Volume 1 No. 6, October 2011 ISSN-2223-4985
International Journal of Information and Communication Technology Research

http://www.esjournals.org

Phishing Websites Detection based on Phishing Characteristics in the Webpage Source Code

Mona Ghotaish Alkhozae, Omar Abdullah Batarfi
Department of Computer Sciences, FCIT
King Abdulaziz University, Jeddah, KSA

ABSTRACT

World Wide Web Consortium (W3C) is the international standards organization for the World Wide Web (www). It develops standards, specifications and recommendations to enhance the interoperability and maximize consensus about the content of the web and define major parts of what makes the World Wide Web work. Phishing is a type of Internet scams that seeks to get a user's credentials by fraud websites, such as passwords, credit card numbers, bank account details and other sensitive information. There are some characteristics in webpage source code that distinguish phishing websites from legitimate websites and violate the w3c standards, so we can detect the phishing attacks by check the webpage and search for these characteristics in the source code file if it exists or not.
In this paper, we propose a phishing detection approach based on checking the webpage source code, we extract some phishing characteristics out of the W3C standards to evaluate the security of the websites, and check each character in the webpage source code, if we find a phishing character, we will decrease from the initial secure weight. Finally we calculate the security percentage based on the final weight, the high percentage indicates secure website and others indicates the website is most likely to be a phishing website. We check two webpage source codes for legitimate and phishing websites and compare the security percentages between them, we find the phishing website is less security percentage than the legitimate website; our approach can detect the phishing website based on checking phishing characteristics in the webpage source code.

Keywords: *phishing, phishing characteristics, w3c standards, webpage source code, secure website weigh*.

1. INTRODUCTION

Phishing is a fraudulent attempt to gain personal information from victims such as bank information, credit card information, social security, employment details, and online shopping account passwords and so on. Phishing attacks use fraudulent e-mails or websites designed to fool users into divulging personal financial data by stealing the trusted brands of well-known banks, e-commerce and credit card companies [8]. People regularly trust about any information they receive through email or website and phishers use injection attacks to hide his website by email or by URL redirection [1]. Phishing websites are like legitimate websites, even to the point of using the graphics and links straight of the legitimate website.

While phishing tricks are continuously growing, one common trick is to have a login screen in a popup window, which allows them to copy the legitimate website exactly [8]. Phishers send an email contain a hyperlink to open a new window or popup windows while browsing the web that claims to be legitimate website. The popup window may ask to update, validate or confirm account information and it is like official organizations websites. One of the W3C standards is avoid using popup windows or using a hyperlink to open in a new window in the webpage [9]. So, we conclude that phishers use some tricks to fool the user and tempting them, some of these are external links for images or logos, suspicious URLs, external domains, email, iframe, suspicious script, and popup window [6].

Phishing detection can be roughly classified into two categories: List-Based and Heuristic-Based [2]. List-based anti-phishing approaches are widely used today. Classifying a website as phishing or trusted is a simple database lookup. List-based approaches can be broken down into blacklist and whitelist.

Blacklists hold URLs that refer to websites that are considered phishing. Whenever a browser loads a page, it queries the blacklist to determine whether the currently visited URL is on this list. If so, appropriate countermeasures can be taken. Otherwise, the page is considered legitimate. The blacklist can be stored locally at the client or hosted at a central server. Whitelist is a list of trusted websites. The basic idea is that the user builds a list of trusted websites that he/she accesses on a regular basis. If the user attempts to navigate to a website that is not in the trusted list, he/she is either blocked from the website or prompted to add the website to the trusted list.

Heuristic-based approaches check one or more characteristics of a website to detect phishing rather than look in a list. These characteristics can be the uniform resource locater (URL), the hypertext markup language (HTML) code, or the page content itself. Most of the heuristics were targeted at the HTML source code while two considered the content of the URL.

W3C recommends standards for web development; the following are overview of some web languages and their standards [4]:

- Hypertext Markup Language (HTML) is used to structure text documents and put a hypertext links

between documents on the web .The text coding consists of commands in angle brackets <> that affect the display of elements and can be interpreted by an internet browser. For example, browsers display the italic element (…) as italic text.

- Extensible Markup Language (XML) is more flexible than HTML, because we can add our format and elements and defining data elements on a webpage, HTML defines how elements are displayed but XML defines what those elements contain. For example :

```
<person>
    <name> Mona Alkhozae </name>
    <city> Jeddah </city>
    <university> KAU </university>
    <department> CS </department>
</person>
```

In XML:

  - All document types are declared via the correct DOCTYPE declaration.
  - The structure of a conforming document contains the DOCTYPE declaration, an html element with the XHTML namespace declared, a head element with the title element, and a body element.
  - All elements and attribute names must be written in lower case and attribute values are quoted.
  - All non-empty elements (e.g. p, li) are terminated with a closing tag.
  - All empty elements (e.g. br, hr, img) are properly terminated with a trailing slash (
).

- Cascading Style Sheets (CSS) is for webpages presentation and element appearance, including colors, fonts, etc., CSS separate content from presentation and the W3C recommended that layout in HTML be phased out and replaced by stylesheets to control the webpage layout and presentation.

