Design and Implementation of Heuristic based Phishing detection techniques

by

Jaynish Patel

B.Eng. Gujarat Technological University, 2014

A Project Report Submitted in Partial Fulfillment of the Requirements for

the Degree of

MASTER OF ENGINEERING

in the Department of Electrical and Computer Engineering at University of Victoria, Victoria, British Columbia, Canada

Supervisory Committee

Design and Implementation of Heuristic based Phishing detection techniques

by

Jaynish Patel

B.Eng., Gujarat Technological University, 2014

Supervisory Committee

Dr. Fayez Gebali, (Department of Electrical and Computer Engineering)

Supervisor

Dr. Issa Traore, (Department of Electrical and Computer Engineering)

Co-Supervisor

Table of Contents

List of Figures

List of Tables

Abstract

In today's world, the internet has brought a tremendous change in e-commerce aspect of people's lives. However, it is prone to a wide variety of security attacks. One of the most dangerous security threats is phishing. Phishing is a nontrivial problem involving deceptive emails and webpages that trick unsuspecting users into willingly revealing their confidential information. In this project, various phishing detection techniques are discussed and one technique based heuristic rule are implemented to detect the phishing URL. The different features are extracted from the given URL. The feature groups include address-bar related features, abnormal- based features, HTML – JavaScript based features and domain based features. The different heuristic rules are implemented and decision is made based on the output of the heuristic rules. Furthermore, different weightage is also assigned to each heuristic rule to detect the URL correctly.

The API id developed in Java to classify URL as phishing and Legitimate. To test the application, a dataset from Alexa and Phish tank was collected. An automated script was written, which takes the URL in the JSON format and send to API running on some server and gets the Output in JSON format as Legitimate or Phishing along with the score.

Chapter 1 Introduction

1.1 Context

In today's era, everybody is using internet with the speed of generations like 2G, 3G, 4G, LTE and Wi-Fi. The Internet is a system of connected computer networks that use the standard Internet protocol suite (TCP/IP) to serve billions of users worldwide. The Internet plays a vital role in many aspects of human life. It offers services for various age groups and applications like social networking sites by which anyone can connect chat or communicate regardless of their geographic locations. Due to its ubiquity, the Internet has become a utility in every day life, it also has its dark side as well, as it exposes its users to wide variety of threats. One such threats that was enacted in the last decade and has had far reaching impact is Phishing. Phishing is a form of social engineering in which an attacker, also known as a phisher, attempts to fraudulently retrieve legitimate user's confidential or sensitive credentials by mimicking electronic communications from a trustworthy or public organization in an automated fashion. The information collected by phisher is then used to access the important information of the company, organization or person, which further can result into identity theft and financial loss. The main vehicle by which phishing occurs is electronic messages, such as email and mobile text messages (i.e. SMS). In a typical phishing attack, the potential victim is baited with an email, phone call, or, a text message and tricked into voluntarily responding with information or by clicking a URL. There are 3 main steps involved in executing a phishing attack as shown in Figure 1.

Figure 1 Phishing information flow

Initially, then Phisher crafts some persuasive message that is mass-mailed to a large pool of potential victims. The phishing may be designed so as to look like it is coming from a legitimate organization such as financial institutions, government services, etc.

The fake email asks the victim either to update personal information by directing them to a malicious website as supposedly a condition to avoid losing some services.

By clicking the link provided, the victim is directed to a bogus website implemented by the attacker. The phishing website is structured as a clone of the original website so that the victim is not able to differentiate from the service that has access to. In some variation of the attack, the phishing is crafted at specific individual or group of individuals in position of authority. This is known as spearheaded targeted phishing attacks, and can have very damaging consequences, if successful.

1.2 ISOT Message Security System

Nowadays data security and privacy is a crucial concern of any organization. Data breach can cause huge financial loss as well as loss of customer trust for any organization or company. Phishing is a type of cyber attack, which can result in the theft of personal information of the customers of organizations such as financial institutions and government services. The last decade has seen a dramatic increase in the amount of phishing attacks, which is compounded by lack of effective detection and prevention systems. Therefore, there is an urgent need to develop effective techniques and tools that will warn about the phishing emails so that users do not disclose their personal data during these attacks.

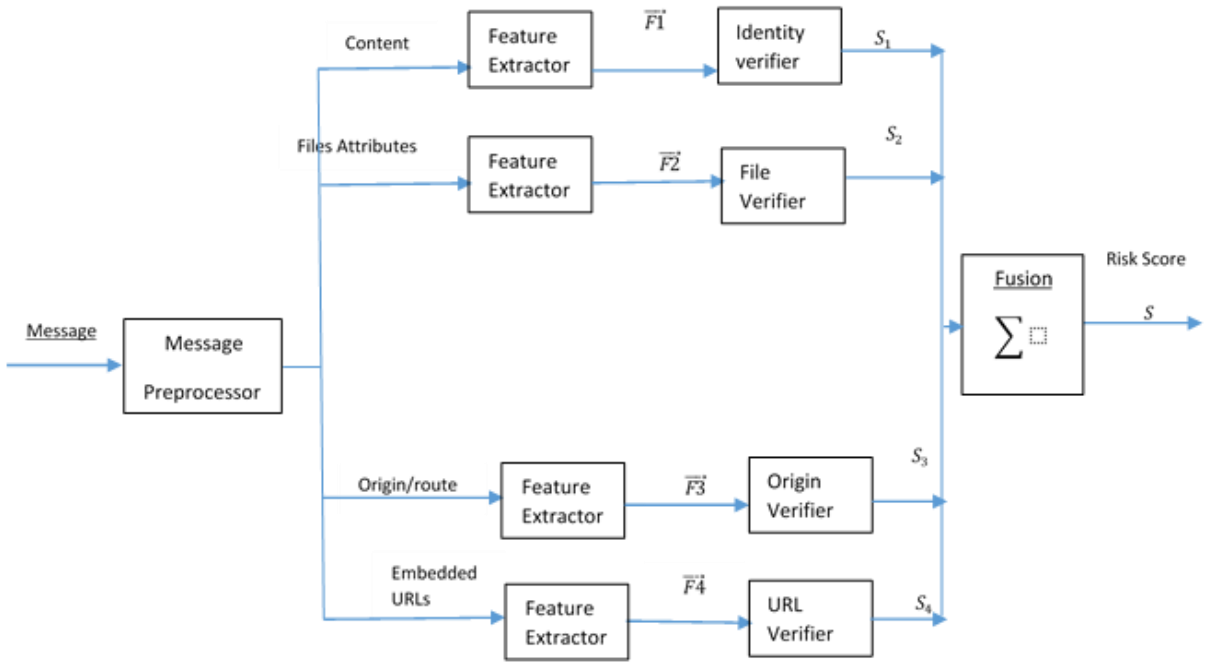

Figure 2: ISOT Message Security System (from [1])

