Detection of Malicious User in Facebook App Using Public Posts

Abhishek B A[1], Dr. Chandramouli H[2]
[1, 2] Department of CSE
[1, 2] East Point College of Engineering, Bangalore, Karnataka, India

Abstract- *Nowadays among Online Social Networks (OSN), Facebook is the widespread one and it is used by 1.5 billion people across the world. Hackers are finding many new ways to propagate spam and malware on these platforms, which we refer to as social malware. They can easily access the personal details. Social malware cannot be identified with existing security mechanisms (e.g. URL blacklists). Facebook app called My Page Keeper is used to protect Facebook users from social malware. The tool can detect malicious apps with complete accuracy. The objective of this project is to detect malicious application and block those applications in Facebook using the implemented tool under the set of constraints. Offensive words are detected and blocked using dictionary. There is already an overview given about just finding malicious app but not on blockage of offensive words or posts. It provides only a high-level overview about threats to the Facebook graph. The main disadvantage of existing system is security is missing. In proposed system certain techniques are implemented in finding the Offensive words or any posts, and dictionary detects the words. These words will not display in public wall. Instead of that such post will be automatically migrated to blocked post list. The user can view it secretly and also a warning mail is send to user. It is safe and secure. Unnecessary information will not be added in our wall. Thus the Offensive words and posts are blocked with the help of dictionary using filters and it is not publicly posted to user wall.*

Keywords- OSN (Online Social Network), JFC (Java Foundation Classes), JDT (Java Development Tools), EFS(Encrypting File System), IPSec (IP Security) GUI (Graphical User Interface),MVC (Model view Controller), DFD (Data Flow Diagram), JVM (Java Virtual Machine). FB (Facebook).

I. INTRODUCTION

Online social networks (OSN) enable and encourage third party applications (apps) to enhance the user experience on these platforms. Such enhancements will include interesting or entertaining ways of communicating among online friends, and diverse activities such as playing games or listening to songs. For example, Facebook provides developers an API [10] that facilitates app integration into the Facebook user-experience. There are 500K apps available on Facebook [25], and on average, 20M apps are installed everyday [1]. Furthermore, many apps have acquired and maintain a large user base. For instance, FarmVille and CityVille apps have 26.5M and 42.8M user's to date. Recently, hackers have started taking advantage of the popularity of this third-party apps platform and deploying malicious applications [17, 21, 24]. Malicious apps can provide a lucrative business for hackers, given the popularity of OSNs, with Facebook is leading way with 900M active users [12]. There are many ways that hackers can benefit from a malicious app. (a) the app can reach large numbers of users and their friends to spread spam. (b) The app can obtain users personal information such as email address, home town, and gender. (c) The app can "re-produce" by making other malicious apps popular. To make matters worse, the deployment of malicious apps is simplified by ready-to-use toolkits starting at $25 [13]. In other words, there is motive and opportunity and as a result, there are many malicious apps spreading on Facebook every day [20].

Despite the above there are some other trends, today, a user has very limited information at the time of installing an app on Facebook. In other words, the problem is given an apps 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-available information, or research-based tool to advise a user about the risks of an app. The malicious apps are widespread and they easily spread, as an infected user jeopardizes the safety to all of his friends. So far, the research community has paid little attention to OSN apps specifically. Most research related to spam and malware on Facebook has focused on detecting malicious posts and social spam campaigns [31, 32, 41]. A recent work studies how app permissions and community ratings correlate to privacy risks of Facebook apps [29]. Finally, there are some community-based feedbacks driven efforts to rank applications, such as Whatsapp [23], though these could be very powerful in the future, so far they have received little adoption.

II. SURVEY WORK

Detecting and Characterizing Social Spam Campaigns.

Authors: Hongyu Gao, Jun Hu, Christo Wilson,Zhichun Li, Yan Chen, Ben Y. Zhao.

