Fall 2015

Interactive Phishing Filter

Rushikesh Joshi
San Jose State University

Interactive Phishing Filter

A Project

Presented to

The Faculty of the Department of Computer Science

San Jose State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

Rushikesh Joshi

December 2015

The Designated Project Committee Approves the Project Titled

Interactive Phishing Filter

by

Rushikesh Joshi

APPROVED FOR THE DEPARTMENTS OF COMPUTER SCIENCE

SAN JOSE STATE UNIVERSITY

December 2015

ABSTRACT

Interactive Phishing Filter

by Rushikesh Joshi

Phishing is one of the prevalent techniques used by attackers to breach security and steal private and confidential information. It has compromised millions of users' data. Blacklisting websites and heuristic-based methods are common approaches to detect a phishing website. The blacklist method suffers from a window of vulnerability. Many heuristics were proposed in the past. Some of them have better accuracy but a lower performance. A phishing filter should have better accuracy and peformance. It should be able to detect fresh phishing websites. Jo et al. [2] present a list of attributes of the web page to find the disparity between an original website and a spoofed website. The main aim of this project is to integrate the approach presented by Jo et al. [2] into web browser via Firefox add-on. Our phishing filter collects the list of attributes and compares it with the help of approximate string matching algorithms and WHOIS [14] server queries. For machine learning techniques, we used Weka [21]. All the algorithms, available in Weka were applied to our testing data set. Our phishing filter achieves 94.3% accuracy with reasonable performance.

ACKNOWLEDGMENTS

This project would not have been possible without the kind supervision and help of many individuals. I would like to express my sincere thanks to all of them.

I would like to express my gratitude to my project adviser Dr Thomas Austin for his continuous supervision, useful comments, and engagement through the learning process of this masters project and for suggesting to me a good topic. Furthermore, I would like to thank Dr Christopher Pollett for his valuable suggestions during the execution of the project. Special thanks to Professor Teodoro Cipresso for encouraging me to execute this project and for his valuable feedback.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

Introduction

The Internet, today, has become another world in itself, capable of providing all basic necessities with a click of a button. According to statistics, almost 3 billion users have access to the Internet [40]. On the one hand, the advent of on-line services like e-banking has made life of people convenient by allowing them to manage their transactions, sitting at home. The other side of this has exposed them to countless security threats. Internet banking, e-commerce and email services share a significant amount of usage in today's modern world which requires transmission of confidential and critical data. Therefore, it is imperative to keep these security aspects in mind while making such an application.

There are many security threats on the Internet like phishing, malware attack, man-in-the-middle attack, etc., among which Phishing is the most prevalent of all. Phishing is a fraudulent act wherein the attacker contacts the user in e-mail, a phone call or other communication channels, pretending to be an authorized person, to learn sensitive and confidential information like user-id and password. According to Kaspersky Lab, an increase of around a million phishing cases have been reported since the first quarter of 2015. Many solutions exist to detect phishing websites, such as CANTINA approach [6] and blacklist. However, none of these solutions provide good accuracy and performance when it comes to real-time safe browsing.

Most browsers use a "Blacklist", a database of verified phishing websites to detect malicious websites. However, a "Window of vulnerability" is a major drawback of blacklists. Window of vulnerability is the time period between detection of a threat and protective steps taken for it. Insoon Jo, EUNJIN Jung, and HEON Y. Yeom [2] wrote a paper, providing a theoretical concept to solve this problem. In this project, we will implement this solution using WHOIS query, string similarity algorithm and machine learning approach [2].

1.1 What is phishing

"Phishing" was first identified and defined by International HP Users Group, Interex in 1987 [1] and can be defined as a fraudulent activity of collecting sensitive and confidential information such as username, password, credit card details, etc where the attacker pretends to be an authorized organization or person. Typically, the user under attack will receive a message or email that would appear to have been sent by an authorized user or community. This email or message would either result in the installation of a malware on the user's system or redirecting the user to a malicious website.

Phishing can be done in the following ways: [12]

- Link manipulation [12]

  This is done by designing a URL such that it appears to be authentic and similar to original website's URL. This can be done by modifying the display text between <a> tags and redireting the users to the phisher's site [12].

- Filter evasion [12]

  This is done by using images to provide malicious links, which can redirect the users to phisher's website. This way the user fails to identify malicious links hidden behind the image. However, most of anti-phishing filters apply image processing techniques to recover hidden text from images, thereby identifying the attacks [12].

- Website forgery [12]

  A very common attack, also known as "cross-site scripting" [12] is used to redirect the user to phisher's site using JavaScript code. Since the user is oblivious to the execution of this malicious code, this type of attack is one of the most difficult to identify attacks.

- Phone phishing [12]

  As the name suggests, the users receive fake calls from the phishers pretending to have valid authorization. They try to convince the user to give confidential information over the phone like account number and PIN [12].

The main focus of our project will be on "Link manipulation" type of phishing in which malicious URLs are obfuscated to appear as if they belong to valid organizations, making it difficult to be detected by human eyes. Figure 1 shows an example of a website that claims to be PayPal, a worldwide online payment system company. Figure 2 shows the original website of PayPal. On close observation, you can see some differences between the two websites, which are very difficult to identify otherwise. Some of the differences include difference in logo, favicon and a secured certificate. There can be multiple ways to re-direct the users to these malicious websites and sometimes the users fail to see these differences, therefore becoming victims of the phishing site.

Once the desired data is collected from the user, different types of forgeries can be done by the hackers, sometimes resulting into a loss of millions of dollars. Sometimes these attacks are aimed at private and confidential information of celebrities, which are later leaked online for monetary or other benefits.

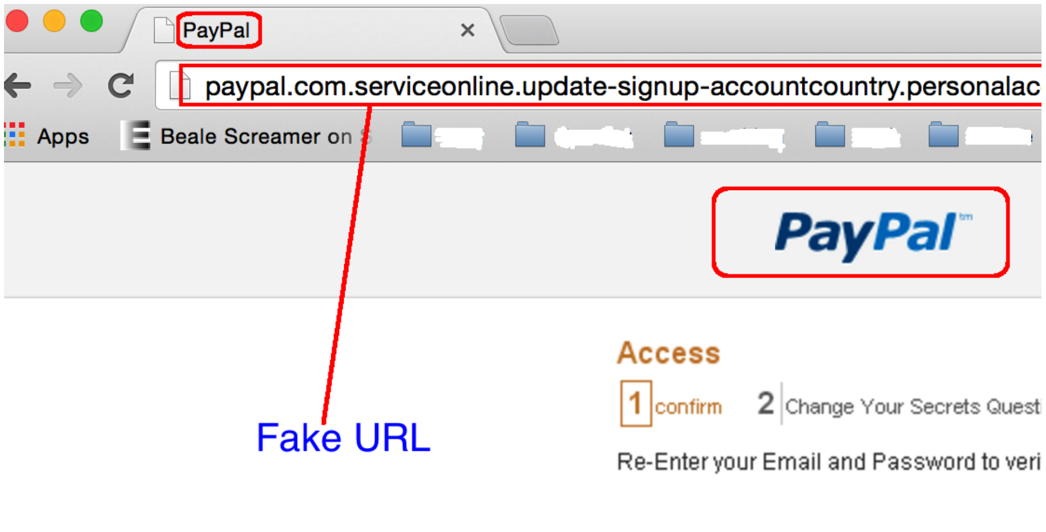

PayPal
paypal.com.serviceonline.update-signup-accountcountry.personalac
Apps
Beale Screamer on
PayPal
Access
1 confirm
2 Change Your Secrets Quest
Re-Enter your Email and Password to veri
Fake URL

Figure 1: Fake Paypal website

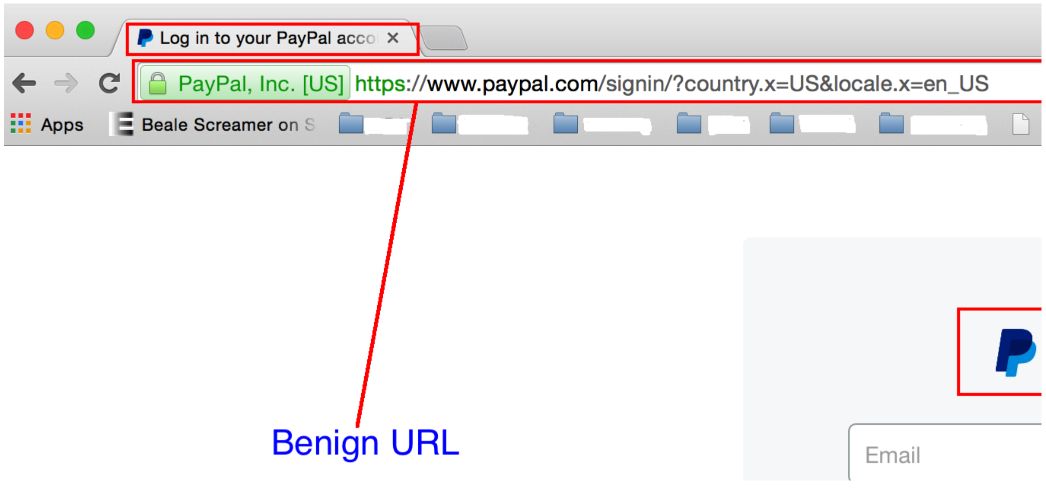

Log in to your PayPal acco
PayPal, Inc. [US] https://www.paypal.com/signin/?country.x=US&locale.x=en_US
Apps
Beale Screamer on S
Benign URL
Email

Figure 2: Original Paypal website

1.2 Problems

A lot of academic and business research has been done so far to address the problem of "Phishing". Most of the solutions either use blacklist or some type of heuristics or machine learning techniques. The blacklist method is the most common approach, adapted by most of the browsers, in which each browser defines and stores a list of verified phishing websites. Based on this list, a browser can easily identify phishing attacks. However, since the list contains previously identified websites, it is still susceptible to new attacks.

The second approach is to adopt heuristics. Different features of a websites are collected and based on this information, the authenticity of a website is determined. Accuracy and performance are the two prime filters to measure phishing. The solution should be flexible enough to identify new threats. Keeping this in mind, the authors of this [2] paper derived a technique to find disparity between two websites using approximate string matching and the WHOIS[14] server queries. Various attributes of each of the websites were collected and experimented upon with a different machine learning classifiers. The test results using this filter showed some promising results. With the help of these attributes and approach, we were able to achieve 94% accuracy and 0.978 ROC area.