In order to detect phishing attacks, with an emphasis on targeted phishing, the ISOT Lab has been developing a framework and tool that is centered around user stylometric identity, depicted by Figure1 1 and 2. The system allows checking the genuineness of the messenger's identity, and checking whether key message characteristics match the messaging behavior of the claimed messenger identity with respect to the recipient. The genuineness of the messenger's identity is established through stylometric authorship verification, which is known to be very challenging when dealing with short unstructured text. Messaging behavior is based on message origin location patterns (i.e. typical locations from where genuine messages were sent by this messenger), file attachment pattern (i.e. likelihood for the messenger to attach files of certain characteristics), and embedded URLs pattern (i.e. likelihood for the messenger to embed URLs, and validity of such URLs). The proposed model is used to verify both incoming and outgoing messages. Checking outgoing messages allows determining whether a legitimate account has been hijacked and is being leveraged to send out phishing messages to victims known by the account owner. Checking incoming messages protects against phishing attacks by establishing whether the sender is genuine and known by the recipient.

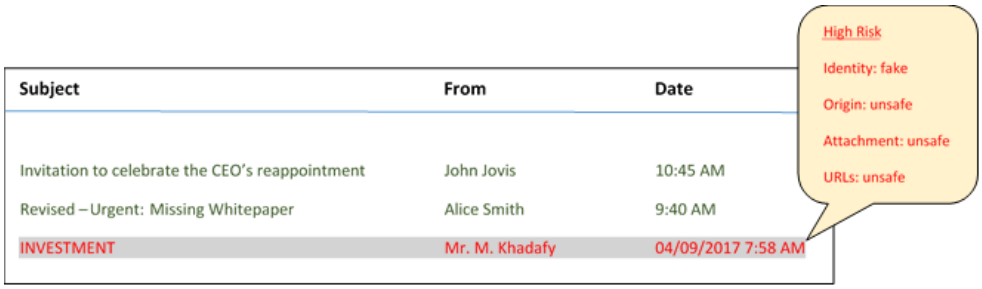

Figure 3 : Mail client display showing sample incoming messages classifications and characterizations (from [1])

1.3 Objective and Contribution

The objective of the current MENG project is to develop one of the modules involved in the aforementioned email security system. Specifically, the goal of the MENG is to develop part of the module that analyzes URL patterns embedded in email messages.

In my work, phishing is detected by extracting features of the URL given from phishing email using the heuristic rules. These heuristics rules are mainly based on the content of the website as well as the features of the URL through which it can be found that a website is legitimate or used for phishing. These heuristic rules are formed by extracting the features from the URLs, and then applying them to if-then-else rules. These rules help in decision making with three possible outcomes: phishing, suspicious and legitimate.

The different features are extracted directly from the URL and checking html content, and domain characteristics to give the result.

1.4 Report Outline

The Structure of the report is as follows:

- Chapter 2 provides some background on different types of Phishing attacks and discuss about various Phishing Detection techniques. Also, it includes how the phishing attacks occurs and motivation to prevent the phishing attacks.
- Chapter 3 summarizes the System architecture and design of the Heuristic based Phishing detection. Also, it discusses about the different heuristic rules implemented in the system.
- Chapter 4 presents the data collection process and testing process. It gives the output of the result showing the false positive, false negative, true positive and true negative.
- Chapter 5 concludes the report and discusses some future work.

Chapter - 2 Background of Phishing

Although it seems that phishing started in the last decade, the concept of phishing attack has been known since 1991 where the AOL first became a victim of such attack. In those early days, AOL was the number one Internet service providing company. The attack was carried out by hackers who copied the code of AOL website and made a fake website, which looked like the original website. Then, they sent messages to users including the fake website trying to convince to login into the site. They, then harvested valid users' credentials, and used those to carry out credit card scam.

The term phishing comes from the fact that cyber-attackers are fishing for data. Some of the earliest hackers were known as phreaks. Phreaking refers to the exploration, experimentation and study of communication system, and because of that, phreaks and hackers are closely related to each other. The "ph" in phishing was derived from "phreaking", helping link these two communities [2].

2.1 Types of Phishing

There are different categories of phishing attacks and different techniques involved as discussed below.

Generic Phishing through Email/Spam

This is the most common type of phishing in which the phishing email is sent to millions of users, requesting them to provide personal data by filling a form or visiting a link. Most of the time, the phishing email is sent with urgent notice to update personal information or to verify the account as soon as possible. These collected data is then used by the phishers for fraudulent activities. The phishers use various techniques to enable the aforementioned phishing scenario, one such technique is link manipulation. Using this technique, the phisher sends the malicious link to the user. When the user clicks on the link, the request will be redirected to the website controlled by the phisher instead of the original website. However, hovering mouse over a link and check the address of the link inform the user of link manipulation. The site to which victims are redirected is masquerading as a genuine site. This is also known as Man in the middle attack. In this type of attack, the phishing site is in the middle between the victim and the original/genuine website. The phishers keep track of each transaction that has been made between the legitimate website and the user. The phishers gather all the information in the middle, and attempt to conduct valid

transactions on behalf of the victims, e.g., withdrawing or transferring funds from their bank accounts.

Several techniques are used to incite users to fall the phishing. For instance, malicious search engine optimization (SEO), also called SEO poisoning, is used to direct users to seemingly benign advertisements website, which offers low cost product and services. However, those sites are compromised and will either redirect silently the user request. The compromised themselves can be traps. If users try to buy the product or service using those websites by inputting credit card information, the corresponding information will be collected by phishers.

Spear Phishing

In spear phishing, the hackers know the target, meaning hackers keep track of target organizations or companies, or individuals. They study and do research on the organization or individual to make an attack more personalized. This type of phishing is more insidious and has much greater likelihood of success as the phishing message is crafted using precise knowledge of the targeted organizations or individuals.

