An App for Phishing Attack Detection in Smartphone

Dinesh.S[1], Saranya K[2]
[1]U.G student, Department of Computer Science and Engineering, Sri Ramakrishna Institute of Technology
[2] Assistant Professor, Department of Computer Science and Engineering, Sri Ramakrishna Institute of Technology
Coimbatore-641010
Email: dineshsaratha8@gmail.com, saranyak249@gmail.com

***Abstract*—Phishing attack is used to steal the username and password from the targeted user. This project proposes a mobile application prototype to detect phishing attacks using an android app called "GoUnPhish". This prototype uses Deceptive by Deceptive approach to detect the phishing attacks. This application uses very low computation power.**

Keywords—phishing; deceptive; hashcode;

I. INTRODUCTION

Phishing Attack violates the security of many authentication procedures. This attack can be identified by user who knows the knowledge of phishing attack. Phishing attack involves cloning of the original website and use that cloned website against targeted user. This cloned website is hosted in the hacker server, and then the hacker forces the targeted user to click on the fake website. When the user clicks that fake website the user will be redirected to the cloned website that is created by the hacker. In this stage, user is going to enter the username and password in the fake website. When the user submits the username and password, that website generates some error and the user will be redirected to the original website. The hacker collects the login data from the fake website he/she created. In this way Phishing attack works in the Internet. This attack can be performed on Computer and also on Mobile Phones. Mostly phishing attacks are identified in desktop browser using its security features but when coming to mobile, this attack is identified in less number because of less security in the mobile browser in order to increase performance of the browser.

Phishing attacks are increasing in recent techno world. Attackers use many innovative engineering techniques to make them believe with the malware or deceptive login-based web pages. Most solutions for this problem concentrate more on desktop computers rather than mobile devices. Mobile users are more vulnerable than the desktop users and they have device limitations such as smaller screen size and low computational power. GoUnPhish enables a mobile device user to create fake login account, with fake login credentials, and automates login procedure every time when the user tries to open a login webpage and generates an alert message, depicating whether the website is malicious or not.

GoUnPhish determines whether the current login page shifts to another webpage after authentication. When the page is loading, it listens to the response code and then makes a decision on whether the website is fraudulent or not. The effectiveness of this mobile app can be measured by conducting user experiment on android platforms and testing its detection accuracy, memory and CPU performance. GoUnPhish uses a very small amount of computational power and it is effective in assisting users to identify the Phishing attacks.

II. RELATED WORK

M.Archana et al, proposed an anti-phishing tool to detect Phishing websites to save the users from fraudulent sites [1]. Behavior of sites can be found by long URL‟s and by seeing its response to the user. This tool includes web page extraction for analysis such as Text Extraction, Color Extraction and Image Extraction.

Sharvari Prakash Chorghe et al, proposes two enforcement mechanisms for protection in system levels and inter component communication level [2]. The detection techniques are divided into static and heuristic based detection based on blacklisted approach. This paper discusses about the phishing attack detection techniques like Mobi-Phish and MP-Shield and QR code based phishing detection techniques.

These are the detection prototype on mobile web browsers. Mobi-phish method uses Optical character recognition tool for detection..

Adrienne Porter Felt et al proposes about identities of mobile application and websites which describes screen control and browser window in mobiles [3]. They conducted a systematic analysis of ways in which mobile applications and web sites link to each other and discovered that web sites and applications regularly ask users to type their passwords into contexts that are vulnerable to spoofing.

CikFeresaMohd Foozy et al, proposed a taxonomy on Phishing Attack and its detection to help future work on Mobile devices [4]. They explained about detections techniques such as Content Based, Blacklist, Whitelist, Hotspot, Gaussian Mixture Model and Graylist Attack strategies for phishing in Bluetooth, SMS, Vishing, Mobile web application.

Victor Clincy et al, published a journal on Information Security which says about techniques of phishing attack such as Small screen and partial display of URLs, Accessibility to app store, Smishing, Wi-Fi and Vishing and just creates an awareness for emerging phishing attacks [5]. In mobile phones, the screen size is small and the chance to see the full URL is less. Through Android Application Market, Phishing Applications are uploaded and Phishing attack is done. Another method of phishing is sending fake login links through SMS which is called Smishing. And major way is through connecting Mobile phones to Wi-Fi Hotspot where attackers set up a Wi-Fi to eavesdrop on wireless communications.