One more research paper [3] was proposed to use image processing techniques with emphasis on the favicon of a website. Limitation of this approach is that a website should have a favicon otherwise this method will fail. As per a recent survey [24] 40% of most visited mobile websites do not have favicon.

1.3 My contribution

The goal of this project is to give a practical shape to the idea presented in the research paper [2].

We split this entire add-on in three different parts. The first component resides on the client side. The second component is a web service. Machine learning classifier is the third component. Web services is the middle layer between browser add-on and machine learning algorithm.

We decided to make Firefox add-on as it has rich libraries for an add-on development. There were a couple of options available for the machine learning libraries out of which we selected "Weka" libraries. Weka [21] is a popular machine learning tool that has implementation of various machine learning algorithms and is easy to use with its Java API.

We collected different samples of phishing websites from Phishtank [9], which is a regcognized organization for providing a database of phishing websites. We used Alexa, an Amazon company, to get samples of benign websites [10]. We collected a total of 1773 URLs for the training data set. Out of 1773 URLs, 829 URLs were phishing and 944 URLs were benign. We experimented with different machine learning classifiers to get the best accuracy. Our results showed almost 94% accuracy to detect phishing websites using these classifiers.

CHAPTER 2

Background

2.1 How big a problem is

The amount of information transmitted through the internet has increased exponentially in the last couple of years [15]. Cloud computing has played a big role in inspiring phishing. Gigabytes of data have been uploaded to the cloud every year. As usage of the cloud increases, a lot of data becomes available to steal and manipulate. The purpose of phishing attacks could be to hack personal accounts, to spy on someone's personal life, etc.

The "Anti Phishing Group" is an international association focused on unifying the global response to cybercrime [16]. This group provides a platform to discuss cybercrime issues, potential technology solutions to cybercrime, and an access to data related to cybercrime forensics. According to a quarterly report published by this group, "Retail/Service" is the most popular industry sector for hackers [13]. In Figure 3, almost 75% of phishing attacks are targeted to "Retail/Service," "Financial" and "Payment services".

Paypal is the top victim of phishing [41]. Paypal has been the most successful way of payment for many online shopping websites for many years [38]. Banks are the second most popular target of attackers. Many users receive warnings that appear to come from a bank. Sometimes they are warned to authorize their bank accounts to save them. Many such big scams are reported every year [39]. A common trend is to save a user's bank account and credit card details on an e-commerce website. Once account credentials are revealed, getting the other details of bank account is easy.

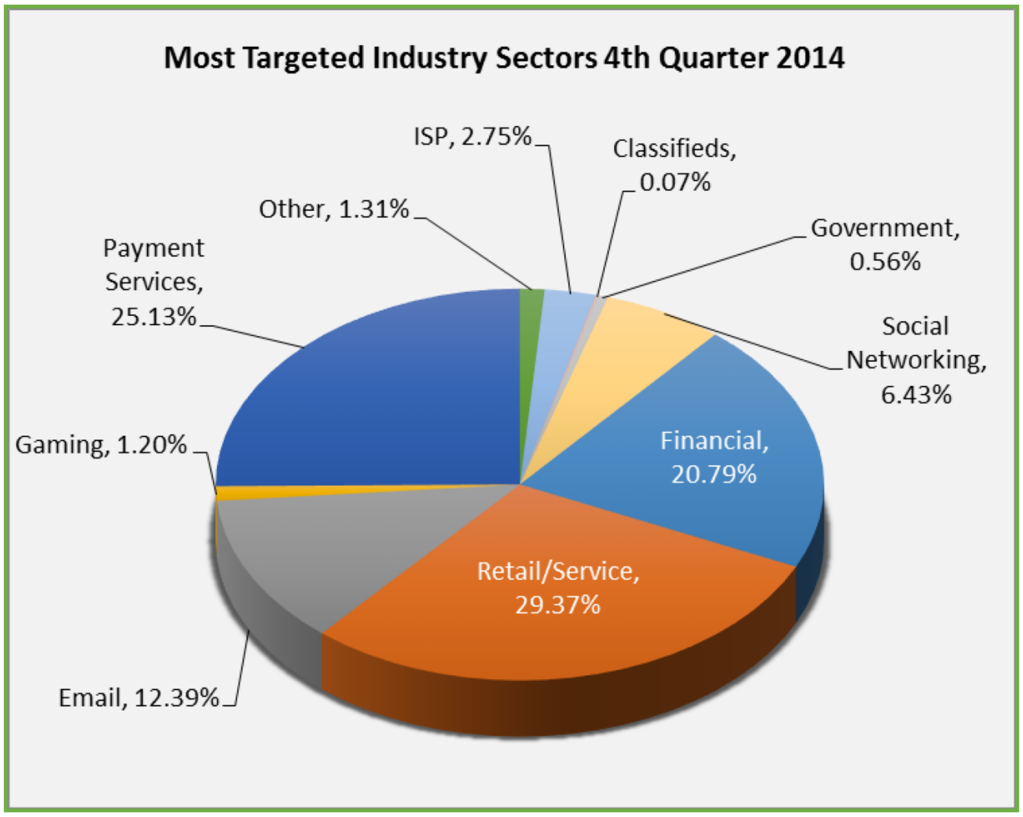


Figure 3: Phishing attack by industry[13]

Other popular sectors are "Emails" and "Social Networking". Many times phishing pages are created to hack individuals' social networking accounts. This trend has increased remarkably over the past few years as per the figure 3.

The number of phishing attacks globally reported in the 4th quarter of 2014 is 197,252 [13]. This number increased by 18% from the third quarter of 2014 [13]. One common observation from this report is that the number of unique domains have decreased from the quarter of 2014. The same domains have been used iteratively for phishing attacks.

In summary, phishing has been a generic problem to all kinds of industries. Financial and e-commerce sectors are the most affected. These attacks have increased

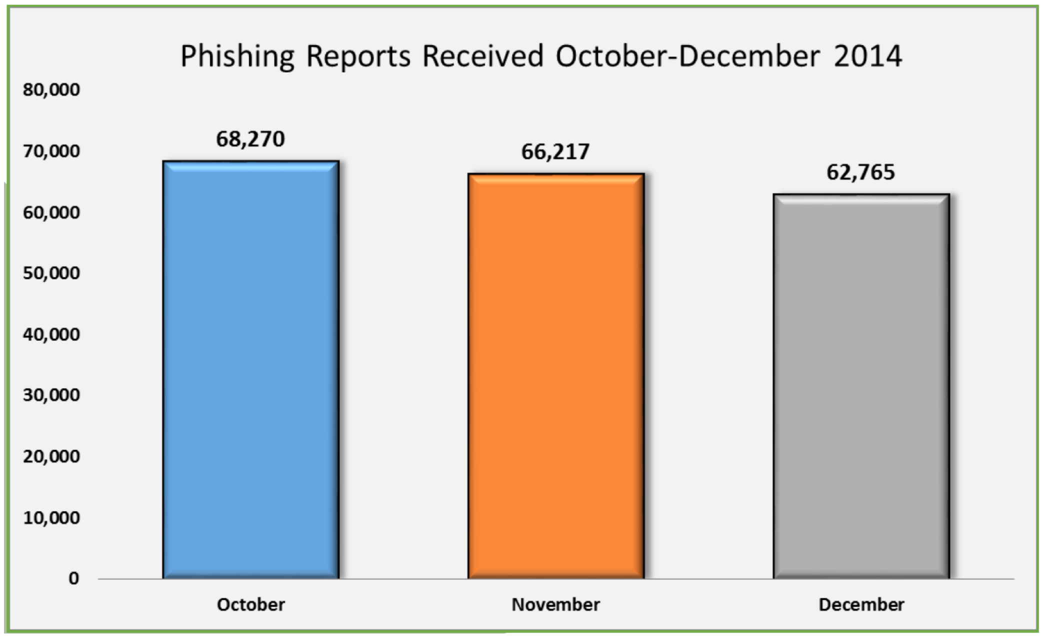


Figure 4: The number of phishing attack

in the last couple of years. Extensive research has been going on to prevent this cyber crime, but still nothing has been able to reduce the number of attacks.

2.2 Previous and Existing Solutions

A lot of research has been conducted going on by different universities and professional organizations to address phishing issues. Here is an overview of different approaches.

1. Anti-prevention (offense and defense) [4]

   Chuan Yue and Haining Wang proposed the offense and defense approach [4]. This approach does not warn users to help them detect phishing websites. It takes the help of the blacklist method to implement the idea.

   When any web page is identified as a phishing web page by the blaklist method, this filter is triggered. If the user ignores the warning and continues browsing the same page, this filter modifies the victim's response and fills out the irrelevant data in response. It creates *(S - 1)* bogus credentials and hides the real credentials between S credentials [4] where S is an integer with very high value . Within a few milliseconds, those S credentials are sent to the phishing website. The same action is taken when a user heeds the warning. However, in this case, total S bogus credentials are being generated instead of *(S - 1)*.

Overall, the aim of this approach is to confuse the phisher with many responses from the victim's machine. It would be difficult for the phisher to filter out the real credentials among thousands of data.

The limitation of this method is that it assumes that the phisher can not have access a query to the targeted website for the validation of any individual username [4]. This might not be the case for email-based services and many other social networking services. For financial websites, this could be very effective.

2. Heuristic

   Heuristic-based approaches are widely accepted and becoming more popular now. These approaches are divided into two components. The first component fetches the required properties from the web page, and the second component is the classifier algorithm to classify given data and make a prediction for the given input.

   Keywords retrieval and identity discovery-based approach [23] [7] uses search engines to get the rank of the given web page. The given web page is scanned and many terms are extracted. Those terms are sent to one or more search engines, such as Yahoo, Google, and Bing. That webpage is declared safe by the heuristic approach if the domain name is in the top N search results. However, the approach suffers from performance issue because of the total time taken by the round trip to search the terms.

3. Visual Similarity

   As the name itself suggests, the approach is related to a comparison between visual appearance of the website. Though we think that a phishing website and the original website look similar in terms of design and look, sometimes there are significant differences.

   One way to differentiate is based on the favicon of the webpage URL [3]. This algorithm compares the original favicon and the spoofed webpage URLs favicon using image processing techniques [3]. This algorithm gives some value. If the score of the page is above some threshold value then the URL is flagged as a phishing URL. The cone of this method [3] is that the website must have a favicon otherwise it gets failed.