Vishing (Voice Phishing)

In this type of phishing, the phishers target victims by calling random numbers, and then ask them to give their credit card information using a dial pad on the phone. The main purpose of this type of phishing is to get personal information or bank account information of the user via phone. This is called phone phishing.

Smishing (SMS Phishing)

In this type of attack, the phisher sends a SMS to the victim that contains a phishing link. The aim is to get the personal information of the victim using the link which is redirected to phishing website.

2.2 Motive of Phishing

The motivation of phishing are diverse including financial gain, identity theft, malware distribution, and harvesting passwords, among others.

Financial gain is the leading motive of phishers as different researches have indicated that the main victims of phishing attacks were the financial institutions. A successful phishing attack against any organization in the financial service industry carry the potential of damaging its brand.

Identity theft is a key motivation of many phishing attacks. Phishers steal user identities and commit fraudulent activities crime by means of these identities or sell the stolen identities in the dark web for financial gain to criminals who acquire and utilize stolen identities to hide their own. Malware distribution is another key motivation of phishing attacks. This occurs by distributing malware by means of phishing messages that are sent in bulk and therefore, zombie networks are the most suitable to transmit large phishing campaigns. These messages enclose malicious links which, when clicked by an inexperienced user, results into malware spreading over the victim's machine.

Hackers may also use phishing campaigns to harvest passwords. Phishers perform this harvesting using diverse methods, for example, by installing key loggers and other sniffing code on victims' machines or as browser extensions. The data gathered from the victims is either used for financial benefit, identity hiding, fraud or sold to attracted parties who may used it for the same purpose.

2.3 Phishing Detection techniques

The phishing detection technology landscape is still in an infant stage. Very limited solutions are available that can effectively detect and block phishing. However, there are a few solutions that have emerged in the last few years. Some of these solutions are outlined in the following.

Content based phishing detection

The content-based approach detects web spoofing by checking the similarity between original and spoofed pages. The similarity between two webpages is usually calculated based on the similarity of the web page content. The approach uses information retrieval based on Term Frequency – Inverse Document Frequency (TF-IDF) metric [3]. Term Frequency indicates the count of a specific term whereas the document frequency indicates how important is the document in the collection of documents.

Visual similarity based phishing detection

In [4], it was proposed to use a phishing detection mechanism based on visual similarity among phishing sites that imitate the same target web site. In the proposed approach, the detection system accesses the suspicious URL by taking the image displayed on the browser. Next, the system compares the obtained image with images from a database. Each entry in the database has one of three labels: legitimate, phishing, and unknown. If an image is recorded in the database, the detector flags the site as an imitating site. If the image of the model site is not in the image database, this method can detect malicious web pages by similarity between phishing sites imitating the same site. Since phishing web pages must imitate targeted sites, visual similarity between phishing sites and their target sites is assumed to be an inherent and not simply concealable property. However, these techniques require images of real target sites for detection.

Google Safe Browsing

Google Safe Browsing uses a URL blacklist to discover the phishing attack [5]. A sample URL is taken as input and checked in the blacklist repository. If the URL is present in the black list repository, the URL is flagged as suspicious, else it is considered as a legitimate website. The main shortcoming of this approach is its inability to detect the phishing URLs which are not present in the blacklist. This is an important shortcoming, as large number of new phishing domains are registered and used by phishing on a daily basis.

Phish Net

Phish Net attempts to overcome the limitations of blacklists. It involves two major components: the generation of URL variations relative to the original ones, which grows the blacklist, and using a data structure that assigns a score to each new URL based on its similarity with existing URLs [6]. The component, uses different heuristics for generating new URLs such as replacing top-level domains, IP address equivalence, directory structure similarity, query string substitution, brand name equivalence, and so on. The second component computes the similarity and uses different matching techniques, including matching IP address, matching hostnames etc. [6].

Spoof Guard

Spoof Guard [7] implements an approach in which, phishing symptoms are tested against suspicious websites to determine whether a website is a legitimate or phishing website. Examples heuristics include Image Check, Links Check, URL Check and Password Field Check. Each of

these heuristics are given a weight based on which the website is classified accordingly. If the total score of the phishing symptoms listed above exceeds a threshold, the site will be classified as phishing website else, it will be considered as a legitimate website. This approach has the potential to detect novel phishing campaigns, however, it can also generate a high rate of false positive [7].

Phish Guard

The methodology used in Phish Guard consists of detecting a phishing website by injecting fake credentials before the user inputs correct credentials in a login form of a website. It then r screens the server response to the submitted fake credentials to decide if the web page is legitimate or not. Although the idea is nonspecific and works with any authentication technology, the current prototype is developed for sites utilizing HTTP authentication and accepting user-id and password combination as credentials [8].

Chapter 3 System Architecture and Design

3.1 Approach Overview

The focus of the project is on checking URLs embedded in e-mail messages using heuristic rules to determine whether it is phishing-related or not. Different features are extracted from the URLs and submitted to a rule base, which outputs a score and a classification, which could be phishing, suspicious or legitimate.

The URL verifier was developed as a Java API, which uses spring boot framework and RESTful Web services. The API takes the URL input as a string in a JSON format and returns a JSON output containing the classification (i.e. Phishing, suspicious, or legitimate) and the Score (as a percentage). The different features extracted from the URL are as follows:

- Address bar related features
- Anomaly based features
- HTML and JavaScript based features
- Domain based features

Once the input URL is submitted, the system extracts the website features using Java standard built-in functions and collects all features, which are then submitted to the classification module (i.e. the rule base), which in its turn makes the final decision.

