four edit operations but also stated that they correspond to more than 80% of all human misspellings. Damerau paper considered only misspellings that could be corrected with at most one edit operation.

The Damerau–Levenshtein distance differs from the classical Levenshtein distance by including transpositions among its allowable operations. The classical Levenshtein distance only allows insertion, deletion, and substitution operations

Presently there are two algorithms first, simpler one, computes what is known as the *optimal string alignment distance* or *restricted edit distance and* second one computes the Damerau–Levenshtein distance with adjacent transpositions. Adding transpositions adds significant complexity.

The difference between the two algorithms consists in that the *optimal string alignment algorithm* computes the number of edit operations needed to make the strings equal under the condition that no substring is edited more than once, whereas the second one presents no such restriction.

Take for example the edit distance between CA and ABC. The Damerau–Levenshtein distance LD (CA, ABC) = 2 because CA → AC → ABC, but the optimal string alignment distance OSA (CA, ABC) = 3 because if the operation CA → AC is used, it is not possible to use AC → ABC because that would require the substring to be edited more than once, which is not allowed in OSA, and therefore the shortest sequence of operations is CA → A → AB → ABC. Note that for the optimal string alignment distance, the triangle inequality does not hold: OSA (CA, AC) + OSA (AC, ABC) < OSA (CA, ABC), and so it is not a true metric.

Applications: Spell checkers, correction systems for optical character recognition, Search programs.

		a		a	b	c	t
	0	1	2	3	4	5	6
a	1	0	1	2	3	4	5
	2	1	0	1	2	3	4
c	3	2	1	2	2	2	3
a	4	3	2	2	3	2	3
t	5	4	3	3	3	3	2

Fig.4. Damerau Levenshtein Distance Example

i. FRAUD DETECTION

The algorithm can be used with any set of words, like vendor names. Since entry is manual by nature there is a risk of entering a false vendor. A fraudster employee may enter one real vendor such as "Real Estates" versus a false vendor "Reel Estates". The fraudster would then create a false bank account and have the company route checks to the real vendor and false vendor. The Damerau–Levenshtein algorithm will detect the transposed and dropped letter and bring attention of the items to a fraud examiner.

ii. EVALUATING APPLICATION

Here using MADE, we will be able to prevent the malicious apps and link based on access token that is being generated to users id by either deactivating the app or activating it.

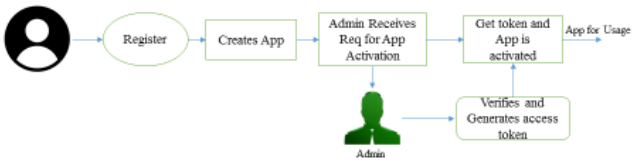

Fig. 5. Allowing Application for public usage.

In fig.6. We see that user after registering for the online social platform the user would be able to creating application for public usage. As app is created permission will be sent to admin for activating the app and on activation access token will be sent to the user registered ID which can actually be used to permit the application for public usage.

MADE can detect malicious apps with 99% accuracy. We develop MADE (Malicious Application Detector and Evaluator) to identify malicious apps using either using only features that can be obtained on-demand or using both on-demand and aggregation-based app information. It can identify malicious apps with 99.0% accuracy, with low false positives and high true positives. By adding aggregation-based information, MADE can detect malicious apps with accuracy, with no false positives and higher true positives.

The most important message of the work is that there seems to be a parasitic eco-system of malicious apps within Online Social network that needs to be understood and stopped. This could also help in other user platforms.

Breaking the cycle of app propagation. We recommend that apps should not be allowed to promote other apps. This is the reason that malicious apps seem to gain strength by self-propagation. Note that we only suggested against a special kind of app promotion where the user clicks the app A installation icon, app A redirects the user to the intermediate installation page of app B, and the user cannot see the difference unless she examines the landing URL very carefully where client ID is different. At the end, the user ends up installing app B although she intended to install app A. Moreover, cross promotion among apps is forbidden Say on Facebook's platform policy.

Enforcing stricter app authentication before posting. We recommend a stronger authentication of the identity of an app before a post by that app is accepted. As we saw, hackers fake the true identify of an app in order to evade detection and appear more credible to the end user.

IX. APPNETS AND HISTORY OF CROSS PROMOTION

A common way in which malicious apps collude is by having one app post links to the installation page of another malicious app. In this section, we conduct a forensics investigation on the malicious app ecosystem to identify and quantify the techniques used in this cross promotion of malicious apps.

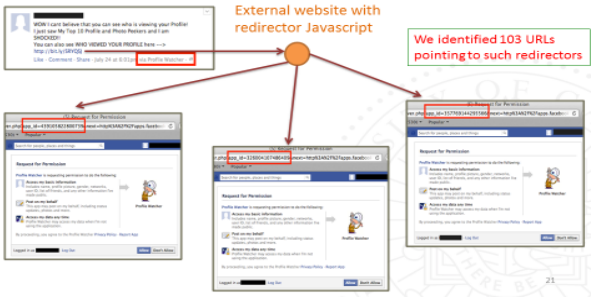

Fig. 6. Cross Promotion

When the malicious URL is clicked in the post, it redirects user to a JavaScript redirector controlled by the malicious hacker which randomly takes users to different malicious app installation pages

i. HISTORY OF CROSS PROMOTION

Cross promotion among apps, which is forbidden as per Facebook's platform policy, happens in two different ways. The promoting app has link to another app, or an redirection URL to set of certain apps.