G.Bottazzi et al, proposed a modular framework that (a) can be easily customized by users, (b) is adaptable in term of detection strategy and algorithms adopted, (c) is reusable over all compatible platforms, (d) is scalable compared to the rate and type of connections [6]. This natively intercepts any outgoing and incoming IP traffic of the device, originated by either an HTTP browser, such as Firefox or Chrome, or any other app requiring an Internet connection. HTTP-get requests are intercepted from filtered IP packets with a twofold aim: on the one hand a public blacklist is queried for well-known phishing URLs, while on the other, a set of number of sub-domains in the URL, length of the URL, fraction of digits in a URL versus the length of the URL are extracted by the proxy and exploited to assure zero hour protection from new phishing campaigns. Also it exploits techniques such as public blacklist search which implements a machine language based engine to ensure zero hour protection from new phishing campaigns.

LongfeiWy et al. presents an automated anti-phishing scheme which verifies the validity of web pages and apps by comparing the actual identity to identity claimed by the web pages and apps [7]. The mobile application has two independent components called WebFish and AppFish where Web Fish does the web modules in detection and AppFish performs registration and alert functions. Motivation, Identity Extraction and OCR are the schemes of Mobi-Fish.Compared to existing OCR based anti-phishing schemes (designed for PC only), Mobi-fish is lightweight as it works without using external search engines or machine learning algorithms.

Luka Malisa et al, proposed a novel spoofing detection approach (8) tailored to the protection of mobile app login screens, using extraction and visual similarity comparison. This explains about the login screen spoofing of Phishing attack. Approaches handled in this paper are Visual handling and Deception rate analysis. Screenshot analysis is done for the detection of the attack.

III. METHODOLOGY

As discussed earlier, mostly phishing attacks are identified only in desktop browser not in the mobile browser so this project proposes a Mobile Application Prototype to identify phishing attack in the mobile browser. This prototype uses Deceptive by Deceptive approach (i.e.) deceiving a system of the attacker with an authentication attempt using fake login credential. By this method user can able to identify the phishing attack in the mobile browser. This method mainly uses two important things one is Hashcode and another one is

Response code. Hashcode refers to the URL we are browsing and Response code refers to code generated when we submit a form in the website. In our case, the form refers to Login form in the login page of the website. Hashcode is used to identify the page shift authentication in the browser. GoUnPhish app intercepts the user when he/she opens the login page in the browser.

When the user opens the login page in the browser, GoUnPhish app notifies the user that whether a website is legitimate or malicious.

As per the RFC 2616 industrial Standard response for client request, if the response code starts with 4 it refers to Client Error. If the response code is 401, it refers that the user is unauthorized to access that website. Suppose if the response code is 403, it refers that the request gets forbidden.

In the Existing system, there is no app to detect phishing attacks in the mobile browser and our proposed system uses completely different approach to detect phishing attack in mobile browser without affecting the performance of the browser. Proposed system uses Deceptive by Deceptive approach. The hacker uses fake page to get the login credential from the targeted user this is called Deceptive Approach. This project uses the same approach to detect the Phishing attack i.e. this app provide fake credential to the hacker so that app can able to identify the phishing attack and also protect our account from the hacker. This is also a Deceptive approach so this method is called as Deceptive by Deceptive method. Here Hacker uses fake page and the user also uses fake credential.

Also Page Shift Authentication is used in the proposed system to identify the malicious site. In Page Shift Authentication URL is tracked to identify whether it is legitimate or malicious. If the URL is redirected to another page while authenticating the page using fake credential, that page is said to be malicious but some legitimate site also redirects to another page (another URL).This problem will be overcome by analysing check frequency of the website. Maximum a legitimate site will have seven login pages so maximum, a login page is redirected to seven other login pages. If the URL is redirected more than seven times that page is also said to be malicious site.

IV. System Specification

4.1 Hardware Requirements

The hardware requirements for the system are as follows

Processor	:	Intel Core i5
RAM	:	8 GB
Hard Disk	:	1 TB
Processor Core	:	Quad Core
Processor Speed	:	1.5 GHz

4.2 Software Requirements

The software requirements for the system are as follows

Language	:	Java, JavaScript
Platform	:	Android
Developing Tool	:	Android Studio
Server	:	Xampp Server
Database	:	SQLite Database

V. Architecture

5.1 System Architecture

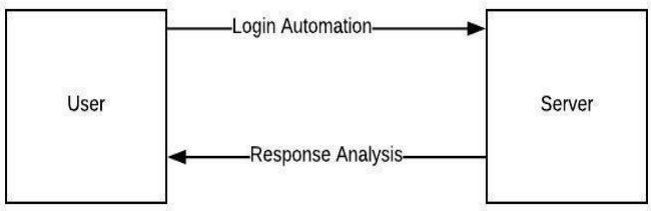

Fig 5.1 System Architecture

The above figure shows System Architecture Design for this mobile application prototype. This architecture consists of two processes one is Login Automation and another one is Response analysis

Login Automation involves automating the procedure of login using fake credential i.e. fake credential is submitted against the login page automatically by the App.