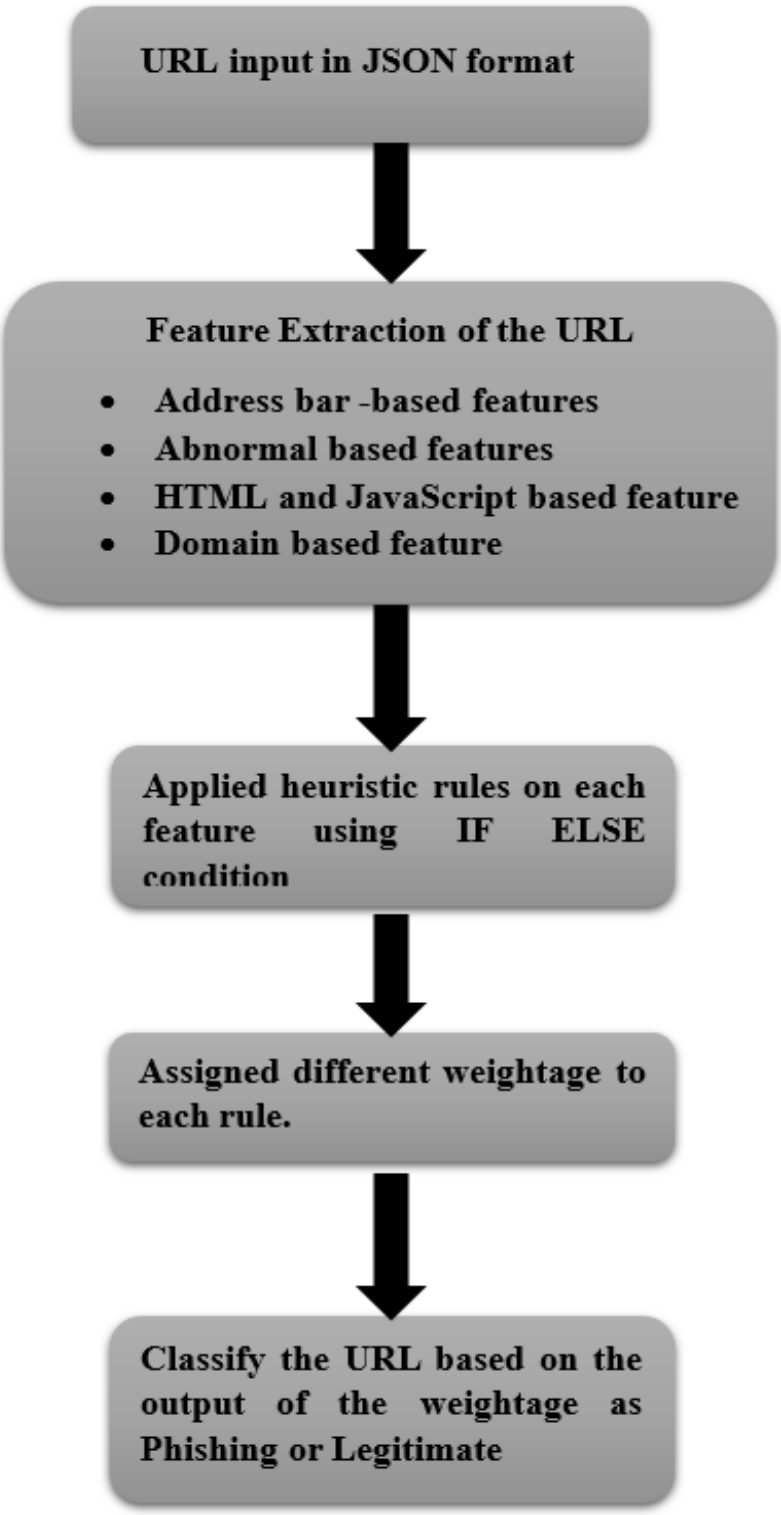

Figure 4: System Architecture of Heuristic Based Phishing Detection System

The main steps involved in the detection system are summarized in Figure 4, and described as follows:

- The fist step is to hit the path of Rest web service running on some server. For example, http://locahost:8080/url . After that, the value of the URL is parsed in the JSON format, so that the URL is loaded as String and parse it to the Java Rest API code.
- The second step is to extract all the features of the URL. The URL features are classified into different groups, including Address-bar based features, Anomaly based features, HTML and JavaScript based features, and Domain based features. A total of 16 heuristic rules are used to classify the URL as phishing, suspicious or legitimate. All the heuristic rules will be discussed later.
- The result is displayed in the JSON output format with the classification decision and a score which is the based on the percentage of triggered heuristic rules.

3.2 URL features

There are 4 different categories of URL features

- Address bar related feature
- Abnormal based feature
- HTML and JavaScript based feature
- Domain based feature

Address bar related feature

Address bar features are related to the URL address, which the users is supposed to reach. Different features from the URL, including the length of the URL, number of dots and slashes, the host name, whether the URL contains special characters or not, whether it contains an IP address or not, and other special symbols, such as ' @' character, and HTTPS with SSL check.

Abnormal based features

The URL features, which relate to anomalies or discrepancies between the W3C objects and Web Identity are known as abnormal based features. Abnormal based features are related to the source code of the webpage, which has HTML and JavaScript content. These features include URL

(RURL), URL of an anchor (AURL), and Server Form Handler (SFH). The request URL includes the external objects such as external scripts, CSS, images and other attachments are tied to their own domain. The Anchor verifies that all the anchors in a specific web page should be pointed to the same domain on the web page itself.

HTML and JavaScript based features

The features, which are related to HTML tags and JavaScript functions, are treated as HTML and JavaScript based features. These features include Redirect Page, Disabling Right Click and Using Pop-up window and on Mouseover. Redirecting is a technique in which the user is navigated to another webpage instead of the requested webpage. On Mouseover event is a track in the JavaScript code used by used by attackers to replace the status bar of a URL with an unauthenticated URL. The Right click feature is the same as on Mouseover feature, which uses the JavaScript code to disable the right click event so the user is unable to see the source code of the webpage.

Domain based features

The URL features related to the domain name are known as domain based features. These features include Alexa Page Rank and Age of the Domain. The Age of Domain is the important feature, which is used to establish how old the domain is and also check weather the domain exists or not. Alexa rank page feature is used to check the traffic through website, which is an indicator of its viability, and gives its ranking it in the different countries and worldwide.

3.3 Heuristic rule base

We consider suite of heuristic inspired from the literature [9, 10]. Several groups of rules are considered. We describe below some of the main groups.

Rule 1: Age of Domain

The domain name is extracted from the URL and the domain creation date is checked by performing a query on WHOIS database [10]

If the Age of Domain is more that or equal to 6 months, it could be legitimate. Otherwise, there is some likelihood that it is phishing.