In this paper, we show some phishing detection approaches, objects of W3C and common properties of phishing attacks, and we propose a phishing detection approach based on checking the phishing characteristics in the webpage source code that extracted out of the W3C standards and detect if there a phishing attacks based on the security percentage.

This paper is organized as follows. Section 2 introduces literature review about relevant phishing website detection and in section 3 we introduce W3C webpage objects and properties of phishing attack, followed by an overview of our methodology in section 4. The testing results are reported in section 5. Section 6 concludes our work and future work in this paper.

2. RELATED WORK:

Phishing website is a huge effect on the financial and online commerce, detecting and preventing this attack is an important step towards protecting against website phishing attacks, there are several approaches to detect these attacks. In this section, we review existing anti phishing solutions and list of the related works.

One approach is an intelligent Phishing Website Detection System using Fuzzy Techniques [5]. It is based on fuzzy logic and produces six criteria's of website phishing attack. There are many characteristics and factors that can distinguish the original legitimate website from the forged faked phishing website like spelling errors, long URL address and abnormal DNS record. Website phishing detection rate is performed based on six criteria and there are different numbers of components for each criterion, the criteria are:

1- URL & Domain Identity.

a) Using the IP address.
b) Abnormal request URL.
c) Abnormal URL of anchor.
d) Abnormal DNS record.
e) Abnormal URL.

2- Security & Encryption.
a) Using SSL certificate.
b) Certification authority.
c) Abnormal cookie.
d) Distinguished Names Certificate (DNC).

3- Source Code & Java script.
a) Redirect pages.
b) Straddling attack.
c) Pharming attack.
d) Using onMouseOver to hide the Link.
e) Server Form Handler (SFH).

4- Page Style & Contents.
a) Spelling errors.
b) Copying website.
c) Using forms with "Submit" button.
d) Using Popups windows.
e) Disabling right click.

5- Web Address Bar.
a) Long URL address.
b) Replacing similar characters for URL.
c) Adding a prefix or suffix.
d) Using @ symbol to confuse.
e) Using hexadecimal character codes.

6- Social Human Factor.
a) Much emphasis on security and response.
b) Public generic salutation.
c) Buying Time to Access Accounts.

The rule base has input parameters (criterion) and one output that contain all the "IF-THEN" rules of the system. The output for each criterion is one of the following: Genuine, Doubtful or Fraud. The output of final website phishing is one of the final output (Very Legitimate, Legitimate, Suspicious, Phishy or Very Phishy) which representing final phishing website rates.

Second approach is a client-side defense against web-based identity theft [6]. It proposes a framework for client-side defense: a browser plug-in called SpoofGuard that examines webpages and warns the user when requests for data may be part of a spoof attack, it computes a spoof index (a measure of the likelihood that a specific page is part of a spoof attack), and warns the user if the index exceeds a level selected by the user. SpoofGuard uses a combination of page evaluation and examination of outgoing post data to compute a spoof index.

When a user enters a username and password on a spoof website that contains some combination of suspicious URL, misleading domain name, images from an honest website, and a username and password that have previously been used at an honest website, SpoofGuard will intercept the post and warn the user with a popup that foils the attack. The paper describes common properties of ten spoof websites recently found, they are Logos, Suspicious URLs, User input, Short lived, Copies, Sloppiness or lack of familiarity with English and HTTPS. The browser plug-in applies tests to all downloaded pages and combines the results using a scoring mechanism. The total spoof index of a page determines whether the plug-in alerts the user and determines the severity and type of alert. Since popup warnings are intrusive and annoying, it attempts to warn the user through a passive toolbar indicator in most situations.

In order to apply image and URL check, the SpoofGuard plug-in is supplied with a fixed database of images and their associated domains. When the browser downloads a login page all images on the page are compared to images in the SpoofGuard database. The spoof-score for the page is increased if a match is found but the page's domain is not a valid domain for the image. The browser history file and additional history stored by SpoofGuard are used to evaluate the referring page. When a user fills in form data, SpoofGuard intercepts and checks the HTML post data, allowing the actual post to proceed only if the spoof index is below the user specific threshold for posts.

Third approach is Anomaly Based Web Phishing Page Detection [7]. It examines the anomalies in webpages, in particular, the discrepancy between a website's identity and its structural features and HTTP transactions. A structured webpage is composed of W3C DOM objects. Among them, it lists five categories, based on their relevance to the web identity. They are Keyword/Description (KD), Request URL (RURL), URL of Anchor (AURL), Server Form Handler (SFH) and Main Body (MB).These categories are the main sources which the identity and features are derived from and lists the characteristics of phishing like Abnormal URL, Abnormal DNS record, Abnormal Anchors, Abnormal Server Form Handler, Abnormal Request URL, Abnormal cookie and abnormal certificate in SSL.

It extracts the related web objects from a webpage and converts them into a feature vector based on the characteristics of phishing analysis. The page classifier takes the feature vector as input and determines whether the page is bogus or not. The proposed phishing detector consists of two components Identity Extractor and Page Classifier.