Response Analysis involves analysing the response code generated while submitting the fake login credential. This process is done by the app while opening the login page in the browser.

VI. IMPLEMENTATION

6.1 Module Description

This project is divided into four modules

1. Login UI
2. WebView
3. Login Automation
4. Response Analysis

6.1.1 Login UI

In this module fake credential (fake username and fake password) is stored in UnPhishMe App to identify whether the site is legitimate or not. SQLite database is used to store the data in the app.

SQLiteDatabase class is used to create the database or open an already existing database.

Syntax to create a database is

SQLiteDatabase mydatabase = openOrCreateDatabase ("your database name", MODE_PRIVATE, null);

SQLiteOpenHelper class is also used to create the database and also to manage the database. To use this class, SQLiteOpenHelper class is inherited. Three major methods used in this class are onCreate() , onUpgrade(), onOpen(). These methods can be override to implement the code on creating database, on upgrading database and also on opening database.

To store data in the database we have to execute the query in the database. To achieve this, exec() method of SQLiteDatabase is used.

Syntax to execute the query is

Mydatabase.exec("Query to be executed");

6.1.2 WebView

Webview module is used to load the website from the server in UnPhishMe App. WebView class is used to create the webview in the UnPhishMe app. This class consist of all methods that are used to interact with the website.

Syntax to create a webview is

WebView wv=(WebView)findViewById(R.id.wv);

URL(Website) is loaded in the webview using loadUrl() method of the webview class.

Syntax to load URL in webview

is webview.loadUrl("Your URL");

html data is loaded in the webview using loadData() method of the webview class.

Syntax to load Data in webview

is webview.loadData("html string") ;

6.1.3 Login Automation

In this module Login Procedure is automated. This project tries to login automatically in the website with the fake credential (fake username and fake password).To do this task, JavaScript is used. JavaScript code is embedded in the webview using loadUrl() method in the webview class. To do login automation the id for the username and password in the html page should be identified. Then the id of the submit button should be identified.

The following JavaScript code is used to do login automation

javascript:(function(){document.getElementById('m_login_email').value="'+email+"';document.getElementById('m_login_password').value="'+password+"';document.getElementById('u_0_5').click();})()

The above JavaScript code is stored as a string then this code is loaded in the webview using loadUrl() method.

Syntax to run JavaScript in the webview is

webview.loadUrl("Your JavaScript code");

6.1.4 Response Analysis

In this module the response code is checked depending upon the page shift authentication and check frequency value.

Many types of response code are generated for a website depending upon the way the website responses. For example: if the response code is 200 it refers that the request is succeeded suppose if the response code is 202 it refers that the request is accepted for processing.

In this project, majorly three response code is used to identify whether the website is legitimate or malicious.

1)200-Ok

2)401-Unauthorized

3)403-Forbidden

According to RFC 2616 Industrial standard following are the description of the above response code.

6.1.4.1 200 OK

This response shows that the request is succeeded [11]. Depending upon the request type the response is generated. There are two type of request one is GET request and another one is POST request.

401 Unauthorized

This response shows that the user is not authorised to access the resources of the webpage. If the username or password mismatches this response is generated. In this project 401 response code is majorly used to identify whether the website is legitimate or malicious.

403 Forbidden

This response shows that the request is forbidden due to wrong format of header that is used to send a request to the server. This also shows that the user is not authorised to access the resources of the webpage

VII. RESULTS

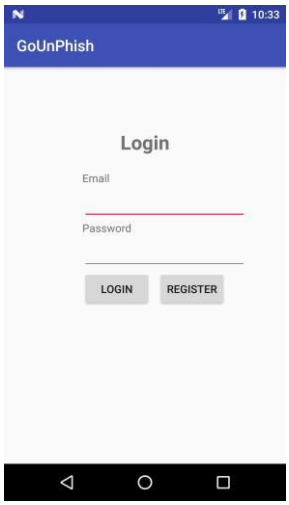

Fig 7.1 Login UI

Figure 5 shows the login page used in the GoUnPhish app. This page is used to register and login into the application. The user needs to register a account in the app. The user must register account with fake username and fake password. After finishing registration, the user can login into the app.

Fig 7.2 WebView

Figure 6 shows the webview which is used to load the webpage in the GoUnPhish app. This webview is used to perform multiple operations in the webpage.