Rule 2: Alexa rank page

The Alexa rank page of the webpage is extracted by using the Alexa rank page web API. The rank page describes the number of visitors to the website. If there are more visitors the webpage is legitimate and if there are fewer visitors the webpage could be phishing [10].

If the Alexa ranking of the webpage is less than or equal to 100,000 it is Legitimate

Otherwise phishing

Rule 3: length of URL

This rule considers that long domain names are probably malicious [10].

If the length of URL is less than 54, it is considered as Legitimate

Else if length of URL is equal to or between 54 and 75, it is Suspicious

Else, phishing

Rule 4: Number of dots and slashes

The domain part is extracted from the URL and the number of dots in the domain part is checked. The number of dots in the domain part of the URL represents the number of subdomains. URLs with more subdomains have greater likelihood of being malicious [10].

If the number of dots in the domain part is less than 3, it is Legitimate

Else if number of dots is equal to 3, it is Suspicious

Else number of dots is greater than 3, it is Phishing

Rule 5: @ Symbol

If the domain part URL contains @ symbol, it is considered as phishing URL, as @ symbol is used to redirect to another domain [10].

If the domain part of the URL contains @ symbol, it is Phishing

Else, Legitimate

Rule 6: Special Characters

Some special are indicative of maliciousness such as redirecting the request to a compromised site.

If the URL has any special characters such as underscore, comma, semicolon, it is classified as Phishing

Else, Legitimate

Rule 7: IP Address

URLs containing numerical addresses are most likely malicious, because this means that some one is giving their IP address of the system, which is not a general scenario [10].

If the URL contains IP Address as part of it, it is classified as Phishing
Else, it is classified as Legitimate

Rule 8: HTPPS with SSL Validation

The URL has either HTTP or HTTPS header. If the URL has HTTPS, the SSL certificate should be verified. This involves checking various attributes of the certificate, such as whether the certificate is expired or not, whether it is self signed or CA signed, also its public key is verified [10].

If the certificate is active and not self signed

Else if self signed or failed to verify public key, it is classified as suspicious

Else if certificate is expired and self signed or failed to verify its own public key, it is classified as Phishing.

Rule 9: Request URL

The request URL feature is to check the URL loaded from other domains. According to the security reports, legitimate URLs contain only 22% of the content such as images, CSS etc., loaded from the other domains. Whereas Phishing Websites contain a greater percentage of the content loaded from multiple domains [10].

If the web page contains 22% of the request URLs loaded from other domains, it is classified as Legitimate

Else if the web page contains request URLs% between 22 and 61, it is classified as Suspicious

Else, Phishing

Rule 10: Anchor URL

The anchor text, link label, link text, or link title is the visible, clickable text in a hyperlink. The words contained in the anchor text can determine the ranking that the page will receive by search engines [10].

If less than 31% anchor part of the URL is connected to other domains, if is classified Legitimate

Else if, anchor part of the URL is in between 31% and 67%, Suspicious

Else, Phishing

Rule 11: Redirect Page

The more redirect URLs are in a webpage the more malicious the URL is.

If the number of the redirects in the website <= 1, it is classified as Legitimate

Else if, the number of the redirects in the webpage is in between 1 and 4, Suspicious

Else, Phishing

Rule 12: On Mouse Over Event

The on Mouseover feature is used by the attacker in the JavaScript code to change the status bar of when the on Mouse event happen on any element on the webpage [10].

If an on Mouseover event is triggered that changes the status bar, it is classified as Phishing.

Else if, it doesn't change the status bar, it is classified as Suspicious.

Else, Legitimate

Rule 13: Redirect Page

A legitimate website never disables the right click event functionality as it allows users and developers to view the source code of the webpage [10].

If in a web page, the JavaScript Right Click code is disabled, it is classified as Phishing.

Else if, it is classified as Legitimate.

Rule 14: Server form Handler (SFH)

Server form handler is served as an important feature to differentiate phishing sites from the legitimate websites, and takes the following form [10]:

<form action= "/login/login.jsp" method ="post" target="_login">

The above form tag is used in server-side script to perform an action based on the user's navigation. The "action" in the above tag describes the path and "method" describes the type of HTTP method used in handling a page request.

If the SFH of a URL contains blank or empty, it is classified as Phishing.

Else if SFH of a URL belongs to a different domain, it is classified as Suspicious.

Else, Legitimate

3.4 System Implementation

The system requirements show the software and hardware used in the system. The software implementation was based on Java 8, and involved the Spring boot framework, Apache tomcat server, NetBeans IDE and Postman to make API call.

NetBeans IDE is used to setup the Spring boot framework project. We added a plugin of Spring-boot framework in NetBeans and then after, created a new project using Spring boot framework. The software deployment requires Java version 1.7 or higher. All the dependencies for apache tomcat server, and for the Rest web services are added to the pom.xml file. The steps in setting up the implementation through a Maven project are outlined as follows:

Step 1: Create a Maven Web project

The structure of the generated project is depicted in Figure 5:

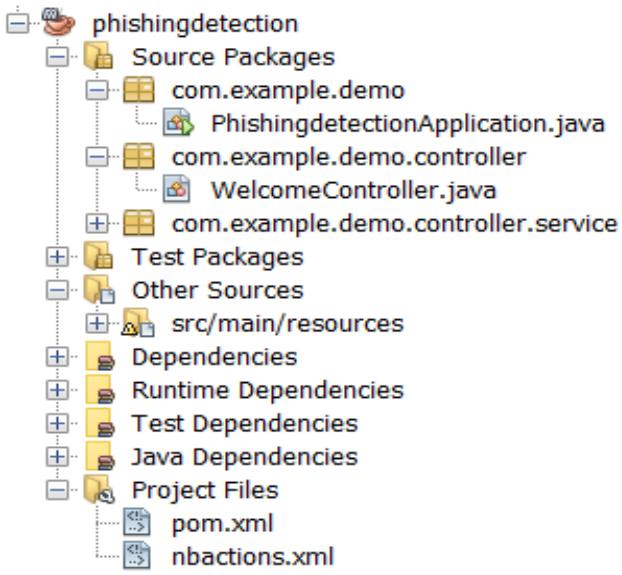

Figure 5: Maven project structure

Step 2: Add Spring dependencies

After creating the web project, the first step is to add Spring dependencies into the file pom.xml, as shown in Figure 6.

```
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>2.0.1.RELEASE</version>
    <relativePath/>
    <!-- lookup parent from repository -->
</parent>
<properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>
    <java.version>1.8</java.version>
</properties>
<dependencies>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-web</artifactId>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-test</artifactId>
        <scope>test</scope>
    </dependency>
    <dependency>
```

Figure 6: pom.xml file dependency

After adding the above dependencies, a set of jars are automatically imported into the project under Maven Dependencies, as shown in Figure 7.