   The second solution proposed to decompose the webpage in to block regions [5]. The visual similarity of the two webpages is then evaluated in three matrics. Those are block level similarity, layout similarity and overall style similarity. If any one of these matrics has got gerater value than the predefined threshold value, then two website is reported as a phishing website.

4. Blacklist

   "Blacklist" is the most efficient method to detect phishing website. It collects the list of verified phishing websites from all over the world. Whenever a user visits the websites, it searches in to the database and responds as per result. It gives a minimum false positive rate, but it fails to detect fresh phishing websites. It has window vulnerability problem [11]. The average time to be added in a blacklist is significantly long, it is not effective for detecting fresh phishing website.

5. Content-based approach

   "CANTINA" [6] is the implementation of this approach. CANTINA [6] is based on TF-IDF information retrieval algorithm [6]. It extracts the content of the webpage to decide whether website is a phishing site or not. The contents can be the URL and domain of the page, etc. It uses search engine results. It uses information retrieval techniques and the "Robust Hyperlinks" algorithm to improve broken hyperlinks [6]. The author has claimed that this algorithm achieves approximately 95 percent accuracy in detecting phishing websites. They also suggested to use CANTINA along with heuristic approaches to reduce false positives.

   There are certain downsides to this algorithm as well. This approach is heavily dependent on searching algorithms. If phishers achieve good page rank for the phishing websites in top search engines, then it might be difficult to detect the phishing website. The second issue is with performance, as it sends a query to search engines and processes the results. Language is also another constraint of this phishing technique.

CHAPTER 3

Problem Solution

3.1 Outline of the Solution

Our interactive filter adopts the heuristic approach to detect the phishing websites. The disparity between the website's true identity and observed identity is the main feature of this filter.

Observed Identity: The features that are part of the web page content and can be spoofed easily [2].

True Identity: The features that represents the real identity of the website and are difficult to spoof [2].

Frequent terms, source domains of iFrames and source of images are observed identities. On the other hand, the host domain is the true identity. The disparity between these two identities is measured by their textual relevance [2]. The unique challenge with this approach is to find the intersection of those characteristics that are also efficient to detect. It is not easy to find such disparity because it could be hidden in any form of the webpage. We use the approximate string matching algorithm [17] to find such disparities. Users are warned for suspected websites-based on threshold values are set. It is their decision whether they want to visit such websites or not. Through the human interaction, our heuristic can gain more accurate knowledge to improve its decision capacity. Our results show that approach suggested by Insoon Jo, EUNJIN Jung, and HEON Y. Yeom [2] is a better solution.

Let's take a small example to understand the disparity between two identities. We will compare it with the original website.

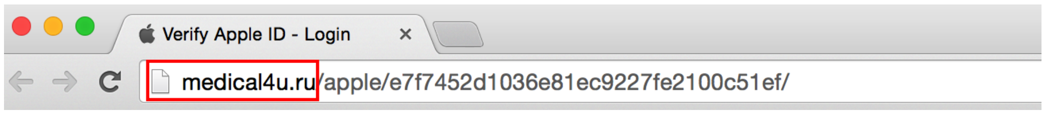


Figure 5: A fake Apple website

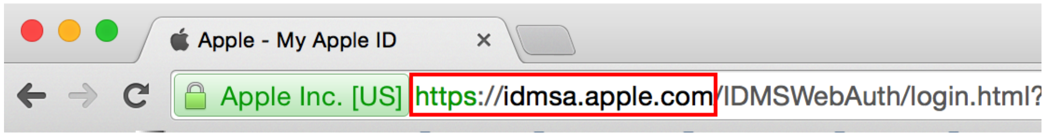

Apple - My Apple ID
Apple Inc. [US] https://idmsa.apple.com/IDMSWebAuth/login.html?

Figure 6: An original Apple website

In Figure 5 and 6, both websites claim to be the Apple website. Let's review both websites. In Figure 5, we see the observed identities such as “Verify Apple ID” from its title and “apple.com” from its image/anchor domains in the web page content. The textual relevance between the title of the website and the domain of the images and anchor tags is very high. Now let's take the observed identities and true identities for comparison. The true identity is its domain name, which is “medical4u.ru”. You can observe a less textual relevance between the true identity and observed identities. Now for Figure 6, observed identities are “apple.com” and “Apple - My Apple ID” from its respective title and domain names of images. It has a higher textual relevance between the true identity(apple.com) from the URL. All the content of this webpage has the same origin compare to WHOIS record and copyright holder.

By seeing an example, we can say that there is higher textual relevance between observed identities for the fake website whereas there is less textual relevance between the true identities and observed identities. We can easily differentiate between a phishing website and a benign website with the help of this set of settings.

3.2 The Component of the Solution

We divided all features into two groups. One is URL features and the second is content features. Let's see the definition and understanding of each features in detail. It would be easy to understand the classification scheme with a better understanding of these features.

3.2.1 URL features [2]

Before going through the URL features let's see the component of the URL.

Consider this example URL:

$\underbrace{\text{http}}_{1}://\underbrace{\text{blog.rushikeshjoshi.com}}_{2}/\underbrace{\text{p/resume.html}}_{3}?\underbrace{\text{date=oct28}}_{4}$

1. Protocol
2. Hostname
3. Path
4. Query Parameter

The URL Obfuscation technique is the common characteristics of phishing websites. No benign website does the same. The Following are the six different categories of URL obfuscation techniques:

1. The existence of IP address and different port number other than 80 [2]

   Many phishing websites contain IP address instead of hostname. It is rare to find a legitimate website that uses IP address instead of a hostname. This could be a good point to differentiate between legitimate websites and phishing website.

Example: `http://200.98.201.164/psh/pshcm1.html` [9]

2. URL contains domains name except hostname [2]

   A Phishing website tries to make the URL looks similar to the original website. To do so they tend to put the actual website hostname in the pathname of the URL. In the case of the original web site, you don't find the domains name in the pathname.

   Example: `http://www.yozgatfirmarehberi.net/paypal.com/login/auth/` [9]

3. Multiple domain names embedded the name in hostname [2]

   Multiple domain names are very frequently found in the phishing websites. Often this includes the domain name of the targeted website. The purpose of putting this in the hostname is to make the victims feel that this must be subdomain of the targeted website.

   Example: `http://paqpal-userz-inter-setting.com.3agroupeg.com/app/inc/intery/main/946a8b07ea1a08c9067ab8f45da5d4a7/index.html?dispatch=xM7XsSPwHpsgiCSB5` [2]

4. The existence of unusual characters in the URL [2]

   Usually any benign website's URL is clean. This is not the case for phishing sites. It might contain multiple unusual characters in the URL.

   Example: `http://itunes.com.care.app-server-updates-members.returnby-default-help.7215-8562-2514.renew-accounts-onlinesetup.adaspmt.org.br/efe2b42ef448716d96047%2006b4e%20152a55/home/index.`

`html` [9]

5. The use of HTTPS protocol [2]

   few phishing websites use the secure “https” connection protocol. It is very difficult to host such website with the verified secure certificate. Not much cases have been so far with this arrangement. In fact, in Figure 6 you can easily differentiate the difference between the original and the fake website using “https” connection.

3.2.2 The Content features [2]

Content features can be considered as an observed identity. Page source a contains lot of such identities.

- Entity Names [2]

  Any legitimate website generally contains its brand name or entity name in the header, meta tags, the title of the page, and copyright information. Since the phishing websites aim is to replicate the targeted website, its content also holds similar names as its target. Such entity names are collected by our filter.

- Frequent Terms [2]

  Any legitimate website generally repeats its entity names throughout the web page. Phishing websites also contains the same entity as they are in the original website. Our filter scans the whole DOM of the web page and strips out HTML keywords from the page source. It sorts the rest other terms and makes the list of the top 10 terms only.

- Resource Domains [2]

Websites usually contains different images, links and scripts in the web page. If you consider legitimate websites, the majority of the resources are hosted on the same server. On the other hand, most of the resources are referenced to targeted servers in phishing websites. These external references can be good hints to detect phishing websites. Our filter extracts the domain name of the image, link, anchor, form and script tags from, the source of the webpage.

3.2.3 WHOIS Features [2]

The WHOIS server is a service which provides the details of a domain name. You can query with an URL and in response you get useful details for that domain name. Greater disparity has been observed in spoofed website compared to the original website. The kind of date we can get from the WHOIS server includes:

- Date when the domain was registered
- Date of expiration of the domain when it will get expire
- Registrant name, address, and the organization name of the domain
- Domain of the name server

All of this information can be compared with the actual URL and the content of the webpage. Lower the textual relevance the higher chance of a phishing website.

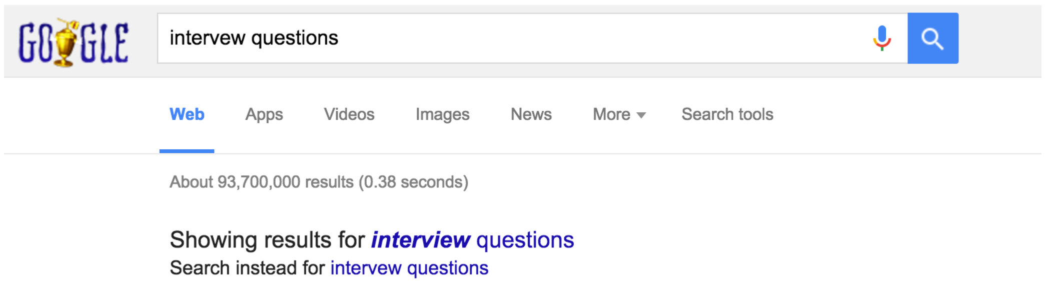
3.3 Approximate string matching algorithm

Our phishing filter collects the attributes like a title, a copyright holder, etc of a web page. In addition, it collects the information from WHOIS [14] server. We find the textual relevance between all of these attributes. Jo et al. [2] proved that a benign website has more textual relevance and a spoofed or phishing website has less textual relevance between those attributes. We used the approximate string matching algorithms to find the textual relevance.

3.3.1 What is an approximate string matching

String matching is a significant problem in information retrieval and computational biology [25]. There are various versions of string matching problems. In many problems, it is required to match both the strings. There are different algorithms available for exact string matching like Naive string search algorithm [28], Rabin–Karp string search algorithm [26], Boyer-âĂŞMoore string search algorithm [27], etc. All of their time complexity varies from $\Theta(n^2)$ to $\Theta(n)$.

