International Journal of Research in Engineering, Science and Management
Volume-2, Issue-5, May-2019
www.ijresm.com | ISSN (Online): 2581-5792

Detecting Malicious Facebook Application

Sudhakar Avareddy[1], Meenakshi Devi V. Patil[2], J. Mallikarjun[3]
[1]*Assistant Professor, Department of Computer Science and Engineering, Ballari Institute of Technology and Management, Ballari, India*
[2,3]*Student, Department of Computer Science and Engineering, Ballari Institute of Technology and Management, Ballari, India*

***Abstract*: With 50 million installs a day, third-party apps are a major reason for the popularity and addictiveness of Facebook. We realized the potential of using apps for spreading malware and spamas we find that at least 30% of apps in our dataset are malicious. Our first path contribution is in developing FRAppE—Facebook's Rigorous Application Evaluator—arguably the first tool focused on detecting malicious apps on Facebook. we show that FRAppE can detect malicious apps FRAppE as a step toward creating an independent alarm watchman for app assessment and ranking, we explore the ecosystem of malicious Facebook apps and identify mechanisms that these apps use to propagate.**

***Keywords*: Facebook, detection, data collection, classification, malicious**

1. Introduction

Networking focuses on the building and verifying of online social networks for communities of people who share interests and activities, or who are interested in exploring the interests and activities of others, and which necessitates the use of software. It provides interaction between users who share interests, attitudes and activities, such as Facebook. Common interface, A possible benefit of social networks may be the common interface which spans work / social boundaries. Since such services are often used in a personal capacity the interface and the way the service works may be familiar, thus minimising training and support needed to exploit the services in a professional context. This can, however, also be a barrier to those who wish to have strict boundaries between work and social activities.

A. Literature survey

A technique for computer detection and correction of spelling errors [1]

The method described assumes that a word which cannot be found in a dictionary has at most one error, which might be a wrong, missing or extra letter or a single transposition. The unidentified input word is compared to the dictionary again, testing each time to see if the words match—assuming one of these errors occurred. During a test run on garbled text, correct identifications were made for over 95 percent of these error types.

A library for support vector machines [2]

LIBSVM is a library for Support Vector Machines (SVMs). We have been actively developing this package since the year 2000. The goal is to help users to easily apply SVM to their applications. LIBSVM has gained wide popularity in machine learning and many other areas. In this article, we present all implementation details of LIBSVM. Issues such as solving SVM optimization problems theoretical convergence multiclass classification probability estimates and parameter selection are discussed in detail.

Beyond blacklists: Learning to detect malicious Web sites from suspicious URLs [3]

Malicious Web sites are a cornerstone of Internet criminal activities. As a result, there has been broad interest in developing systems to prevent the end user from visiting such sites. In this paper, we describe an approach to this problem based on automated URL classification, using statistical methods to discover the tell-tale lexical and host-based properties of malicious Web site URLs. These methods are able to learn highly predictive models by extracting and automatically analysing tens of thousands of features potentially indicative of suspicious URLs. The resulting classifiers obtain 95-99% accuracy, detecting large numbers of malicious Web sites from their URLs, with only modest false positives.

Design and evaluation of a real-time URL spam filtering service [4]

On the heels of the widespread adoption of web services such as social networks and URL softener's, scams, phishing, and malware have become regular threats. Despite extensive research, email-based spam filtering techniques generally fall short for protecting other web services. To better address this need, we present Monarch, a real-time system that crawls URLs as they are submitted to web services and determines whether the URLs direct to spam. We evaluate the viability of Monarch and the fundamental challenges that arise due to the diversity of web service spam. We show that Monarch can provide accurate, real-time protection, but that the underlying characteristics of spam do not generalize across web services. In particular, we find that spam targeting email qualitatively differs in significant ways from spam campaigns targeting Twitter. We explore the distinctions between email and Twitter spam, including the abuse of public web hosting and redirector services. Finally, we demonstrate Monarch's scalability, showing our system could protect a service such as Twitter--which needs to process 15 million URLs/day--for a bit under $800/day.

Detecting spammers on social networks [5]

Social networking has become a popular way for users to meet and interact online. Users spend a significant amount of time on popular social network platforms (such as Facebook, Myspace, or Twitter), storing and sharing a wealth of personal information. This information, as well as the possibility of contacting thousands of users, also attracts the interest of cybercriminals. For example, cybercriminals might exploit the implicit trust relationships between users in order to lure victims to malicious websites. As another example, cybercriminals might find personal information valuable for identity theft or to drive targeted spam campaigns.