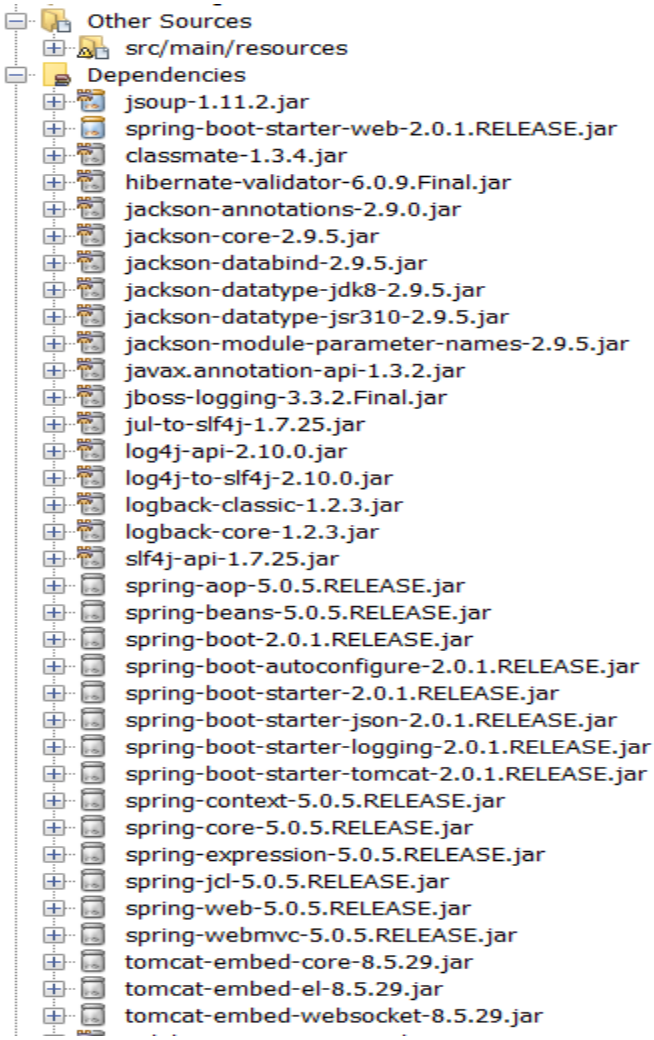

Figure 7: External jar Dependencies

Step 3: Implement REST Resources

Our API will only accept JSON requests and respond back with JSON responses. The most important class in our API is the controller which acts as the interface for client/server communication, and is depicted in Figure 8. Each controller acts as a resource that exposes some services and is accessed via a specific URL.

```java
@RestController
public class WelcomeController {

    @RequestMapping(value = "/url", method = RequestMethod.POST, consumes = "application/json", produces = "application/json")

    public ResponseURL postURL(@RequestBody processURL p) throws Exception {

        UrlMain urlm = new UrlMain(p.getUrl());
        double total_count[] = urlm.checkVulnerability();      //   Calling Main Class
        double max = 0.0;
        int index = -1;
        // Finding the maximum index  of output
        String output[] = {"Phishing", "Suspicious", "Legitimate"};
        for (int j = 0; j < 3; j++) {
            if (total_count[j] > max) {
                max = total_count[j];
                index = j;
            }
        }
        //returning the value
        ResponseURL rs = new ResponseURL();
        rs.setDecision(output[index]);
        rs.setScore((int) (max * 100));
        return rs;
    }

}
```

Figure 8: Controller Annotation

The following are the common annotations used by our controller:

@RestController: this annotation marks the class as a Resource, it defines implicitly both @Controller and @ResponseBody MVC annotations. When annotating a class with @RestController, it is not necessary to write @ResponseBody beside the classes returned from the methods.

@RequestMapping: this annotation defines the URL of the resource in addition to the method type: GET/POST, in our project we expose the URL service as POST which is accessed through /url.

@RequestParam: this annotation represents a specific request parameter, in our example, we map a request parameter called String url to an argument of type String.

@RequestBody: this annotation represents the body of the request, in our example, we map the body of the request to a class of type Process URL (Jackson handles the JSON)

Step 4: Deploy the REST API

At this stage, the API is deployed on Tomcat 1.8/JRE8

To test our API, we used the Postman client plugin from Chrome and we initiated POST requests as shown in figure 9.

Successful request: we pass a URL as a request parameter in the request body. This is shown in Figure 9.