Sometimes a system needs to find similar types of a string, not an exact match. These problems can be solved by an approximate string matching algorithm. Let's take a practical example. A large number of keywords are searched per day on Google, including many mistakes. I think similar kind of issue would happen with everyone. In Figure 7, we can see that I wanted to search “interview questions” instead of “intervew questions”. Google immediately recognized the right word and suggests an improvement to correct your typo.



About 93,700,000 results (0.38 seconds)

Showing results for ***interview*** questions
Search instead for intervew questions

Figure 7: Google smart suggestion

Approximate String Matching is a technique of finding strings whose patterns matches very closely but not exactly [18].

The number of operation required to convert one string to another string is called *Edit Distance.*

The closeness of two strings is measured in terms of "Edit Distance" [19] where a lower score means two strings are more similar. There are three different operations defined: [18] [19].

- Insertion

  sign → *sign**al***

- Deletion

  *lo**o**se* → *lose*

- Substitution

  ***a**ffect* → ***e**ffect*

Dynamic programming is a efficient way to find "Edit Distance" of any two strings. Let's say the `m` and `n` are the length of the first and second string respectively. The total time taken to find the edit distance between strings is $\Theta(mn)$. Space complexity is $\Theta(mn)$ when the edit distance matrix is constructed.

Table 1 shows an example of the distance matrix of two strings. Two strings are "bcdat" and "abcde". In the matrix, "E" represents the "NULL" string. As you can see, the final edit distance between two strings is 3, which means any one of the strings needs 3 operations to convert from one to another.

There are different applications of approximate string matching algorithms. Intrusion detection [20], lagiarism detection [20], bio-informatics, text-mining re-

Table 1: Edit Distance Matrix Example

	E	**b**	**c**	**d**	**a**	**t**
E	0	1	2	3	4	5
a	1	1	2	3	3	4
b	2	1	2	3	4	4
c	3	2	1	2	3	4
d	4	3	2	1	2	3
e	5	4	3	2	2	3

search [20], video retrieval [20] and digital forensic [20]. We used this string matching to find the similarity of the different strings like page title, domain name registrant, etc.

3.3.2 Q-Gram distance

As we saw that "Levenshtein distance" can be solved by a dynamic programming approach which takes $\Theta(mn)$. Ukkonen [8] proposed a better approach to improving the time complexity different to find the similarity using the pattern matching [8]. This q-gram method is the lower bound of the "Levenshtein distance". Its complexity is $\Theta(m + n)$ where m and n are the lengths of the two strings. Q-gram distance represents the difference of two string's q-profiles.

A Q-gram is the vector which contains the number of patterns of q length in the string.

Let $\sum$ be a finite alphabet and let $\sum^*$ denote the set of all strings over $\sum$ and $\sum^q$ be all the strings of length q over $\sum$, for $q = 1, 2, ...$ A q-gram is any string $v = a_1 a_2 ... a_q$in $\sum^q$ [8].

Now Let $G(x)[v]$ represents the total count of the pattern v in x. The q-gram profile of string x is defined as the vector

$$G_q(x) = (G(x)[y]), v \in \sum^q$$

Q-gram distance: Let x and y are the strings in $\sum^*$, and $q > 0$ be an integer. Q-gram distance of two strings x and y is defined as

$$D_q(x,y) = \sum_{v \in \sum^q} |G(x)[v] - G(y)[v]|.$$

3.3.3 Examples

Let us take an example to understand the concepts related to Q-gram distance of two given strings.

Example 1: Let us say s1 = “wxyzv”, s2 = “wxzvvy” and q = 2. V_1 and V_2 are the q-profiles of the respective strings.

Answer: There are four 2-grams in the string s1. Those are "wx", "xy", "yz", and "zv". s2 string has five 2-grams. Those are "wx", "xz", "zv", "vv", and "zv". As per the definition, $V_1 = (1,1,1,1)$ and $V_2 = (1,1,1,1,1)$. We have to combine both the vectors in order to take the difference of vectors. V_1 becomes $(1,1,1,1,0,0,0)$ and V_2 becomes $(1,0,0,1,1,1,1)$. We take positive difference between these two vectors and do summation which is equal to 5.

Example 2: Let us say s1 = “wxyzv”, s2 = “wxyzx” and q = 3. V_1 and V_2 are the q-profiles of the respective strings.

Answer: There are 3 3-grams in the string s1. Those are "wxy", "xyz", and

Table 2: Q-gram distance example 1

	wx	**xy**	**yz**	**zv**	**xz**	**vv**	**vy**
V_1	1	1	1	1	0	0	0
V_2	1	0	0	1	1	1	1
$\|V_1 - V_2\|$	0	1	1	0	1	1	1
$\sum \|V_1 - V_2\|$	5						

"yzv". s2 string has five 2-grams. Those are "wxy", "xyz", and "yzx". For this example, $V_1 = (1, 1, 1, 1)$ and vector $V_2 = (1, 1, 1, 1, 1)$. We have to combine both the vectors in order to take the difference of vectors. V_1 becomes $(1, 1, 1, 0)$ and V_2 becomes $(1, 1, 0, 1)$. Difference of the vector is $(0, 0, 1, 1)$. 3-gram distance for both the strings is computed by adding all the pairs which is 2.

Table 3: Q-gram distance example 2

	wxy	**xyz**	**yzv**	**yzx**
V_1	1	1	1	0
V_2	1	1	0	1
$\|V_1 - V_2\|$	0	0	1	1
$\sum \|V_1 - V_2\|$	2			

Example 3: Let us say s1 = "distance", s2 = "distance" and q = 3. V_1 and V_2 are the q-profiles of the respective strings.

Answer: There are 6 3-grams in the string s1. Those are "dis", "ist", "sta", "tan", "anc", and "nce". s2 string has five 2-grams. "wxy","xyz", and "yzx" are the 3-grams for string s2. 3-gram profiles for both the strings are $V_1 = (1, 1, 1, 1, 1, 1)$ and vector $V_2 = (1, 1, 1, 1, 1, 1)$. We have to combine both the vectors in order to take the difference of vectors. V_1 becomes $(1, 1, 1, 1, 1, 1)$ and V_2 becomes $(1, 1, 1, 1, 1, 1)$. Difference of the vector is $(0, 0, 0, 0)$. 3-gram distance for both the strings is computed by adding all the pairs which is 0. Here zero number represents that both strings are similar.

Table 4: Q-gram distance example 3

	dis	**ist**	**sta**	**tan**	**anc**	**nce**
V_1	1	1	1	1	1	1
V_2	1	1	1	1	1	1
$\lvert V_1 - V_2 \rvert$	0	0	0	0	0	0
$\sum \lvert V_1 - V_2 \rvert$	0					

3.4 Machine learning approach

Over the past two decades, data has become an integral part of every company. Everyone wants to use the data in a productive way. When we talk about data statistics and data analysis "Big Data" is the right answer. "Big Data" has been proved the robust solution for terabytes of data, "Machine Learning" has demostrated its utitlity for many problems like artificial intelligence, natural language processing, image recognition, and malware detection [29]. Machine learning has played a key role to solve all of these big problems in the past couple of years. Data can be used more smartly if you apply machine learning techniques in the right direction.

3.4.1 Machine learning

What is machine learning? Let us take an example to understand this complex problem. We receive emails on a daily basis in our inbox. Most of us have experienced spam messages. In the early days, we were used to seeing those spam messages in the main inbox folder directly. Now, such emails directly go into the spam folder. This has been made possible all because of "Machine learning". There are serveral algorithms which are used to detect the spam emails. Those are "Naive

Bayes Classifier", "Artificial Neural Networks", "k Nearest Neighbors" and "Support Vector Machines" [34]. These algorithms usally ignores the header, and high frequencey words of the email [34]. Remaning body of the email is scanned and fetched important attributes.

3.4.2 Machine learning process

Machine learning is often a continuous process. As time passes, it gathers more knowledge and tries to become more accurate in its prediction. These enable a computer program to automatically analyse a large body of data and decide what information is most relevant. There are different stages of machine learning process.

- Prepare Data

  In this step, we start with collecting the data from various websites such as "Alexa" [10] and "Phishtank" [9]. We collected around 1700 websites. The data is in the form of URLs. Our filter fetches various attributes such as title, domain name, registrar name, etc. After fetching the data, we used string matching algorithm q-gram [8] to produce a set of data showing similarity for each website.

- Select machine learning algorithm

  There are different classifiers available. We applied 75 different classifiers to our data set. In our experiment, tree classifiers have better accuracy compared to other classifiers. We selected "RandomForest" [36] tree classifier that has the highest accuracy amongst all other tree algorithms. Performance is also another decisive factor. Many classifiers have good accuracy but tree classifiers have better speed for large data sets [35].

- Algorithm predicts the result

  Next step is to build a model for given data set. We can input a new URL with all the attributes to algorithm to predict the result. As we mentioned in the previous point, we built a model using "RandomForest" algorithm. We load saved model every-time whenever we want to check whether given URL is phishing or not.

- Human Interaction to improve the accuracy

  This step is helpful to improve the accuracy. It is not a common step for all data set. It depends on the type of data set, and type of algorithm you selected. Sometimes, it does not show any improvement even if you put thousands of data verified by human. Our filter interacts with the user to validate the predicted result and updates the knowledge if it is required.

3.4.3 Types of machine learning tasks

There are mainly two types of machine learning tasks. One is supervised learning and another is unsupervised learning.

- **Supervised learning**: The training data set is given to the algorithm in this task. The expected output and inputs are defined in the training data set. The Algorithm learns from the initial inputs and predicts the result for future inputs. This approach is similar to our conventional education system where the student first learns and then appears for an exam.

- **Unsupervised learning**: No data or expected inputs are given to algorithm. It is algorithm's job to learn about required parameters for the best result. In this task, the algorithm learns by itself. The more data inputs are given, the better the accuracy will be.

3.4.4 Weka - machine learning algorithm library

Weka is an open source software that consists wide range of machine learning libraries [21]. We used Weka libraries to classify our dataset in our project. Weka was developed by the Machine Learning Group at the University of Waikato. It is an open source tool. The following functionalists are available from Weka [22]:

- Data preprocessing and visualization

  In Weka, you can randomized the data set, solve the balancing problem of data set, and handle the missing value problem in the data set. There are many more functionalisties in Weka that gives you a power to make data more effective. It gives nice visualization for distribution of the value for each attribute. You can easily summarize the statistics of a data set for one attribute by seeing an visual graphs.