In this paper, we analyse to which extent spam has entered social networks. More precisely, we analyse how spammers who target social networking sites operate. To collect the data about spamming activity, we created a large and diverse set of "honey-profiles" on three large social networking sites, and logged the kind of contacts and messages that they received. We then analysed the collected data and identified anomalous behaviour of users who contacted our profiles. Based on the analysis of this behaviour, we developed techniques to detect spammers in social networks, and we aggregated their messages in large spam campaigns. Our results show that it is possible to automatically identify the accounts used by spammers, and our analysis was used for take-down efforts in a real-world social network. More precisely, during this study, we collaborated with Twitter and correctly detected and deleted 15,857 spam profiles.

2. Proposed system

We develop FRAppE, a suite of efficient classification techniques for identifying whether an app is malicious or not. To build FRAppE, we use data from My Page- Keeper, a security app in Facebook. We find that malicious applications significantly differ from benign applications with respect to two classes of features: On-Demand Features and Aggregation-Based Features.

A. System Model

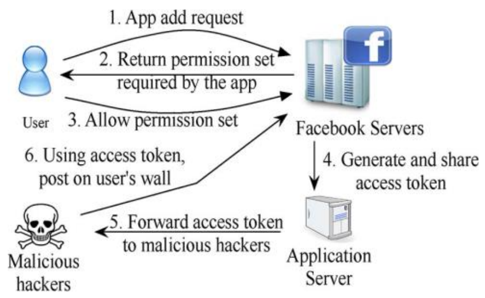

1. App add request
2. Return permission set required by the app
3. Allow permission set
User
Facebook Servers
4. Generate and share access token
5. Forward access token to malicious hackers
6. Using access token, post on user's wall
Malicious hackers
Application Server

Fig. 1. Representation of system model

3. Methodology

Data collection: The data collection component has two subcomponents: the collection of Facebook apps with URLs and crawling for URL redirections. Whenever this component obtains a Facebook app with a URL, it executes a crawling thread that follows all redirections of the URL and looks up the corresponding IP addresses. The crawling thread appends these retrieved URL and IP chains to the tweet information and pushes it into a queue. As we have seen, our crawler cannot reach malicious landing URLs when they use conditional redirections to evade crawlers. However, because our detection system does not rely on the features of landing URLs, it works independently of such crawler evasions.

Feature extraction: The feature extraction component has three subcomponents: grouping of identical domains, finding entry point URLs, and extracting feature vectors.

To classify a post, My PageKeeper evaluates every embedded URL in the post. Our key novelty lies in considering only the social context (e.g., the text message in the post, and the number of Likes on it) for the classification of the URL and the related post. Furthermore, we use the fact that we are observing more than one user, which can help us detect an epidemic spread.

It detects Presence of Spam keywords like ‘FREE’, ‘DEAL’ and ‘HURRY’.

Training: The training component has two subcomponents: retrieval of account statuses and training of the classifier. Because we use an offline supervised learning algorithm, the feature vectors for training are relatively older than feature vectors for classification. To label the training vectors, we use the account status; URLs from suspended accounts are considered malicious whereas URLs from active accounts are considered benign. We periodically update our classifier using labelled training vectors.