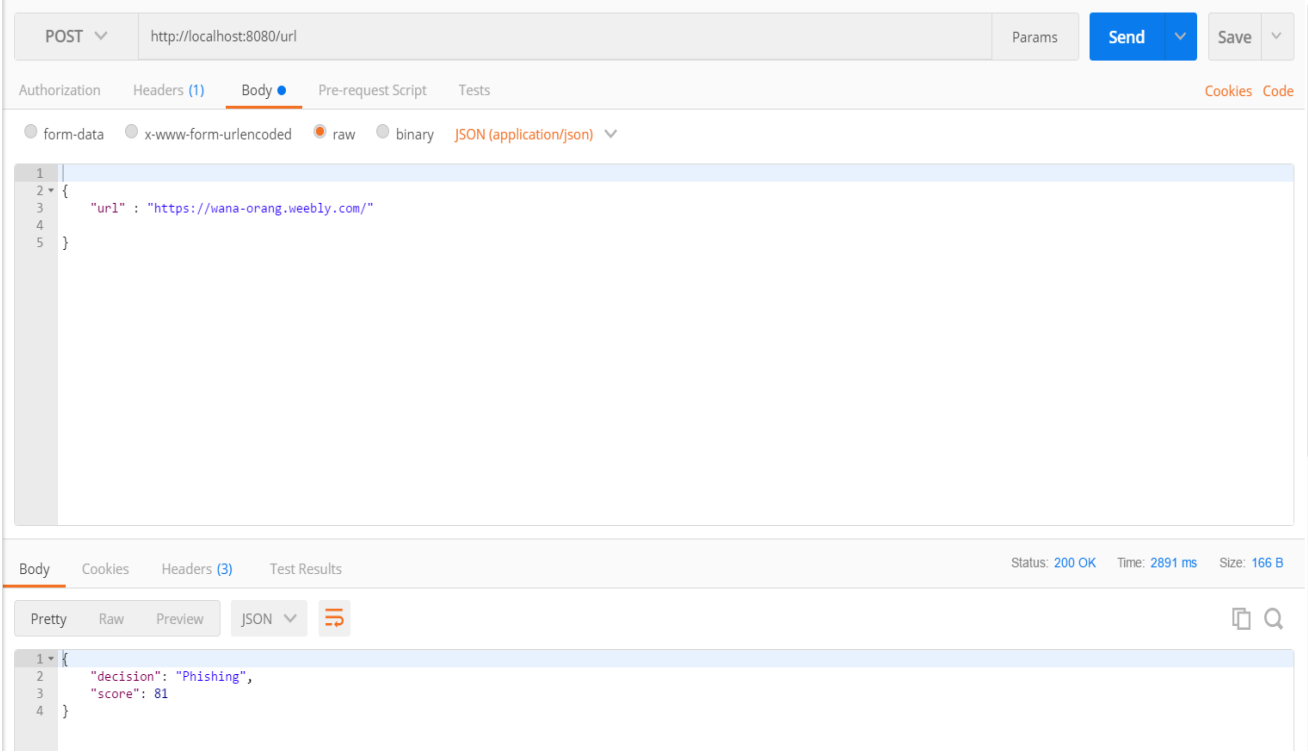

Figure 9: Postman Client interface

Chapter 4 Testing

Testing is required to check whether the functionality of the system is working or not, and take corrective measures if necessary. Testing is performed to ensure that software program complies to all the requirements gathered from the client or customer.

4.1 Datasets

In this project, the main testing task consists of testing using an existing dataset the effectiveness and efficiency of the proposed system in detecting phishing.

Two different datasets are collected for our tests: a dataset of phishing URLs and a dataset of benign URLs

The Phishing dataset was collected from the Phishtank website (https://www.phishtank.com/phish_archive.php). The Phish tank website provides both online and offline datasets. The online dataset consists of URLs for websites that are still up and running, whereas the offline dataset consists of URLs whose domains are not registered and there are no corresponding running servers.

The benign URL dataset was collected from the https://www.Alexa.com site website that maintains the list of the top domains, which are the most frequently visited sites.

As shown in Table 1, a total of 500 benign URLs (from Alexa) and 1020 phishing URLs (from phish tank) were collected. As shown in Tables 2 and 3, the phishing dataset consists of 649 online URLs and 371 offline URLs.

Table 1: Total Dataset

Database (online and Offline)	Number of URL	Phishing/Legitimate
Phish tank	1020	Phishing
Alexa	500	Legitimate

Table 2: Phishing Offline Dataset

Database (Offline)	Number of URL	Phishing/Legitimate
Phish tank	371	Phishing

Table 3: Phishing online Dataset

Database (Online)	Number of URL	Phishing/Legitimate
Phish tank	649	Phishing

4.2 Evaluation Metrics

In any binary classification problem like the problem discussed in this paper, where the goal is to detect phishing pages in a dataset with a mixture of phishing and legitimate pages, classification performance can be measured using the following metrics: true positive rate, false positive rate, true negative rate, false negative rate, and accuracy. These are considered as standard metrics to evaluate the effectiveness of any type of phishing detection system. Those metrics are defined below using the following notation:

- N_P denote the total number of phishing websites and N_L represent the total number of legitimate websites. $N_{P\rightarrow P}$ phishing websites classified as phishing
- $N_{P\rightarrow L}$ phishing websites classified as legitimate
- $N_{L\rightarrow P}$ legitimate websites classified as phishing
- $N_{L\rightarrow L}$ legitimate websites classified as legitimate

The performance metrics are defined as follows:

True positive rate (TPR):

True positive rate is the rate of phishing websites classified as phishing out of the total number phishing websites

$$TPR = \frac{N_{P\rightarrow P}}{N_P} * 100$$

False positive rate (FPR): False positive rate is the rate of Legitimate website classified as Phishing website.

$$FPR = \frac{N_{L\rightarrow P}}{N_L} * 100$$

False negative rate (FNR): False negative rate is the rate of phishing websites classified as legitimate out of the total number phishing websites.

$$FNR = \frac{N_{P \to L}}{N_P} * 100$$

True negative rate (TNR): True negative rate is the rate of legitimate websites classified as legitimate out of the total legitimate websites.

$$TN = \frac{N_{L \to L}}{N_L} * 100$$

Accuracy (A): measures the rate of phishing and legitimate websites which are identified correctly with respect to all the websites.

$$A = \frac{N_{L \to L} + N_{P \to P}}{N_L + N_P} * 100$$

4.3 Performance Results

Figure 10 summarizes the evaluation results in a matrix.

Out of the 1020 phishing URLs, 98 were identified by the system as legitimate, which corresponds TPR=90.4%.

Out of the 500 benign URLs, 18 were classified as Phishing, which corresponds to FPR=3.6%. The overall accuracy of the system is 92.4%. Also, the average time to test one URL is 0.1s to 4.9s depending the response of the WHOIS database server. If the IP address is parse instead of URL, the output will be Phishing as Domain based related featured and Address bar related feature have more weighting than the Abnormal based feature and HTML and JavaScript based feature. Out of all rules, if the rules classify URL as 50% Legitimate and 50 % Phishing, then the output of the system will be Phishing.

