FbDetector: Malicious App Tracing Using Neural Networks

JebastyAjitha.S[1], Dr.D.Sharmila[2],

[1]III MCA, Lord Jegannath College of Engineering & Technology

[2]Assistant Professor/HOD, Department of Computer Applications,

Lord Jegannath College of Engineering & Technology

Abstract: Facebook provides developers an API that facilitates apps integration into the Facebook user experience. Recently, hackers have started taking advantage of the popularity of this third-party apps platform and deploying malicious applications. Our key contribution is in developing FbDetector (Facebook Detector) arguably the first tool focused on detecting malicious apps on Facebook. To develop FbDetector, we use information gathered by observing the posting behavior of 111K Facebook seen across 2.2 million users on Facebook. First, we identify a set of features that help us differentiate malicious apps from benign ones. In FbDetector we are using neural network classifier for training and testing benign and malicious Facebook application.

I. INTRODUCTION

Online social networks enable and inspire third-party applications (apps) to improve the user knowledge on these platforms. Such improvements include interesting or entertaining ways of communicating among online friends and diverse actions such as playing games or listening to songs. For example, Facebook provides developers an API that facilitates app incorporation into the Facebook user experience. There are 500K apps available on Facebook, and on average, 20M apps are installed every day. Furthermore, many apps have acquired and maintain a really large user base. For instance, FarmVille and CityVille apps have 26.5M and 42.8M users to date. Recently, hackers have started taking benefit of the popularity of this third-party apps platform and deploying malicious applications. Malicious apps can give a gainful business for hackers, given the popularity of OSNs, with Facebook leading the way with 900M active users.

There are many ways that hackers can profit from a malicious app:1) The app can reach great numbers of users and their friends to spread spam; 2) The app can obtain users' personal information such as e-mail address, home town, and gender and 3) The app can "reproduce" by making other malicious apps popular. To make matters worse, the operation of malicious apps is simplified by ready-to-use toolkits starting at $25. In other words, there is motive and chance, and as a result, there are many malicious apps scattering on Facebook every day. Despite the above worrisome trends, today a user has very limited information at the time of installing an app on Facebook. In other words, the problem is the following: Given an app's identity number (the unique identifier assigned to the app by Facebook), can we detect if the app is malicious? Currently, there is no commercial service, publicly offered information, or research-based

tool to advise a user about the risks of an app. As we show in Section III, malicious apps are widespread and they easily spread, as an infected user jeopardizes the safety of all its friends.

II. RELATED WORKS

In mid-February 2008, there were approximately 866M installations of 16.7K distinct FB applications, and 200K developers are utilizing the platform. As of today, more than 100 OSN application development companies have been founded and FB application based advertising campaigns have been surprisingly successful [1]. Motivated by this unprecedented success, we became interested in studying the popularity (both its distribution and change over time), and adoption dynamics (how applications are installed by users) of FB applications. An in-depth understanding of these characteristics is important both for engineering and marketing reasons. Understanding the popularity and adoption dynamics can assist advertisers and investors to form strategies for acquiring applications or acquiring advertising real-estate, so as to reach a large portion of the targeted user-base at a low cost. At the same time, determining which applications tend to attract more users can help to better engineer the applications' user interface and features and to better provision the OSN system.

The previous work FRAppE uses the support vector machine (SVM) classifier for classifying malicious Apps.Thus SVM has the limitation of speed and size, both training and testing.

III. PROPOSED METHODOLOGY

FbDetector Lite is a lightweight version that makes use of only the application features available on demand. Give a specific app ID, FbDetector Lite crawls the on-demand features for that application and evaluates the application based on these features in real time. We envision that FbDetector Lite can be incorporated, for example, into a browser extension that can evaluate any Facebook application at the time when a user is considering installing it to her profile. Profile and description features used as inputs to FbDetector Lite and the source of each feature. All of these features can be collected on demand at the time of classification and do not require prior knowledge about the app being evaluated. We use the Neural Network (NN) classifier for classifying malicious apps. NN is widely used for binary classification in security and other disciplines.

Since we have applied NN, the model works well even for less statically training and NN have the ability to detect complex non-linear relationship between variables. So, the entire systems have the advantage of more precision and speed.