Posting Direct Links to Other Apps: We found evidence that malicious apps often promote each other by making posts that redirect users to the promotee's app page; here, when app A posts a link pointing to app B, we refer to app A as the promoter an app N as the promotee. Promoter apps post on user who is basically tricked. These posts then appear in the news feed of the victim's friends. The post contains an appropriate message to lure users to install the promoted app, thereby enabling the promotee to accumulate more victims. We crawled the URLs posted by all malicious apps in our dataset and identified those where the landing URL

corresponds to an app installation page containing IDs.

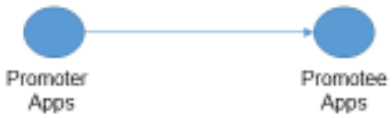

Fig. 7. Promoter posting links of Promotee Apps

Indirect App Promotion: Alternatively, hackers use Web sites outside actual OSNs to control them on higher level. In fact, the operation here is more sophisticated, and it obfuscates information at multiple places. Specifically, a post made by a malicious app includes a shortened URL, and that URL, once resolved, points to a Web site outside Facebook. This external Web site forwards users to several different app installation pages over time.

The use of the indirection mechanism is quite widespread, as it provides a layer of protection to the apps involved. In the course of MyPageKeeper operation, if we find any shortened URL points to an app installation URL (using an instrumented browser), we mark the URL as a potential indirection URL. Then, we crawl such potential indirection URL five times. If it redirects more than one landing URL, we mark it as an indirection URL.

ii. COLOBORATION OF APPLICATIONS

Here we will focus on groups of malicious apps that have collaborated with each other.

Two things are considered in here based

- *URL campaign*: Two apps if share a same URL they will listed here.
- *Domain campaign*: If two apps redirects to the same page then they are coming under this Campaign
- *Promoting URL campaign*: Apps promoted by same URL will be under this campaign.

iii. DOMAIN HOSTING

We investigate the hosting domains that enables redirection Web sites. First, our major focus will be on shortened URLs say *bit.ly*. Second, will try to track domains hosting malicious apps.

iv. PIGGYBACKING OF APPLICATION

From our set, we will discover about the impersonation of malicious applications. For e.g. to do so, they exploit weaknesses in Facebook's API. We call this phenomenon *app piggybacking*. One of the ways in which hackers achieve this is by luring users to "Share" a malicious post to get promised gifts. When the victim tries to share the malicious post, hackers invoke the Facebook API call http://www.facebook.com/connect/prompt_feed.php?api_key=POP_APPID, which results in the shared post being made on behalf of the popular app POP_APPID. The vulnerability here is that anyone can perform this API call, and Facebook does not authenticate that the post is indeed being made by the application whose ID is included in the request

For these apps, we manually examine the malicious posts flagged by MyPageKeeper.

Popular App	Malicious post by the app	Malicious link in the post
Farm Ville	WOW I just got 5000 Facebook Credits for Free	http://offers5000credit.blogspot.com
Facebook for iPhone	NFL Playoffs Are Coming! Show Your Team Support!	http://SportsJerseyFever.com/NFL
Mobile	WOW! I Just Got a Recharge of Rs 500.	http://ffreerechargeindia.blogspot.com/

Fig. 8. App Piggybacking

X. CONCLUSION

In this paper, using a large corpus of malicious apps observed over a 9-month period, we showed that malicious apps differ significantly from benign apps with respect to several features. For example, malicious apps are much more likely to share names with other apps, and they typically request fewer permissions than benign apps. Leveraging our observations, we developed MADE, an accurate classifier for detecting malicious applications. Most interestingly, we discussed about the emergence of app-nets—large groups of tightly connected applications that promote each other. We will continue to dig deeper into this ecosystem of

malicious apps on social networks say Facebook, and we hope that they will benefit from our recommendations for reducing the menace of hackers on their online platforms.

XI. FUTURE WORK

In proposed system the access token generated are sent to the E-mail ID provided by the user during registration. Online social network shall be able to enable features that will allow the user by themselves to prevent such malicious apps spread throughout the internet.

REFERENCES

[1] *Facebook, Palo Alto, CA, USA, "Facebook Opengraph API," [Online]. Available: http://developers.facebook.com/docs/reference/api/*

[2] *"Wiki: Facebook platform," 2014 [Online]. Available: http://en. wikipedia.org/wiki/Facebook_Platform*

[3] *"Pr0file stalker: Rogue Facebook application," 2012 [Online]. Available: G. Cluley, "The Pink Facebook rogue application and survey scam," 2012 [Online]. Available: http://nakedsecurity.sophos.com/2012/02/ 27/pink-facebook-survey-scam/*

[4] *D. Goldman, "Facebook tops 900 million users," 2012 [Online]. Available: http://money.cnn.com/2012/04/23/technology/facebookq1/index.htm*

[5] *R. Naraine, "Hackers selling $25 toolkit to create malicious Facebook apps," 2011 [Online]. Available: http://zd.net/g28HxI*

[6] *HackTrix, "Stay away from malicious Facebook apps," 2013 [Online]. Available: http://bit.ly/b6qWn5*

[7] *T.Stein,E.Chen,and K.Mangla,"Facebook immune system," inProc. 4th Workshop Social Netw. Syst., 2011, Art. no. 8.*

[8] *L. Parfeni, "Facebook softens its app spam controls, introduces better tools for developers," 2011 [Online]. Available: http://bit.ly/LLmZpM*

[9] *"Norton Safe Web," [Online]. Available: http://www.facebook.com/apps/application.php?id=310877173418*

[10] *"Damerau-Levenshtein Edit Distance Algorithm," [Online]. Available: https://en.wikipedia.org/wiki/Damerau%E2%80%93Levenshtein_distance*