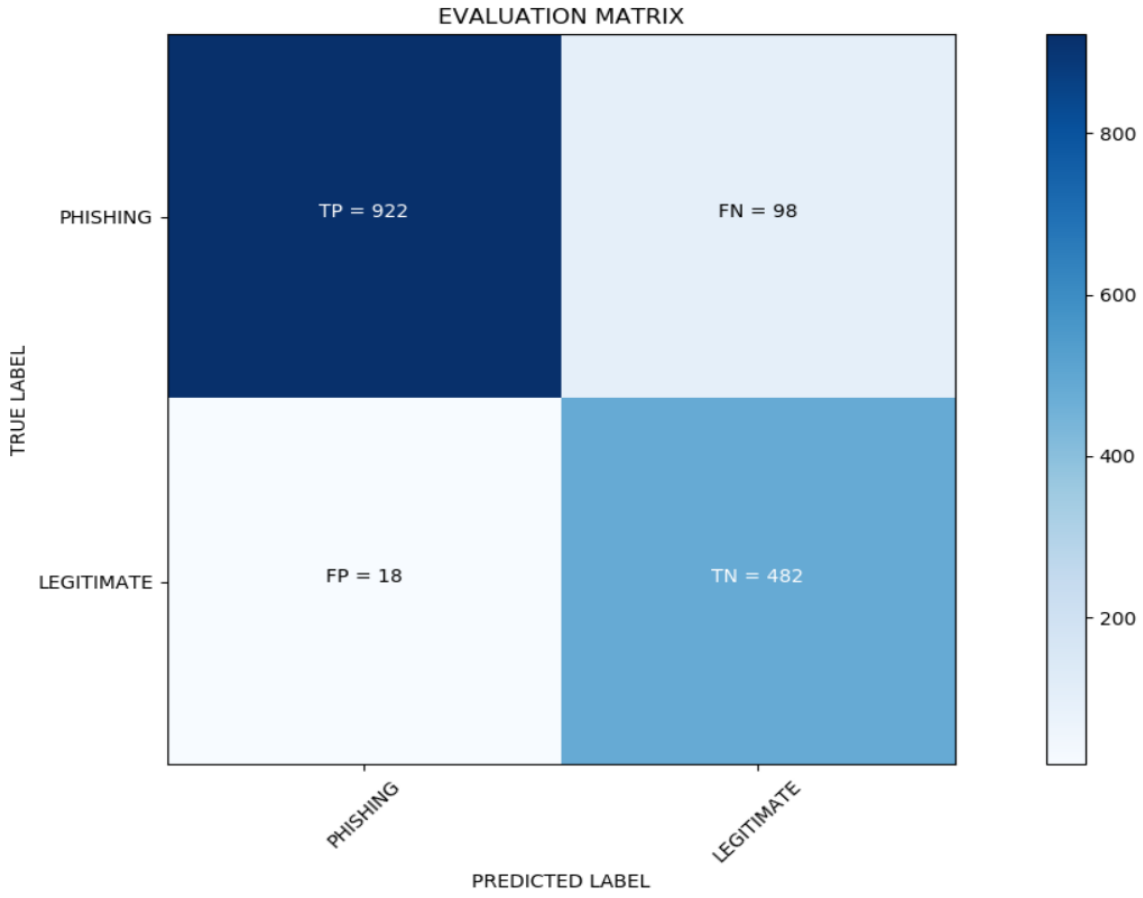

Figure 10: Evaluation Matrix

Chapter 5 Conclusion and Future Enhancements

Phishing websites mimic legitimate websites to trick victims into submitting credentials to access private content. Phishing attacks through URLs is one of the major issues faced by the Internet Community because of the vast amount online transactions performed on a day-to-day basis. The phishing attacks cause severe losses to companies, customers and web users. Social networking sites such as Facebook, Twitter and LinkedIn have been the victims of phishing.

In this project, an outline of the existing phishing detection techniques was discussed, and a heuristic based phishing detection system was implemented.

The testing was done on a mixture of benign and phishing URLs. The majority URLs tested have been classified correctly by the system, yielding an accuracy of 92.4%, which is very encouraging.

Future Enhancement

The accuracy can be improved through a series of enhancements as follows:

- The main weighting scheme used in the heuristic model help check whether the domain exist or not, and the creation date of the domain if the domain does exist. To uncover the above information, a WHOIS server is used to query the result. Currently, we are using the free servers available to check the WHOIS database. However, to increase the accuracy of the system, paid APIs can be used as alternative, and this can reduce the false positive rate.
- Machine learning algorithms such as the discriminative classifier algorithms such as random forest, Support Vector Machine, and so on, can be used to predict the URL.
- The time taken for testing ranges from 0.1 sec to 4.9 seconds per page depending on the page being tested due to WHOIS queries and page rendering. Threading is one of the possible enhancements, which reduce the computational time. Our application can be used as a background script for browser plug-ins and thus an API should be implemented. It could be used to identify phishing URLs in web server referrer logs, or it could be used by web hosting providers to check for phishing sites on their servers.

Chapter 6 References

[1] I. Traore, M. L. Brocardo, "Secure Personalized Trust-based Messages Classification System and Method", US Provisional Patent Application No. 62/569,250, Filed: October 6, 2017.

[2] http://www.phishing.org/#

[3] Yue Zhang, Jason I. Hong, Lorrie F. Cranor, CANTINA- A Content based approach to detect phishing websites. Published in Proceedings of the 16th international conference on World Wide Web,2007.

[4] M. Hara, A. Yamada, and Y. Miyake, "Visual similarity-based phishing detection without victim site information," in IEEE Symposium on Computational Intelligence in Cyber Security, 2009. CICS '09, pp. 30 – 36, 2009

[5] Google safe browsing API," http://code.google.com/apis/safebrowsing/ developers guide.html.

[6] P. Prakash, M. Kumar, R.R. Kompella, M. Gupta, Phishnet: Predictive Blacklisting to Detect Phishing Attacks, In INFOCOM Published in 2010 Proceedings IEEE, pp. 1– 5, March 2010

[7] N. Chou, R. Ledesma, Y. Teraguchi, and J. C. Mitchell, "Client-side defense against web-based identity theft," in NDSS. The Internet Society, 2004.

[8] P. Likarish, D. Dunbar, and T. E. Hansen, "Phishguard: A browser plug-in for protection from phishing," in 2 nd International Conference on Internet Multimedia Services Architecture and Applications, 2008. IMSAA 2008, 2008, pp. 1 – 6.

[9] https://www.techopedia.com/definition/3811/heuristic-programming

[10] Jeeva, S. Carolin, and Elijah Blessing Rajsingh. "Intelligent phishing url detection using association rule mining." Human-centric Computing and Information Sciences 6.1 (2016): 1-19.