IV. MODULES

There are four modules are used as follows:

A. Benign and Malicious Dataset Representation:

The basis of our study is a dataset obtained from 2.2M Facebook users, who are monitored by My Page Keeper [5], our safety application for Facebook.1 My Page Keeper estimates every URL that it sees on any user's wall or news feed to determine if that URL points to social spam. My Page Keeper classifies a URL as social spam if it points to a Web page that: 1) spreads malware; 2) attempts to "phish" for individual information; 3) requests the user to carry out tasks (e.g., fill out surveys) that profit the owner of the Web site; 4) promises false rewards; or 5) attempts to entice the user to artificially inflate the reputation of the page (e.g., forcing the user to "Like" the page to access a false reward). My Page Keeper estimates each URL using machine-learning-

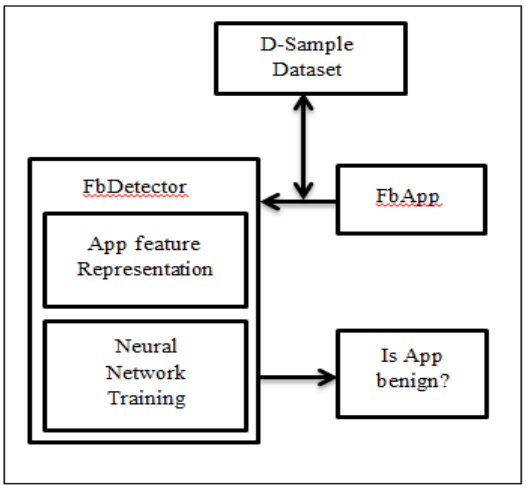

Fig 1. Architecture Diagram

based classifier that leverages the *social context* associated with the URL. For any particular URL, the features used by the classifier are attained by combining information from all posts (seen across users) containing that URL. Example features used by MyPageKeeper's classifier include the similarity of text message across posts and the number of comments/Likes on those posts. MyPageKeeper has false positive and false negative rates of 0.005% and 3%.

B. Facebook App Features Extraction:

Malicious apps typically have incomplete application summaries. First, we compare malicious and benign apps with respect to attributes present in the application's summary—*app description*, *company name*, and *category*. Description and company are free-text attributes, either of which can be at most 140 characters. On the other hand, category can be selected from a predefined (by Facebook) list such as "Games," "News," etc., that matches the app functionality best. Application developers can also specify the company name at the time of app creation. For example, the "Mafia Wars" app is configured with description as "Mafia Wars: Leave a legacy behind," company as "Zynga," and category as "Games" The fraction of malicious and benign apps in the D-Summary dataset for which these three fields are nonempty. We see that, while most benign apps specify such information, very rarely malicious apps do so. For example, only 1.4% of malicious apps have a nonempty description, whereas 93% of benign apps configure their summary with a description. We find that the benign apps that do not configure the explanation parameter are typically less popular (as seen from their monthly active users).

C. Neural Network Training:

Neural network design can best be give details with an example. The problem we will attack, recognize single letters in an image of text. This pattern recognition task has received much awareness. It is easy enough that many approaches achieve partial success, but hard enough that there are no perfect solutions. Many successful marketable products have been based on this difficulty, such as: reading the addresses on letters for postal routing, document entry into word processors, etc. The first step in increasing a neural network is to create a database of examples. For the text gratitude problem, this is accomplished by printing the 26 capital letters: A,B,C,D … Y,Z, 50 times on a sheet of paper. Next, these 1300 letters are renewed into a digital image by using one of the many scanning devices available for personal computers. This large digital image is then divided into small images of 10×10 pixels, each containing a single letter. This in sequence is stored as a 1.3 Megabyte database: 1300 images; 100 pixels per image; 8 bits per pixel. We will use the first 260 images in this database to *train* the neural network (i.e., determine the weights), and the remainder to *test* its presentation. The record must also contain a way of identifying the memo limited in each image. For instance, an additional byte could be added to each 10×10 image, containing the letter's ASCII code.

D. Malicious App Detection:

FbDetector Lite is a lightweight version that makes use of only the application features offered on demand. Given a specific app ID, FbDetector Lite crawl the on-demand features for that application and

evaluate the application based on these features in real time. We imagine that FbDetector Lite can be integrated,for example, into a browser addition that can evaluate any Facebook application at the time when a user is allowing for installing it to her profile.

We use the Neural Network (NN)[6] classifier for classifying malicious apps. NN is widely used for binary categorization in security and other disciplines [7], [8].We use the D-Complete dataset for training and testing the classifier. As shown previous in the D-Complete dataset consists of 487malicious apps and 2255 benign apps.

V. PERFORMANCE EVALUATION

The driving inspiration for detecting malicious apps stems from the misgiving that a significant fraction of malicious posts on Facebook are posted by apps. We find that 53% of malicious spots flagged by MyPageKeeper were posted by malicious apps. We further quantify the prevalence of malicious apps in two different ways.60% of malicious apps get at least a hundred thousand clicks on the URLs they post. We quantify the reach of malicious apps by influential a lower bound on the number of clicks on the links included in malicious posts. For each malicious app in our D-Sample dataset, we identify all bit.ly URLs in posts made by that application. We focus on bit.ly URLs since bit.ly offers an API [10] 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 6273 malicious apps in the D-Sample dataset, we found that 3805 of these apps had posted5700 bit.ly URLs in total. We queried bit.ly for the click count of each URL. Fig. 2 shows the distribution across malicious apps of the total number of clicks received by bit.ly links that they had posted. We see that 60% of malicious apps were able to accumulate over 100K clicks each, with 20%receiving more than 1M clicks each.