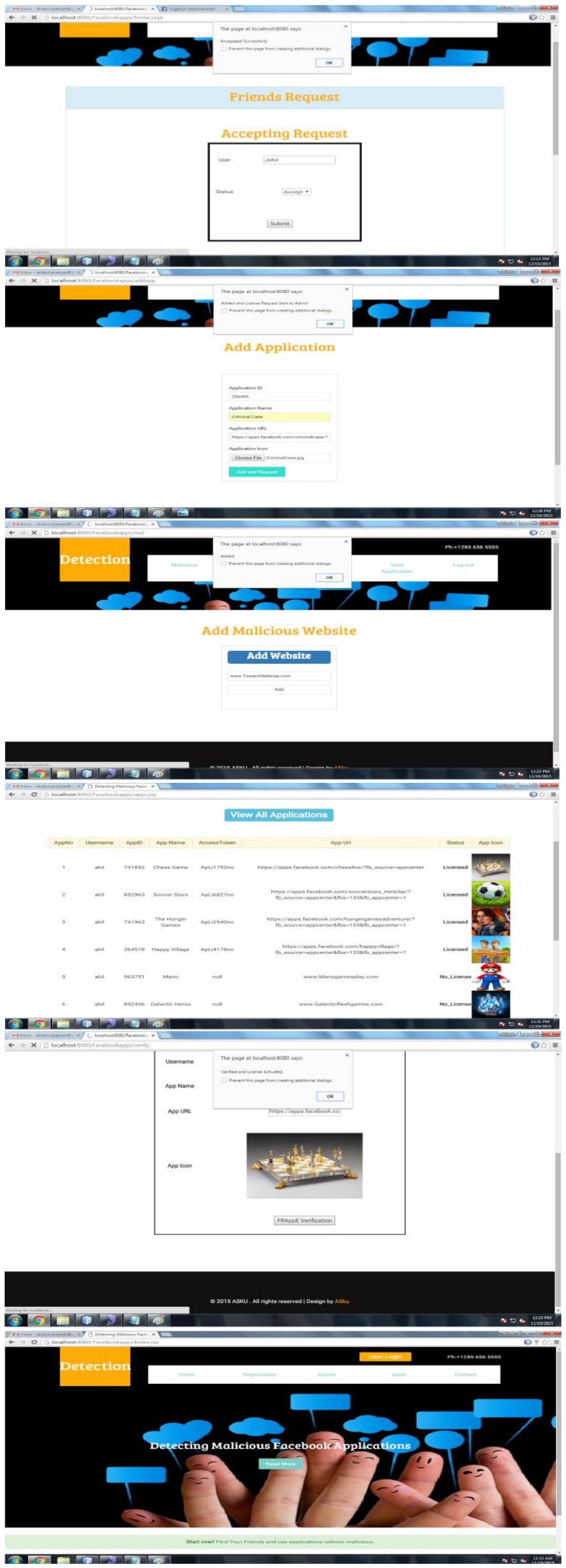
Classification: The classification component executes our classifier using input feature vectors to classify suspicious URLs. When the classifier returns a number of malicious feature vectors, this component flags the corresponding URLs information as suspicious.

The classification module uses a Machine Learning classifier based on Support Vector Machines, but also utilizes several local and external white lists and blacklists that help speed up the process and increase the over-all accuracy. The classification module receives a URL and the related social context features extracted in the previous step.

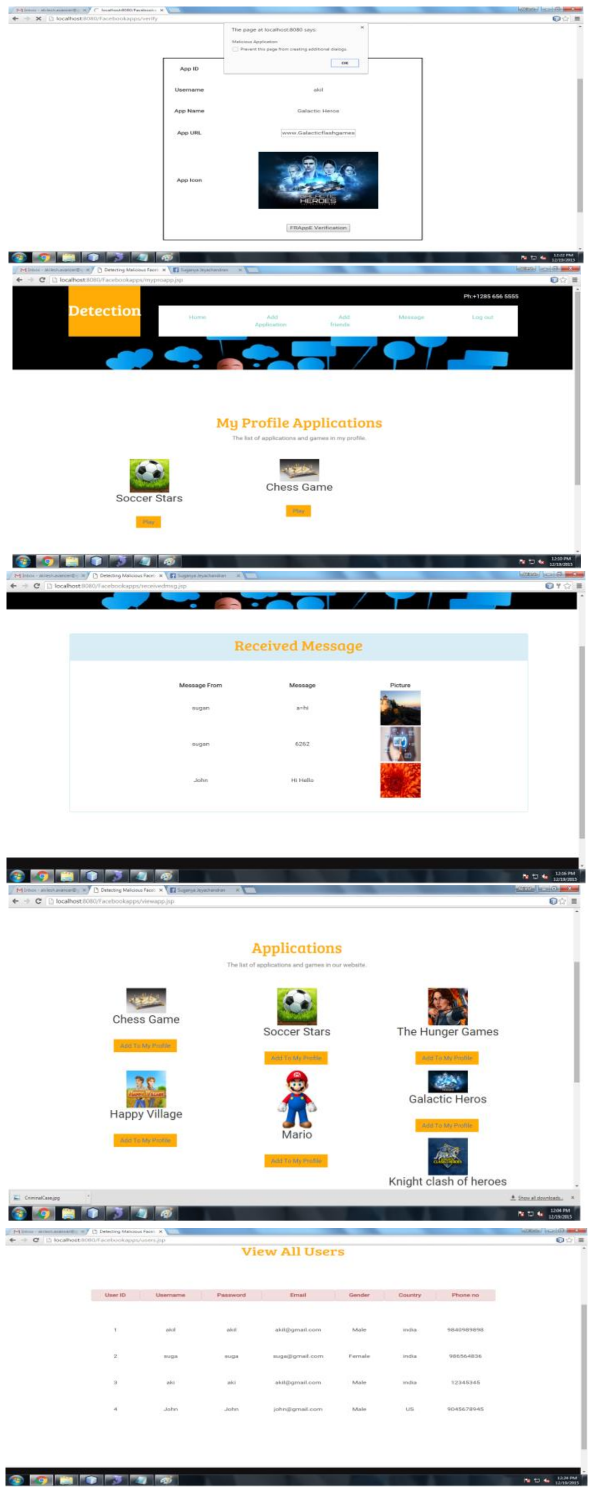
These URLs, detected as suspicious, will be delivered to security experts or more sophisticated dynamic analysis environments for an in-depth investigation.