Description: Authors presented a primary study to calculate and analyze spam campaigns launched on online social networks. They calculated a huge anonym zed dataset of asynchronous "wall" messages in between Facebook users. System detected generally 200,000 malicious wall posts with embedded URLs, originating from more than 57,000 user accounts. Authors found that more than 70% of all malicious wall posts advertise phishing sites. To study the distinctiveness of malicious accounts, and see that more than 97% are compromised accounts, rather than "fake" accounts formed solely for the principle of spamming. Finally, when adjusted to the local time of the sender, spamming dominates actual wall post in the early morning hours when users are feel good in public from the social spam activities.

Is this App Safe? A Large Scale Study on Application Permissions and Risk Signals.

Authors: Pern Hui Chia, Yusuke Yamamoto, N.Asokan.

Description: Third-party applications capture the attractiveness of web and platforms providing mobile application. Many of these platforms accept a decentralized control strategy, relying on explicit user consent for yielding permissions that the apps demand. Users have to rely principally on community ratings as the signals to classify the unsafe and inappropriate apps even though community ratings classically reflect opinions regarding supposed functionality or performance rather than concerning risks. To study the advantages of user-consent permission systems through a large data collection of Facebook apps, Chrome extensions and Android apps. The study confirms that the current forms of community ratings used in app markets today are not reliable for indicating privacy risks an app creates. It is found with some evidences, indicating attempts to mislead or entice users for granting permissions free applications and applications with mature content request "look alike" applications which have similar names as that of popular applications also request more permissions than is typical. Authors find that across all three platforms popular applications request more permission will have more than the average.

Social Applications: Exploring A More Secure Framework.

Authors: Andrew Besmer, Heather Richter Lipford,Mohamed Shehab, Gorrell Cheek

Description: OSNs such as Orkut, Facebook and others have grown-up rapidly, with hundreds to millions of active users. A new feature provided on several sites is social applications and services written by third party developers that supply additional functionality linked to a user's profile. However, present application platforms put users at risk by permitting the discovery of huge amounts of personal data and information to these applications and their developers. This paper generally abstracts main view and defines the current access control model gave to these applications, and builds on it to generate a more secure framework.

III. EXISTING SYSTEM

Analysis is the process of finding the best solution to the problem. System analysis is the process by which we learn about the existing problems, define objects and requirements and evaluates the solutions. It is the way of thinking about the organization and the problem it involves, a set of technologies that helps in solving these problems. Feasibility study plays an important role in system analysis which gives the target for design and development.

Feasibility study depending on the results of the initial investigation the survey is now expanded to a more detailed feasibility study. "Feasibility study" is a test of system proposal according to its workability, impact of the organization, ability to meet needs and effective use of the resources. Economic feasibility this study is carried out to check the economic impact that the system will have on the organization. the amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased. Technical feasibility is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

Social feasibility is aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His

level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

IV. PROPOSED SYSTEM

Design is a creative process; a good design is the key to effective system. The system "Design" is defined as "The process of applying various techniques and principles for the purpose of defining a process or a system in sufficient detail to permit its physical realization". Various design features are followed to develop the system. The design specification describes the features of the system, the components or elements of the system and their appearance to end-users.

A set of fundamental design concepts has evolved over the past three decades. Although the degree of interest in each concept has varied over the years, each has stood the test of time. Each provides the software designer with a foundation from which more sophisticated design methods can be applied. The fundamental design concepts provide the necessary framework for "getting it right". The fundamental design concepts such as abstraction, refinement, modularity, software architecture, control hierarchy, structural partitioning, data structure, software procedure and information hiding are applied in this project to getting it right as per the specification.

The input Design is the process of converting the user-oriented inputs in to the computer-based format. The goal of designing input data is to make the automation as easy and free from errors as possible. Providing a good input design for the application easy data input and selection features are adopted. The input design requirements such as user friendliness, consistent format and interactive dialogue for giving the right message and help for the user at right time are also considered for the development of the project. Input design is a part of overall system design which requires very careful attention. Often the collection of input data is the most expensive part of the system, which needs to be route through number of modules .It is the point where the user ready to send the data to the destination machine along with known IP address; if the IP address is unknown then it may prone to error. A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other systems through outputs. It is most important and direct source information to the user. Efficient and intelligent output improves the systems relationship with source and destination machine. Outputs from computers are required primarily to get same packet that the user has send instead of corrupted packet and spoofed packets. They are also used to provide to permanent copy of these results for later consultation.