- Attribute selection

  You can select and remove the list of attributes of a data set while building a classification model. In real world problem, you do not get perfect data most of the time. Many times you would like to deal with few attributes present in the file.

- Classification

  Weka has many classification algorithms available. Weka has very user friendly GUI to classify the data using different classification algorithm. You can also save the classified model for future use.

- Prediction

  Once you apply classification to data. In future, you can apply the previously saved classified model to predict the result for a given instance.

- Model evaluation

  This is the most useful feature of Weka. You can easily get an overview of any machine learning algorithm once you apply on the given data set. Weka presents all the statistical numbers once you load the model. You can even plot the different types of graphs like ROC curve.

- Clustering

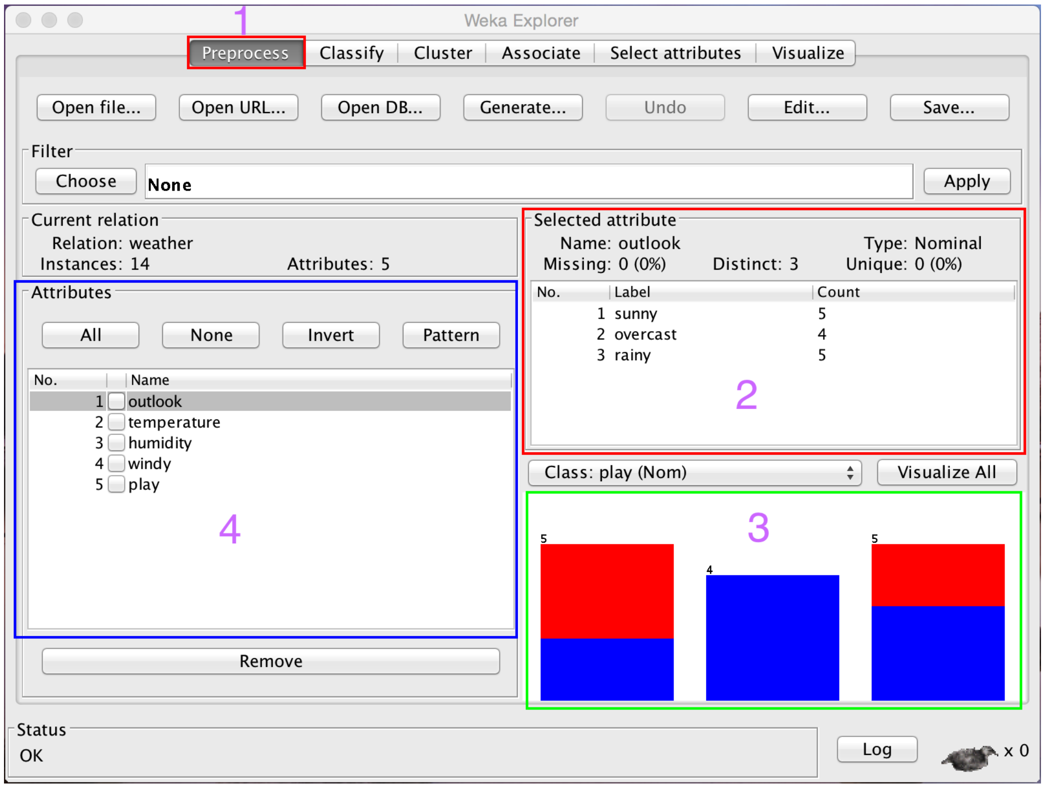
  Weka has few algorithms available for clustering the data. You can change the different attributes of the clustering algorithm to get the best result.

Weka provides APIs for different languages like Java, and Python. Weka supports ".arff" files by default to load, classify, and cluster the data. You can also load a CSV file, but there are a few drawbacks to use it. We used Java APIs to interact with Weka.

3.4.5 Classification with Weka

Let us see how can we load the data in Weka through the GUI and classify it using different classifiers. The data set taken for an example is available in the sample data set folder while you install Weka. This data set is weather. You can decide whether you can play out door games or not. This depends on certain attributes of whether. There are five attributes present in the data set file. In the file, there are 14 instances to train the model.

In Figure 8, the GUI is divided into four parts. The first section contains the core functionality that is associated with the data. Our first task is to load the data so we have to select “Preprocess” tab. The fourth section in the figure 8 holds all the attribute associated with data. The second section represents the statistics of one particular attribute.“Outlook” attribute can contain 3 distinct values. Out of 14 instances, 5 instances hold "sunny", 4 instances hold "overcast" and 5 instances hold "rainy" values. The third section depicts all instances and their value in a coloured graph. You can download the data from here [37].



Filter: None

Current relation
Relation: weather
Instances: 14 Attributes: 5

Selected attribute
Name: outlook Type: Nominal
Missing: 0 (0%) Distinct: 3 Unique: 0 (0%)

No.	Label	Count
1	sunny	5
2	overcast	4
3	rainy	5

Attributes

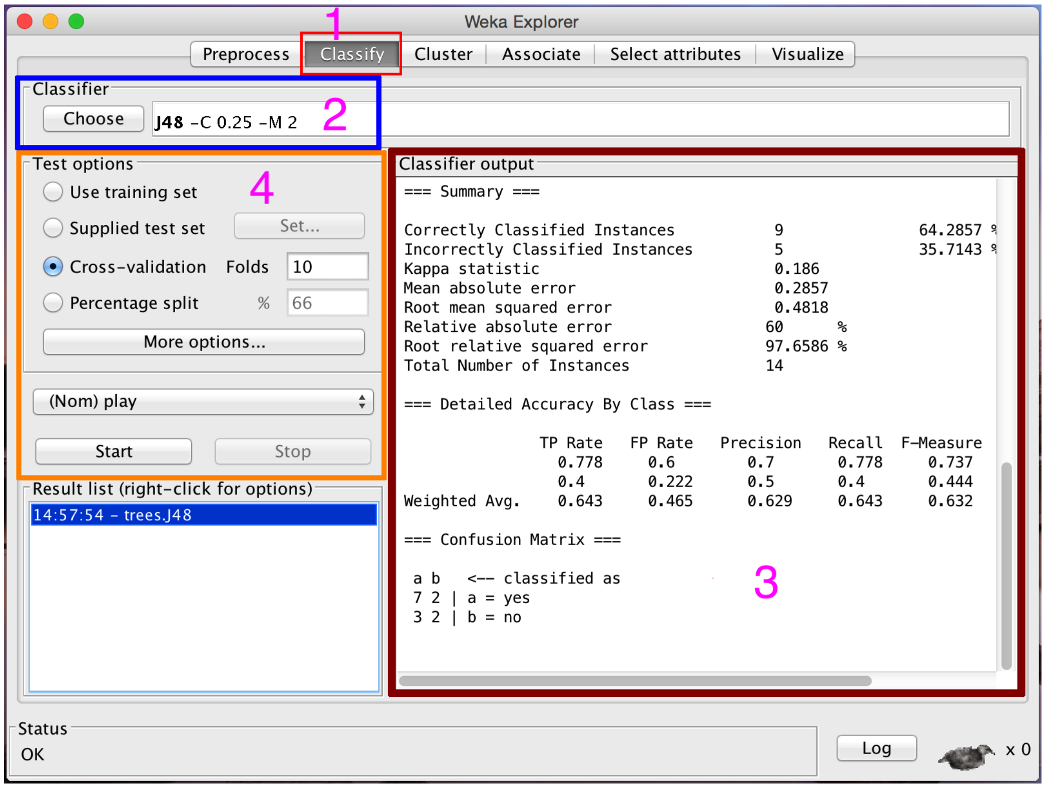
No.	Name
1	outlook
2	temperature
3	humidity
4	windy
5	play

Class: play (Nom)

Status
OK

Figure 8: Weka Preprocess Data

Figure 9 represents the classification process. As we saw in the previous figure, the first section shows that we are in classification mode. The second section contains all the list of classifiers. We applied “J48”, a tree classifier. In Figure 9, the fourth section holds various test options, like how to use this data, how many percentages of data to use for the training data set. The third section displays the classification output and summary of the results. It shows all the required information for a given classifier.



```
=== Summary ===

Correctly Classified Instances           9               64.2857 %
Incorrectly Classified Instances         5               35.7143 %
Kappa statistic                          0.186
Mean absolute error                      0.2857
Root mean squared error                  0.4818
Relative absolute error                 60      %
Root relative squared error             97.6586 %
Total Number of Instances               14

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure
                 0.778     0.6       0.7        0.778     0.737
                 0.4       0.222     0.5        0.4       0.444
Weighted Avg.    0.643     0.465     0.629      0.643     0.632

=== Confusion Matrix ===

 a b   <-- classified as
 7 2 | a = yes
 3 2 | b = no
```

Figure 9: Weka Classification Data

CHAPTER 4
Phishing filter

This chapter reviews my phishing filter that detects phishing website and warns the user. Jo et al. [2] defines the solutions in the research paper. Our role is to apply their research ideas and solutions [2] to implement the phishing filter. In a practical approach, the phishing filter is not a stand alone application that runs on the client, but it contains multiple components in its design. There are multiple layers in its architecture. All of these layers are interconnected to each other. As we have defined in the previous chapter, we used Weka for machine learning libraries, Firefox add-on to start with.

4.1 Phishing Filter architecture and workflow

Figure 10 depicts the work flow of the phising filter process. The steps of this process are:

1. The user enters a URL.
2. The URL is captured and processed by the browser add-on on the client side. The RESTful API is called with the given URL and data by add-on.
3. The Webserver calls the phishing filter Backend system.
4. The phishing filter Backend calls third party services like WHOIS server.
5. Third party services send back the response to the phishing filter Backend system.

6. The phishing filter Backend system processes the request using machine learning algorithms and the responses received from the third party services. The Backend system sends the result to the webserver.

7. The Webserver sends the response to the browser add-on.

8. The Browser add-on warns the user if the given URL is a phishing website otherwise it will do nothing.

9. The user will decide whether the warning given by the add-on is legitimate or not, and sends the response back to the add-on.

10. The browser add-on sends the last request of the final decision from the user to the webserver.

11. The webserver forwards the user's decision to phishing filter Backend system to upgrade the knowledge if it is wrong.