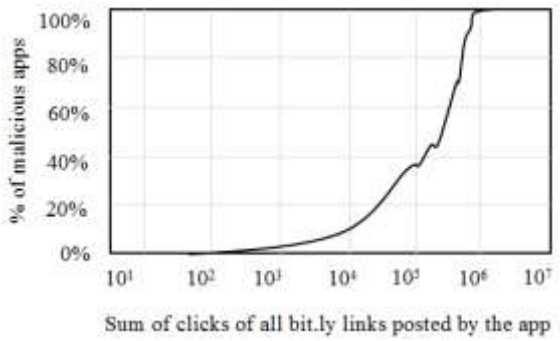

Fig.2: Clicks received by bit.ly links posted by malicious apps

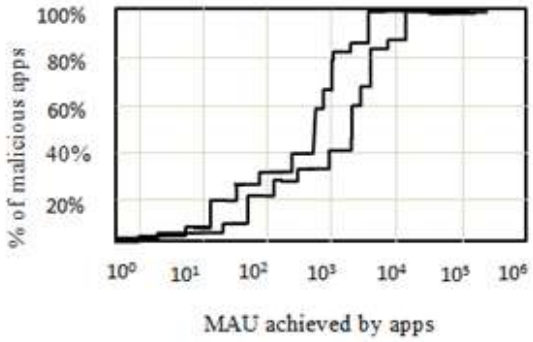

Fig.3: Median and maximum MAU achieved by malicious apps

The application with the highest number of bit.ly clicks in this experiment—the "What is the sexiest thing about you?" app—received 1 742 359clicks. 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. We can query bitly's API only for the number of clicks received by a link. 40% of malicious apps have a median of at least 1000 monthly active users. We examine the reach of malicious apps by inspecting the number of users that these applications had. To study this, we use the Monthly Active Users (MAU) metric provided by Facebook for every application. The number of Monthly Active Users is a measure of how many unique users are engaged with the application over the last 30 days in activities such as installing, posting, and liking the app Fig. 3plotsthe distribution of Monthly Active Users of the malicious apps in our D-Summary dataset. For each app, the median and maximum MAU values over the

three months are shown. We see that 40% of malicious applications had a median MAU of at least 1000 users, while 60% of malicious applications achieved at least 1000 during the 3-month observation period. The top malicious app here—"Future Teller"—had a maximum MAU of 260 000 and median of 20 000.

VI. CONCLUSION

In this paper, using a large corpus of malicious Facebook 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 a lot of more expected to share names with other apps, and they classically request less permission than gentle apps. Leveraging our clarifications, we improved FbDetector, an truthful classifier for detecting malicious Facebook applications. Most interestingly, we highlighted the emergence of app-nets—large groups of tightly linked applications that promote each other.

We have offered the first measurement-based characterization of the popularity and usage of third-party Facebook applications. We plan to extend this work with additional datasets, improved models, and study of more dynamic aspects such as application vitality on the social graph.

REFERENCE

1. [1] M. S. Rahman, T.-K.Huang, H. V. Madhyastha, and M. Faloutsos,"Efficient and scalable socware detection in online social networks,"in*Proc. USENIX Security*, 2012, p. 32.
2. H. Gao, Y. Chen, K. Lee, D. Palsetia, and A. Choudhary, "Towardsonline spam filtering in social networks," in *Proc. NDSS*, 2012.
3. P. Chia, Y. Yamamoto, and N. Asokan, "Is this app safe? A large scalestudy on application permissions and risk signals," in *Proc. WWW*, 2012, pp. 311–320.
4. "WhatApp? (beta)—A Stanford Center for Internet and SocietyWebsite with support from the Rose Foundation," [Online].Available:https://whatapp.org/facebook/
5. "MyPageKeeper," [Online]. Available: https://www.facebook.com/apps/application.php?id=167087893342260
6. C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *Trans. Intell. Syst. Technol.*, vol. 2, no. 3, 2011, Art.no. 27.
7. J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, "Beyond blacklists:Learning to detect malicious Web sites from suspicious URLs," in*Proc. KDD*, 2009, pp. 1245–1254.
8. A. Le, A. Markopoulou, and M. Faloutsos, "PhishDef: URL names sayit all," in *Proc. IEEE INFOCOM*, 2011, pp. 191–195.Business Week. Building a Brand with Widgets.http://www.businessweek.com/technology/content/feb2008/tc20080303_000743.htm.
9. "bit.ly API," 2012 [Online]. Available: http://code.google.com/p/bitlyapi/wiki/ApiDocumentation