Swing actually makes use of a simplified variant of the MVC design called the model-delegate. This design combines the view and the controller object into a single element that draws the component to the screen and handles GUI events known as the UI delegate. Communication between the model and the UI delegate becomes a two-way street. Each Swing component contains a model and a UI delegate. The model is responsible for maintaining information about the component's state. The UI delegate is responsible for maintaining information about how to draw the component on the screen. The UI delegate (in conjunction with AWT) reacts to various events that propagate through the component.

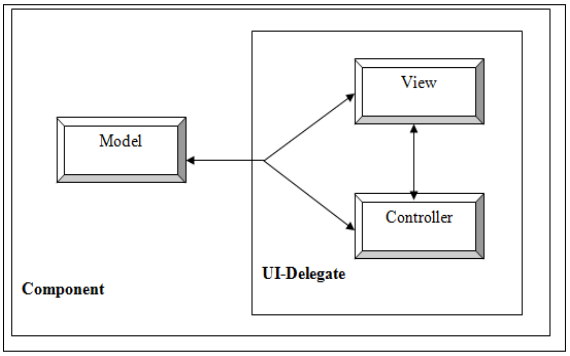

Figure 1.

The design method that has been followed to design the architecture of the system is MVC design pattern. Swing uses the model-view-controller (MVC) architecture as the fundamental design behind each of its components. Essentially, MVC breaks GUI component into three elements. Each of these elements plays a crucial role in how the component behaves. The MVC design pattern separates a software component into three distinct pieces: a model, a view, and a controller. The model is the piece that represents the state and low-level behavior of the component. It manages the state and conducts all transformations on that state. The model has no specific knowledge of either its controllers or its views. It encompasses the state data for each component. There are different models for different types of components. For example, the model of a scrollbar component might contain information about its current position of its adjustable "thumb", its minimum and maximum values, and the thumb's width. A menu on the other hand, may simply contain a list of the menu items the user can select from. The system itself maintains links between model and views and notifies the views when the model changes state. The view refers to how you see the component in the screen. It is the piece that manages the visual display of the state represented by the

model. Almost all window frames will have a title bar spanning the top of the window. However the title bar may have a close box on the left side or on the right side. These are the examples of different types of views for the same window object.

A model can have more than one view, but that is typically not the case in the Swing set. System architecture is the conceptual design that defines the structure and behavior of a system. An architecture description is a formal description of a system, organized in a way that supports reasoning about the structural properties of the system. It defines the system components or building blocks and provides a plan from which products can be procured, and systems developed, that will work together to implement based on Facebook in the overall system.

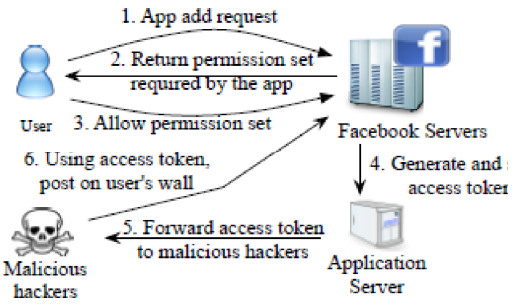

Figure 2.

We systematically profile apps and show that malicious app profiles are significantly different than those of benign apps. A striking observation is the “laziness" of hackers; many malicious apps have the same name, as 8% of unique names of malicious apps are each used by more than 10 different apps (as defined by their app IDs). Overall, we profile apps based on two classes of features: (a) those that can be obtained on-demand given an application’s identifier (e.g., the permissions required by the app and the posts in the application’s profile page), and (b) others that require a cross-user view to aggregate information across time and across apps (e.g., the posting behavior of the app and the similarity of its name to other apps).