12. The phishing filter Backend system updates its knowledge base as per the user's response.

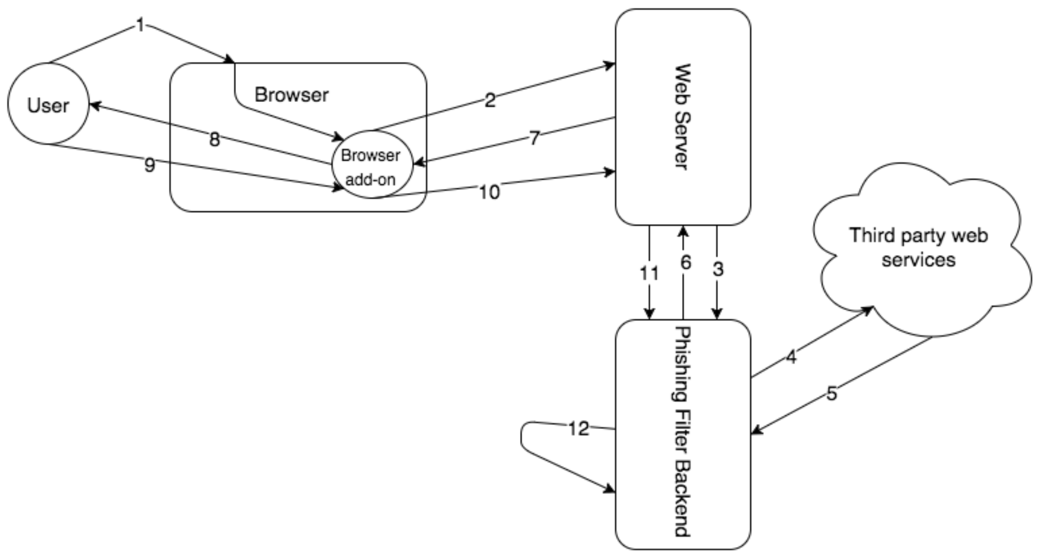


Figure 10: Phishing filter workflow

4.2 Firefox add-on

Firefox add-ons can modify the default behaviour of a web browser. You can enhance the overall experience of browsing by providing some extra features like modifications of theme and security. Some add-ons provide extremely useful features like developer tools, security tools, password management tools and many more [32]. For more information and better understanding of the add-on please visit appendix B.

4.3 Phishing Filter Backend

As described, in the beginning of this section, the main idea and solution is referenced from this [2] paper. The Backend system is uses approximate string matching algorithms, machine learning algorithms and a WHOIS server API. The Backend system fetches all the attributes of the URL and performs string matching comparison. The following are the list of attributes that are compared with each other:

- Type of protocol (HTTP or HTTPS)

- The top 10 most frequent terms in the webpage except for html tags
- The title of the webpage
- The name of the Registrant
- Whether the URL contains an IP address
- The copyright holder, if it is in the webpage
- The domains of the anchor tags
- The domains of the image tags
- The domains of the form submission
- The name of the domain name registrant
- The name server domain

Now this is the only list of attributes related to any URL or web page. The next task for the Backend system is to find textual relevance between the attributes, which are the strongest factors to decide whether a website is phishing website. most for the decision of the phishing websites. Following are the final attributes and its possible value or data type [2]:

- Whether URL uses HTTPS connection
- The Textual relevance between the most frequent terms and the name of the domain name registrant
- The Number of anchors tag, images and form element that uses the correct hostname

- Whether the URL has an ip address as a hostname
- The string similarity between the title of the webpage and registrant of the hostname
- The maximum string similarity between the title of the webpage and domain name candidates in the hostname.
- The maximum string similarity between the most frequent terms and the host domain
- The string similarity between the domain name registrant and the copyright holder
- The maximum string similarity between the different hostnames of the anchor tags and the domain name registrant

The Backend system is developed in the Spring framework [33]. Weka APIs for java are used for the implementation of the machine learning algorithms in Java. In addition, Weka is used to get the overall summary of the classifiers. Roc curves are also plotted by the Weka tool.

CHAPTER 5

Testing Result

Weka has a rich libraries of different machine learning algorithms. We tested our results by using all the classifiers and selected few final candidates for our implementation. Our data set contains 1761 websites. Our source of phishing website was Phishtank [9]. Phishtank contains large collections of on-line as well as offline phishing websites. We used Alexa [10] as a source for benign websites. Alexa contains list of top N websites across the world. All our tests were performed on the macbook pro with the following configuration and software versions:

Table 5: Hardware configuration

System	Macbook Pro (Retina, 15-inch, Mid 2015)
Processor Name	Intel Core i7
Processor Speed	2.8 GHz
Number of Processors	1
Total Number of Cores	4
L3 Cache	6 MB
Total Number of Cores	4
Memory	16 GB 1600 MHz DDR3
Startup Disk	Macintosh HD
Disk Capacity	1 TB

Table 6: Software Version configuration

FireFox version	42.0
Add-on SDK jpm version	1.0.1
Java JDK version	1.8.0_45
Weka version	3.6.12

Table 7 shows the distribution of the training data sets.

Table 7: Dataset distribution for accuracy

	Phishing website	Benign website
Number of URLs	829	932
Total URLs	1761	

5.1 Accuracy

Figure 11 shows the comparison between different classifiers-based on different accuracy class. We run the tests of all the available classifier algorithms. In the figure, you can find the classifiers whose results look promising with respect to others.

In the figure, there are total 6 types of accuracy measures. Among six measures, we give more importance to "Roc Area" and "F-measure" classes. Evaluation of classifier is a debatable topic. It can be evaluated differently for each problem. In our problem, it is advisable to balance between false positive and false negative. "F-measure" and "Roc Area" are the correct accuracy measures in that case. "Random Forest" and "LAD Tree" are the two final candidates whose accuracy are best. "Random Forest" classifier represents better balance between false positive and false negative. On the other hand, "LAD Tree" classifier has better accuracy when it comes to true positive. We can argue that it would be annoying for users to get too many false alarms.

Table 8: Accuracy measures for different classifier

Classifiers	**TP**	**FP**	**Precision**	**Recall**	**F-measure**	**ROC Area**
LADTree	0.915	0.079	0.92	0.915	0.915	0.965
RandomForest	0.943	0.058	0.943	0.943	0.943	0.978
tree.adTree	0.909	0.092	0.909	0.909	0.909	0.967
tree.FT	0.918	0.079	0.919	0.918	0.918	0.929
tree.J48	0.906	0.097	0.906	0.906	0.906	0.919
tree.J48graft	0.912	0.092	0.912	0.912	0.912	0.929
tree.LMT	0.918	0.083	0.918	0.918	0.918	0.934
tree.NBTree	0.912	0.089	0.912	0.912	0.912	0.97
lazy.KSTar	0.918	0.082	0.918	0.918	0.918	0.965
AttributeSelectedClassifier	0.92	0.08	0.92	0.092	0.92	0.922
meta.Baggin	0.923	0.075	0.924	0.923	0.923	0.975
ClassificationViaRegression	0.915	0.083	0.916	0.915	0.915	0.97
meta.Decorate	0.915	0.087	0.915	0.915	0.915	0.963
meta.FIlteredClassifier	0.915	0.084	0.915	0.915	0.915	0.956
meta.RandomCommittee	0.926	0.075	0.926	0.926	0.926	0.973
meta.RandomSubSpace	0.923	0.076	0.924	0.923	0.923	0.973
meta.RotationForest	0.932	0.069	0.932	0.932	0.932	0.973
rules.Jrip	0.92	0.077	0.922	0.92	0.921	0.939
rules.PART	0.929	0.067	0.931	0.929	0.929	0.945
tree.BFTree	0.926	0.078	0.927	0.926	0.926	0.912
meta.LogitBoost	0.912	0.087	0.912	0.912	0.912	0.966

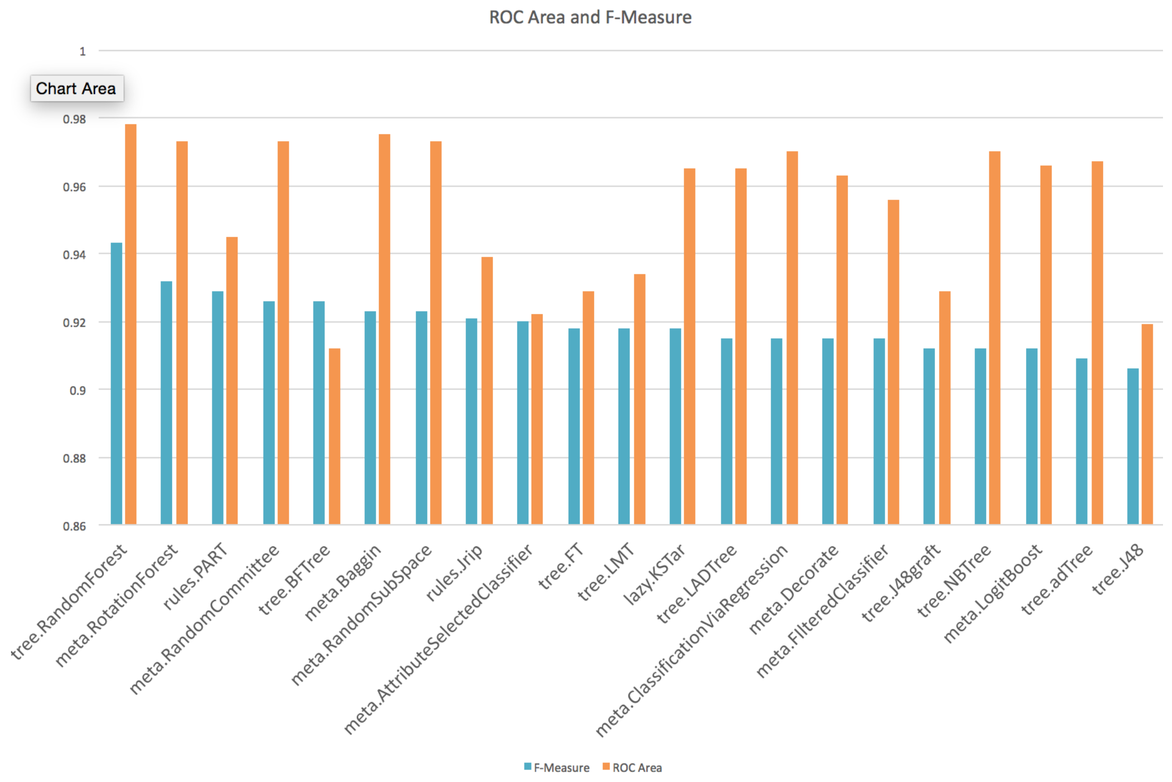


Figure 11: Comparision of accuracy between different classifiers

5.2 Performance

Performance is decisive factor for phishing filter. Performance impacts the user experience of browsing. Security should not be achieved at the cost of a browsing experience. Table 5 shows the system of server where testing was done. Server and client both are on the same system in our testing. Our filter gives a fairly good performance. We measured the total time taken by an add-one to take the decision. We did not calculate the time that involves the user's input. We did not include the time taken by updating the file on server. Average time is calculated by taking the 200 sample websites. These 200 websites include phishing and benign websites. Average time is 2.55 seconds for 200 websites. We used Alexa [10] as a source for benign websites for our performance testing. Phishtank [9] is the source for our phishing