Detecting Suspicious: The Detecting Suspicious and notification module notifies all users who have social malware posts in their wall or news feed. The user can currently specify the notification mechanism, which can be a combination of emailing the user or posting a comment on the suspect posts.

4. Results



AppNo	Username	AppID	App Name	AccessToken	App Url	Status
1	akil	741852	Chess Game	ApLi1752no	https://apps.facebook.com/chesslive/?fb_source=appcenter	Licensed
2	akil	852963	Soccer Stars	ApLi6827no	https://apps.facebook.com/soccerstars_miniclip/?fb_source=appcenter&fbs=133&fb_appcenter=1	Licensed
3	akil	741963	The Hunger Games	ApLi2940no	https://apps.facebook.com/hungergamesadventure/?fb_source=appcenter&fbs=133&fb_appcenter=1	Licensed
4	akil	264578	Happy Village	ApLi4178no	https://apps.facebook.com/happyvillage/?fb_source=appcenter&fbs=133&fb_appcenter=1	Licensed
5	akil	963751	Mario	null	www.Mariogamesplay.com	No_License
6	akil	852456	Galactic Heros	null	www.Galacticflashgames.com	No_License



Message From	Message
sugan	a=hi
sugan	6262
John	Hi Hello

User ID	Username	Password	Email	Gender	Country	Phone no
1	akil	akil	akil@gmail.com	Male	india	9840989898
2	suga	suga	suga@gmail.com	Female	india	986564836
3	aki	aki	akil@gmail.com	Male	india	12345345
4	John	John	john@gmail.com	Male	US	9045678945

Fig. 2. Results (Application)

5. Conclusion

Applications present convenient means for hackers to spread malicious content on Facebook. However, little is understood about the characteristics of malicious apps and how they operate. 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 much more likely to share names with other apps, and they typically request fewer permissions than benign apps. Leveraging our observations, we developed FRAppE, an accurate classifier for detecting malicious Facebook applications. Most interestingly, we highlighted 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 Facebook, and we hope that Facebook will benefit from our recommendations for reducing the menace of hackers on their platform.

References

[1] C. Pring, “100 social media statistics for 2012,” 2012. http://thesocialskinny.com/100-social-mediastatistics-for-2012/
[2] Facebook, Palo Alto, CA, USA, “Facebook Opengraph API,” http://developers.facebook.com/docs/reference/api/
[3] “Wiki: Facebook platform,” 2014. http://en. wikipedia.org/wiki/Facebook_Platform
[4] “Profile stalker: Rogue Facebook application,” 2012. https://apps.facebook.com/mypagekeeper/?status=scam_report_fb_survey_scam_pr0file_viewer_2012_4_4
[5] “Which cartoon character are you—Facebook survey scam,” 2012. https://apps.facebook.com/mypagekeeper/?status=scam_report_fb_survey_scam_whiich_cartoon_character_are_you_2012_03_30
[6] G. Cluley, “The Pink Facebook rogue application and survey scam,” 2012. http://nakedsecurity.sophos.com/2012/02/27/pink-facebook-survey-scam/