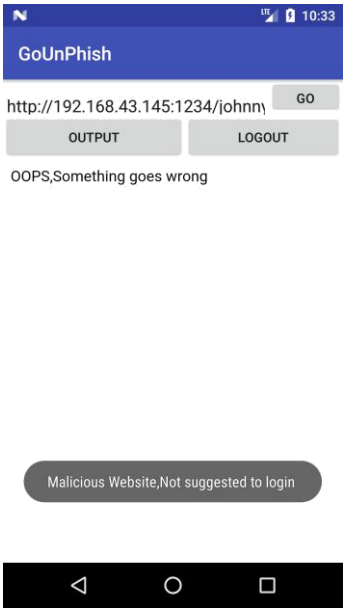

Fig 7.3 Output for Fake Page

Figure 7 shows the Output generated for fake facebook website. If the the given page is malicious, GoUnPhish app generates an alert to the user.

Figure 8 shows the Output generated for Original facebook website. GoUnPhish app suggest the user to continue to login if the login page is original.

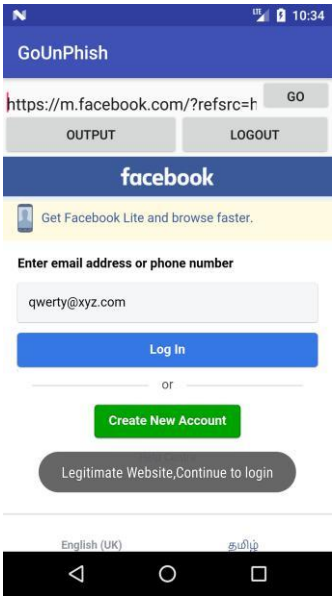

Fig 7.4 Output for Original Page

Xampp server is used to create a fake page. Source code is obtained from the original website to create a fake webpage. Xampp server acts as a local server in that particular network. Creating and Hosting a fake page is illegal, we created this fake page for testing purpose only.

VIII. Conclusion and Future work

In today‟s world usage of mobile phones got increased rapidly. Almost all website have login page to authenticate the user. This login page is easily cloned by the hackers. Hackers use the cloned website to get the username and password from the targeted user. Hackers use their own server to host this fake website. Mobile phone usage is increasing rapidly year by year due to this, hackers are forced to target the users mobile phone users. Our Mobile Application prototype helps to avoid phishing attack in the mobile phone. In future this prototype can be implemented to avoid the phishing attacks in the mobile phones.

GoUnPhish application is just a mobile application prototype to identify whether the website is legitimate or malicious. This prototype works for normal phishing attack. This prototype is completely focused on detecting the normal phishing attacks so further research should be done to improve this prototype. Man-In-The-Middle attack is the advance level attack used by the hacker to know the targeted user‟s credentials. This prototype should be improved to give solution to this type of attack. In Future this prototype will be developed to avoid all type of attacks that is performed in the mobile phones.

References

[1] Archana, M. ,Duarai Raj Vincent, P.M. and Naveen Kumar Boggavarapu (2011) „Architecture for the Detection of phishing in the Mobile Internet‟, International Journal of Computer science and Information Technologies, Vol. 2.

[2] Sharvari Prakash Chogre and Narendra Shekokar (2016) „A Survey on Anti-Phishing techniques in Mobile Phones‟.

[3] Adrienne Porter Felt and David Wagner „Phishing on Mobile Devices‟ (2012).

[4] Cik Ferasa Mohd Foozy, Mohd Faizal Abdollah and Rabiah Ahmad (2016) „Phishing Detection Taxonomy for Mobile devices".

[5] Hossain Shahriar, Tulin Klintic and Victor Clincy (2015) „Mobile Phishing Attacks and Mitigation Techniques" , Journal of Information Security.

[6] Bottazzi, G. ,Casalicchio, E.,Cingolani D. ,Marturana E. and Piu M. (2011) „MP-Shield: A Framework for

[7] Longfei Wy, Xiaojiang Du and Jie Wu (2013) „MobiFish: A Lightweight Anti-Phishing Scheme for Mobile phones".

[8] Luk Malisa, Kari Kostiainen and Srdjan Capkun (2012) „Detection Mobile Application Spoofing Attacks by Levering User Visual Similarity Percepton".

[9] https://www.phishingbox.com/news/phishing-news/summary-of-global-phishing-survey-2h-2014 .

[10] https://en.wikipedia.org/wiki/Phishing.

[11] https://tools.ietf.org/html/rfc2616.

[12] Jema David Ndibwile, Youki Kadabayashi and Doudou Fall (2017) „UnPhishMe: Phishing Attack Detection by Deceptive Login Simulation through an Android Mobile App", 12th Asia Joint Conference on Information Security.