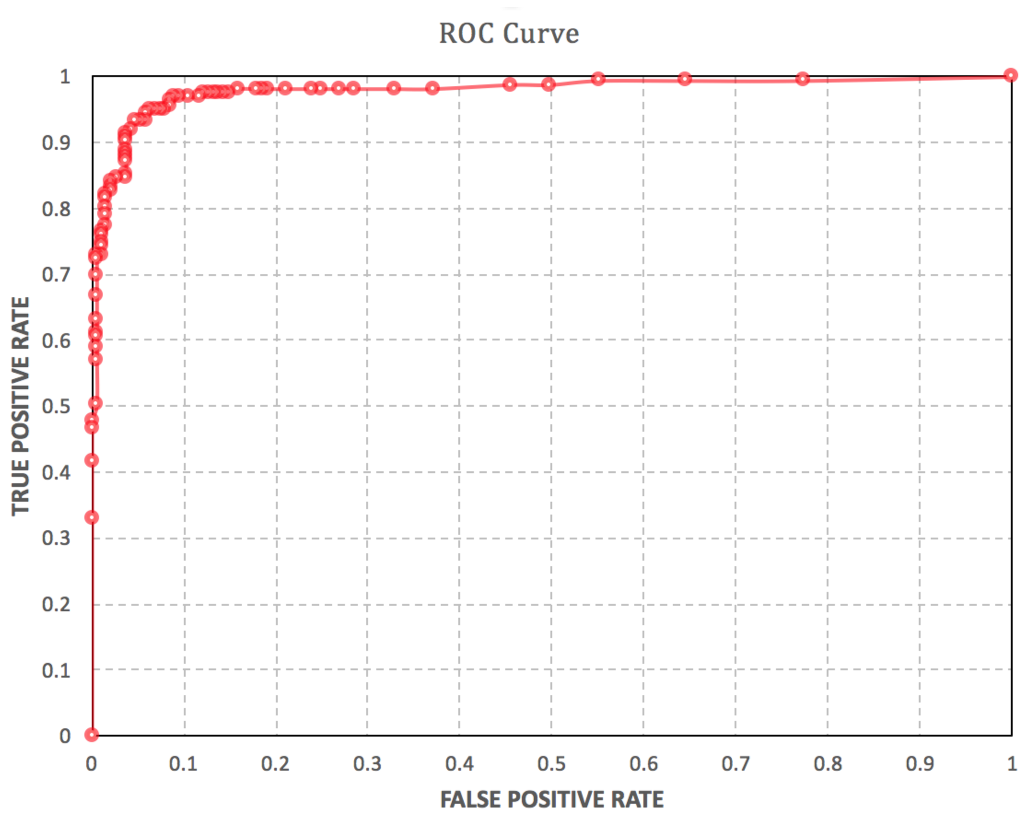


Figure 12: ROC curve for Randomforest classifier

websites.

Table 9: Dataset distribution for performance

	Phishing website	Benign website
Number of URLs	100	100
Total URLs	200	

Table 10: Performance distribution

	Phishing website	Benign website
Average time for decision	2.70 Sec	2.41 Sec
Average time for decision for all the websites	2.55 Sec	

Performance varies sometimes. There are lots of factors that can affect the performance. Speed of the Internet, size of the source code of the web page could be the major factors. It also depends on the system because string matching algorithm does lot of computation sometimes.

CHAPTER 6

Conclusion and future work

6.1 Summary

Our aim was to build the phishing filter in a web browser by taking the reference of Jo et al. research paper [2]. We built a Firefox add-on that gives minimal overhead in browsing and detects fresh phishing websites, overcoming the problem of window of vulnerability. Our test results show that our filter is able to detect the many kind of phishing websites.

We tested our filter with different classifiers available in Weka [21]. As per our test results, we were able to achieve up to 94.3% accuracy. Our experimental results show that our filter helps to overcomes the problem of the window of vulnerability. Performance is another major factor is to be considered. Results showed that average time to detect the phishing website is 5 seconds on our hardware and environment.

6.2 Future work

Though our filter achieves up to 94.3% accuracy, there could be many things that can be improved. Performance can be increased by detecting other attributes that differ from the original websites. We can include those attributes in our data set. We can also combine the several approaches that were proposed earlier [3, 4]. To provide a double layer of protection, we can also combine the apporach of "Anti-Phishing in Offense and Defense" proposed by Yue and Wang [4]. We believe that performance can also be increased by several techniques like caching the values of whois queries [3] and load balancing the servers. Our filter currently detects only phishing websites developed in "English". Extending the ideas of philter to other languages could be another interesting direction of this research.

LIST OF REFERENCES

[1] Jerry, Felix and Chris, Hauck. System Security: A Hacker's Perspective. *1987 Interex Proceedings.* September. 1987. 8. 6.

[2] Jo, Insoon, Eunjin Jung, and Heon Young Yeom. Interactive website filter for safe web browsing. *Journal of Information Science and Engineering* 29.1 (2013): 115-131.
`http://www.cs.usfca.edu/~ejung/pubs/JISE_final.pdf`

[3] Geng, Guang-Gang, et al. Favicon-a clue to phishing sites detection. *eCrime Researchers Summit (eCRS), 2013* . IEEE, 2013.

[4] Yue, Chuan, and Haining Wang. Anti-phishing in offense and defense. *Computer Security Applications Conference, 2008 ACSAC 2008. Annual.* IEEE, 2008.

[5] Wenyin, Liu, et al. Phishing Web page detection. *Document Analysis and Recognition, 2005. Proceedings. Eighth International Conference on.* IEEE, 2005.

[6] Zhang, Yue, Jason I. Hong, and Lorrie F. Cranor. Cantina: a content-based approach to detecting phishing web sites. *Proceedings of the 16th international conference on World Wide Web.* ACM, 2007.

[7] Huh, Jun Ho, and Hyoungshick Kim. Phishing detection with popular search engines: Simple and effective. *Foundations and Practice of Security.* Springer Berlin Heidelberg, 2012. 194-207.

[8] Ukkonen, Esko. Approximate string-matching with q-grams and maximal matches. *Theoretical computer science* 92.1 (1992): 191-211.

[9] PhishTank | Join the Fight against Phishing. Web. 28 Apr, 2015.
`http://www.phishtank.com/`

[10] Alexa Top 500 Global Sites. Alexa. Web. 6 Aug, 2015.
`http://www.alexa.com/topsites`

[11] Sheng, S., Wardman, B., Warner, G., Cranor, L. F., Hong, J., & Zhang, C. An empirical analysis of phishing blacklists. (2009).

[12] Phishing. Wikipedia. Wikimedia Foundation. Web. 18 Sept, 2015.
`https://en.wikipedia.org/wiki/Phishing`

[13] Phishing Activity Trends Report. APWG. Web. 9 Aug, 2015.
`http://docs.apwg.org/reports/apwg_trends_report_q4_2014.pdf`

[14] JSON API Whois Service. Whois API LLC. Web. 24 May 2015.
`https://www.whoisxmlapi.com/whois-api-doc.php`

[15] The Zettabyte Era: Trends and Analysis. Cisco Visual Networking Index (VNI) pag. Cisco, 27 May 2015. Web. 12 Sept. 2015.

[16] Unifying the Global Response to Cybercrime, APWG Unifying the Global Response to Cybercrime, APWG. AntiPhishing Group. Web. 29 Oct. 2015.
`http://antiphishing.org`

[17] Hall, Patrick AV, and Geoff R. Dowling. Approximate string matching. *ACM computing surveys (CSUR)* 12.4 (1980): 381-402.

[18] Approximate String Matching. Wikipedia. Wikimedia Foundation, Web. 30 Oct. 2015.
`https://en.wikipedia.org/wiki/Approximate_string_matching`

[19] Levenshtein, Vladimir I. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet physics doklady.* Vol. 10. No. 8. 1966.

[20] SaiKrishna, Vidya, Akhtar Rasool, and Nilay Khare. String Matching and its Applications in Diversified Fields. *International Journal of Computer Science Issues* 9.1 (2012): 219-226.

[21] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, Ian H. Witten. The WEKA Data Mining Software: An Update (2009). SIGKDD Explorations, Volume 11, Issue 1.

[22] Zdravko Markov, and Ingrid Russell. An Introduction to the WEKA Data Mining System. (2004): 1-26. Web. 20 Jan, 2015
`http://www.cs.ccsu.edu/~markov/weka-tutorial.pdf`

[23] Xiang, Guang, and Jason I. Hong. A hybrid phish detection approach by identity discovery and keywords retrieval. *Proceedings of the 18th international conference on World wide web.* ACM, 2009.

[24] 40% of the Most Visited Websites Have No Favicon for Mobile Platforms. Favicons Blog. Web. 09 Nov, 2015.
`https://realfavicongenerator.net/blog/the-most-visited-websites-have-no-favicon-for-mobile-platforms`

[25] Navarro, Gonzalo. A guided tour to approximate string matching. *ACM computing surveys (CSUR)* 33.1 (2001): 31-88.

[26] Karp, Richard M., and Michael O. Rabin. Efficient randomized pattern-matching algorithms. *IBM Journal of Research and Development* 31.2 (1987): 249-260.

[27] Boyer, Robert S., and J. Strother Moore. A fast string searching algorithm. *Communications of the ACM* 20.10 (1977): 762-772.