V. DATA FLOW DIAGRAM

A data-flow diagram (DFD) is a graphical representation of the "flow" of data through an information system. DFDs can also be used for the visualization of data processing (structured design). On a DFD, data items flow from an external data source or an internal data store to an internal data store or an external data sink, via an internal process.

A context-level or level 0 data flow diagram shows the interaction between the system and external agents which act as data sources and data sinks. On the context diagram (also known as the Level 0 DFD) the system's interactions with the outside world are modeled purely in terms of data flows across the system boundary. The context diagram shows the entire system as a single process, and gives no clues as to its internal organization.

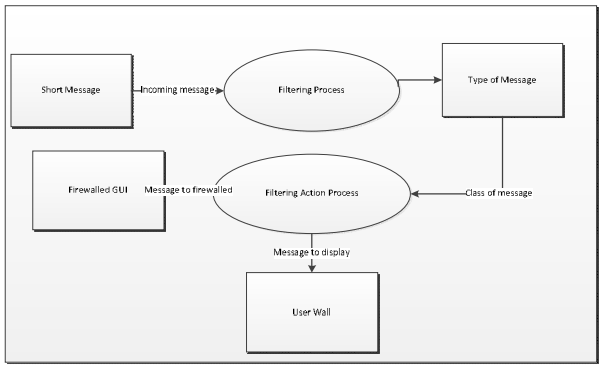

Figure 3.

The Level 1 DFD shows how the system is divided into sub-systems (processes), each of which deals with one or more of the data flows to or from an external agent, and which together provide all of the functionality of the system as a whole. It also identifies internal data stores that must be present in order for the system to do its job, and shows the flow of data between the various parts of the system.

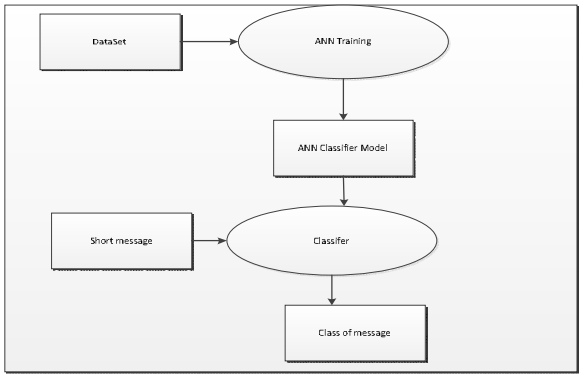

Figure 4.

VI. CONCLUSION

Applications present a 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 work, using a large corpus of malicious Facebook apps observed, over a nine 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 less permission than benign apps. Leveraging our observations, we developed Facebook App, an accurate classifier for detecting malicious Facebook applications. Most interestingly, we highlighted the emergence of AppNets— 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 on there are hackers on their platform.

Future Scope: In the Future, we can go with the other than text posts like image post, video post, we can classify based on the image features and video features so that we can easily detect the FB as a malicious if the user posts the unwanted images or videos.

REFERENCES

[1] C. Pring, "100 social media statistics for 2012," 2012 [Online].

[2] Facebook,PaloAlto,CA,USA, "Facebook Opengraph API," [Online].

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

[4] "Profile stalker: Rogue Facebook application," 2012 [Online].

[5] "Which cartoon character are you—Facebook survey scam," 2012 [Online].

[6] G. Cluley, "The Pink Facebook rogue application and survey scam," 2012 [Online].

[7] D. Goldman, "Facebook tops 900 million users," 2012 [Online].

[8] HackTrix, "Stay away from malicious Facebook apps," 2013 [Online].

[9] 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.

[10] H. Gao et al., "Detecting and characterizing social spam campaigns," in Proc. IMC, 2010, pp. 35–47.

[11] H. Gao, Y. Chen, K. Lee, D. Palsetia, and A. Choudhary, "Towards online spam filtering in social networks," in Proc. NDSS, 2012.

[12] "WhatApp? (beta)—A Stanford Center for Internet and Society Website with support from the Rose Foundation," [Online].