[28] Marc GOU. Approximate String Matching. *String Searching Algorithms Lecture Notes Series on Computing* (1994): 111-89. Web. 2 Aug, 2015.
`http://www.student.montefiore.ulg.ac.be/~s091678/files/OHJ2906_Project.pdf`

[29] NICK MCCREA. An Introduction to Machine Learning Theory and Its Applications: A Visual Tutorial with Examples. Toptal Engineering Blog. Toptal. Web. 2 Sept, 2015.
`http://www.toptal.com/machine-learning/machine-learning-theory-an-introductory-primer`

[30] Jetpack Manager. Npm: Jpm. Web. 13 Oct, 2015.
`https://www.npmjs.com/package/jpm`

[31] Moizilla add-on tutorials. Mozilla Developer Network. Web. 4 Jan, 2015.
`https://developer.mozilla.org/en-US/Add-ons/SDK/Tutorials`

[32] Mozilla add-on. Mozilla Developer Network. Web. 6 Jan, 2015.
`https://developer.mozilla.org/en-US/Add-ons`

[33] Spring Framework. Web. 13 Nov, 2015.
`http://projects.spring.io/spring-framework`

[34] Machine Learning for Email Spam Filtering. Csewiki. Web. 9 Oct, 2015.
`http://cse-wiki.unl.edu/wiki/index.php/Machine_Learning_for_Email_Spam_Filtering`

[35] Lim, Tjen-Sien, Wei-Yin Loh, and Yu-Shan Shih. An empirical comparison of decision trees and other classification methods. (1998).

[36] Gehrke, Johannes, Raghu Ramakrishnan, and Venkatesh Ganti. RainForest-A framework for fast decision tree construction of large datasets. *VLDB.* Vol. 98. 1998.

[37] WEKA Data Sets. Web. 11 Oct, 2015.
`http://storm.cis.fordham.edu/~gweiss/data-mining/weka-data/weather.arff`

[38] The 10 Most Popular Online Payment Solutions. Search Engine Journal. Web. 14 Nov, 2015.
`http://www.searchenginejournal.com/the-10-most-popular-online-payment-solutions`

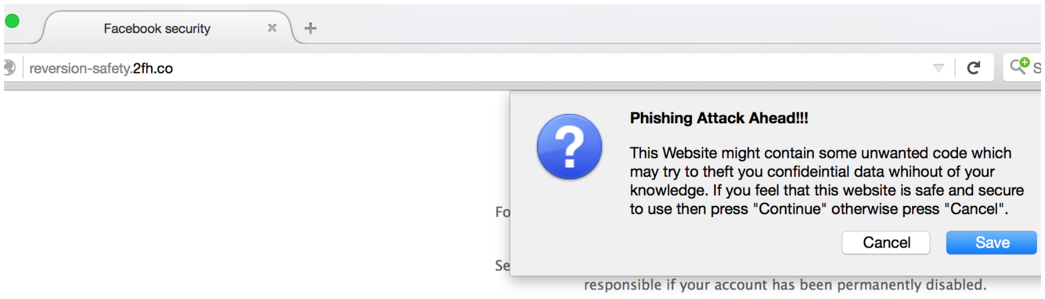
[39] TechSpective. Phishing Attacks Cost Companies $3.77 Million per Year. TechSpective, 13 Oct. 2015. Web. 14 Nov. 2015.
`https://techspective.net/2015/10/13/phishing-attacks-cost-companies-3-77-million-per-year`

[40] Number of Internet Users. Internetlivestats. Web . 11 Nov, 2015.
`http://www.internetlivestats.com/internet-users/#trend`

[41] Paypal Phishing Softication Growing. OpenDNS Security Labs. Web. 9 Nov. 2015.
`https://labs.opendns.com/2015/02/11/paypal-phishing-sophistication-growing/`

APPENDIX A

Screen Shots of a Phishing filter



Phishing Attack Ahead!!!

This Website might contain some unwanted code which may try to theft you confideintial data whihout of your knowledge. If you feel that this website is safe and secure to use then press "Continue" otherwise press "Cancel".

responsible if your account has been permanently disabled.

Figure A.13: Warning by phishing filter to users

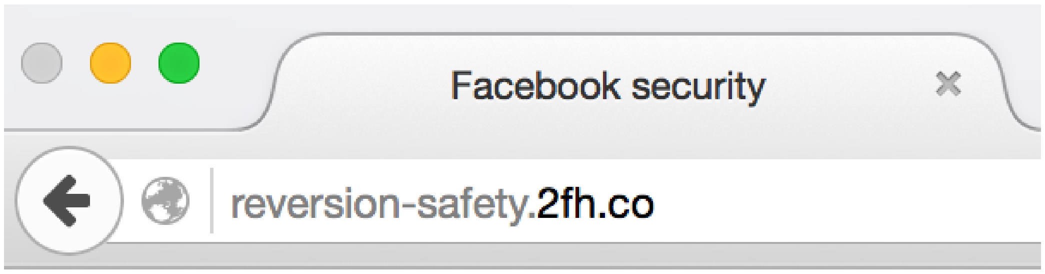


Figure A.14: Spoofed URL

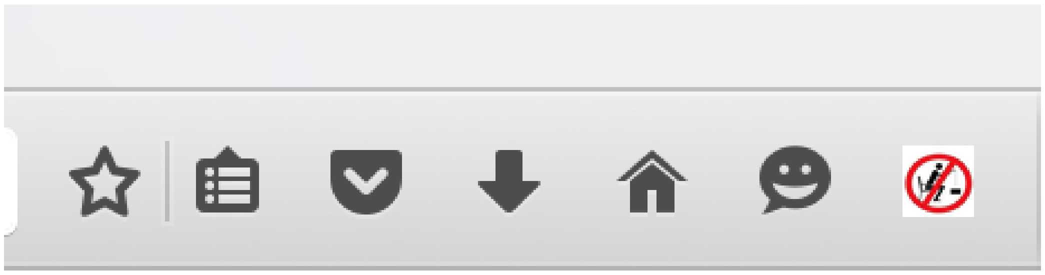
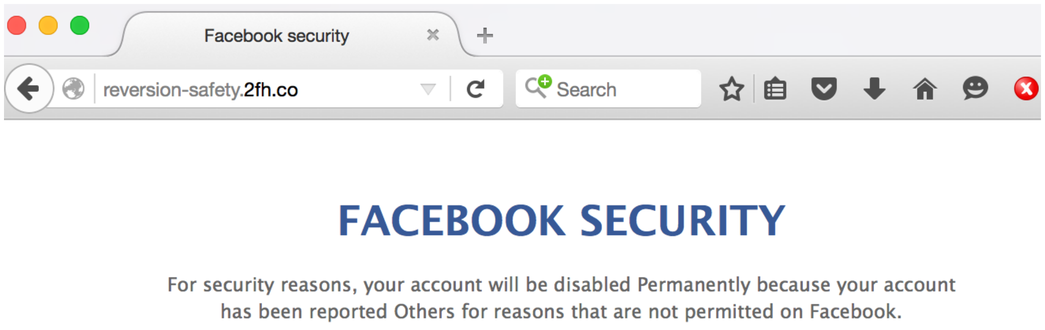


Figure A.15: Phishing filter in Firefox



FACEBOOK SECURITY

For security reasons, your account will be disabled Permanently because your account has been reported Others for reasons that are not permitted on Facebook.

Figure A.16: Icon becomes red when user ignores the warning

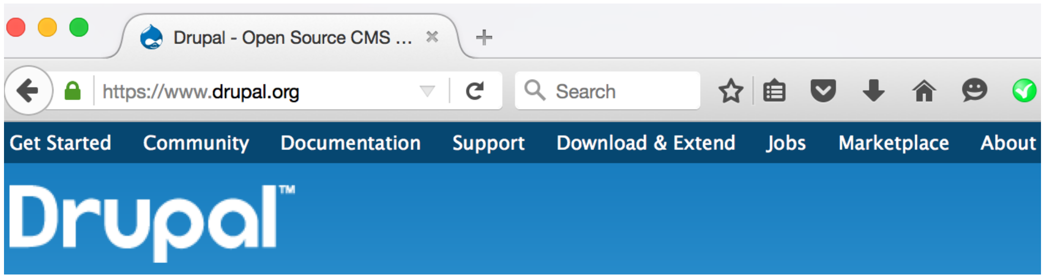


Figure A.17: Icon turns into green without warning when benign website is visited

APPENDIX B

Firefox add-on

B.1 Firefox add-on installation

Let us start with an installation of all the required tools for the development of Firefox add-ons. Previously they were using "cfx" tool to install the SDK. Now they replaced it with the "jpm SDK" [30]. We are going to install "jpm SDK". We are assuming that "Node.js" and "npm" package manager is installed on your system. If it is not then you have to install both the software first on your system.

```
sudo npm install jpm --global
npm
```

Listing B.1: jpm installation

Following are the improtant commands which are used most frequently during the development:

- **jpm init** : creates the basic folder structure and required files to start with simple add-on.
- **jpm run** : runs the firefox instance with installed add-on
- **jpm xpi** : packages the whole add-on in one file which is installation file for firefox

B.2 Playing with add-ons

Mozilla has serveral tutorials available [31]. This add-on is very simple. Just a bunch of lines and you are done. This add-on don not let you open a new tab.

Whenever you open a new tab, it will immediately close it. Let us call it a "New tab killer".

In the add-one, we add a function that listens for a new tab. As soon as the user open a new tab, the function is called. We pass the object of tab as a argument in the function. We close the tab in the function using passed object of tab.

```
// importing the tab sdk
var tabs = require("sdk/tabs");

// registering the event whenever new tab
// will be opened
tabs.on('open', function(tab){
  tab.on('ready', function(tab){
    // closes the tab as soon as the
    // user will try to open the tab.
    tab.close();
  });
});
```

Listing B.2: Tab Killer

B.3 shows the code for an add-on that does not allow the user to open same tab two times. Same tabs are defined as two tabs with the same URL. An add-on listens for the new tab when a user opens it. It will iterate through all the open tabs in the browser. The new tab is immediately closed, if it's URL is matched with any previously opened tab.

```
// import tabs package and register the function
// when tab's content will be laoded
require("sdk/tabs").on("load", logURL);

```

```
/**
 * function to remove duplicate tabs
 * called when new url is loaded in the tab
 */
function logURL(tab) {
  // url which is requested by user
  var url = tab.url;
  var tabs = require("sdk/tabs");

  // flag because loop will be iterate
  // for current requeseted url as well
  var flag = 0;
  for (let list_tab of tabs) {
    if (list_tab.url==url) {
      // if flag is not set then
      // set it to 1
      if (flag==0) {
        flag = 1;
      }
      // if flag is set and still duplicate url
      // is detected then close the tab
      else {
        tab.close();
        break;
      }
    }
  }
}
```

Listing B.3: Duplicate Tab killer