Identity Extractor uniquely identifies the website's ownership; the identity is an abbreviation of the organization's full name and/or a unique string appearing in its domain name.

Page classifier refers to these objects/properties as structural features. One source of structural features is those identity related W3C DOM objects in a webpage, e.g. URI domain of an anchor. Another source of structural features is HTTP transactions. Page classifier employs Support Vector Machine (SVM), a well-known algorithm for classification. It outputs a label 1 which indicating a phishing page or a label -1 which indicating an authentic one.

To facilitate SVM based classification, they quantify those features into vectors. The output from the execution of the webpage identity extractor is a character strings from extracted identity words. The feature vector initialization of webpage are URL address, DNS record ,URL of anchor , request URL , server form handler , domain in cookie and certificate in SSL. Given an identity and a set of features, the task of determining the genuineness of a webpage is executed by Support Vector Machine (SVM), which is a well-known classifier and has been widely employed in pattern recognition.

From the approaches we conclude that, there are many characteristics and anomalies can be found in the webpage and we can detects any possible attacks based on these characteristics, phishers use these characteristics in phishing webpages to gain sensitive information from users.

In our approach is based on checking the phishing characteristics in the webpage source code file, we extract these characteristics out of W3C standards to evaluate the website security and make a security percentage based on the final weight to decide if the webpage secure or not.

3. PHISHING ATTACKS CHARACTERISTICS

A. World Wide Web Consortium (W3C) objects:

A structured webpage is composed of W3C objects, some of these objects are [7]:

1) Request URL (RURL): External objects (such as images, external scripts, CSS) in a webpage are loaded from other URLs. For a normal corporate website, a large percent of those URLs are in its own domain. For instance,

<img src="https://home.peoplepc.com/i/60/common/ppco logo.gif"> in http://www .peoplepc.com.

2) URL of Anchor (AURL): A high portion of anchors in a legitimate webpage point to the same domain as the page itself. One example is <a href="http://www.ebay.com/"> in http://www.ebay.com. Webpage names must be meaningful to the visitor .Webpage names must be short, in all lower case, contain no spaces, use hyphens, not underscores between words.

3) Server Form Handler (SFH): For security and management reasons, most finance/e-business web portals require usernames and passwords. Therefore, those pages usually contain a server form handler. For example, <form action="/inetSearch/index.jsp" method="post" target=" top"> in http://www.chase.com. For phishing websites, the SFH usually is void or refers to a different domain.

B. Common properties of Phishing attacks:

The following lines represent number of properties of phishing attacks in the websites, they are [6]:

1) Logos: The Phishing website uses logos found on the legitimate website to mimic its appearance. So phishers can load it from the legitimate website domain to their phishing websites (external domain).

2) Suspicious URLs: Phishing websites are located on servers that have no relation with the legitimate website. The phishing website's URL may contain the legitimate website's URL as a substring (http: //www.ebaymode.com), or may be similar to the legitimate URL (http://www.paypa1.com) in which the letter 'L' in PayPal is substituted with number '1'. IP addresses are sometimes used to mask the host name (http://25255255255/top.htm). Others use @ marks to make host names difficult to understand (http://ebay.com:top@255255255255/top.html) or contain suspicious usernames in their URLs (http://middleman/http://www.ebay.com).

3) User input: Phishing websites typically contain pages for the user to enter sensitive information, such as account number, password and so on.

4) Short lived: Most phishing websites are available for only a few hours or days – just enough time for the attacker to defraud a high enough number of users.

5) Copies: Attackers copy HTML from the legitimate websites and make minimal changes.

6) Sloppiness or lack of familiarity with English: Many Phishing pages have misspellings, grammatical errors, and inconsistencies.

4. PHISHING WEBSITES DETECTION METHODOLOGY

A. The phishing characteristics out of the W3C standards:

Phishers use some tricks and to fool users and tempting them, so our approach is to check for these tricks and factors in the webpage source code and calculate the security percentage based on these factors to classify the webpage if it is secure or not, they are:

1) Https: It is the secured protocol which used to tell us that this website is secured but it should be in "URL" of the website not in the body source of the webpage because phishers used Https inside their source code file to tell us that this images or this links is secured but it is not. The normal page should be like this . But there is some phishers use the SSL certificate in the source code like this . They use https to make us think its secured website buts it is not. Many similar phishing attacks in which phishing websites use a certificate that can be expected to trigger a browser warning.

2) Images: All images in the website including website logo should load from the same URL of the website not from another website, so all links should be internal links not external links. Therefore, we check the links to detect any external links inside the source code like this : <img src=https: //www.Phishers.com/logo.jpg>" it is a phishing character.

3) Suspicious URLs: Most of the phishers use an IP address instead of using the actual domain name. Others use @ marks to ambiguous their host names.

4) Domain: It is the external domains mean: if we logged to website which its name is www.paypal.com and we found there is some URLS of links in the source code like this " www.pay-pal.com" which it is not the source URL so it means that this website try to hack our information .Phisher use forward domain also called domain redirection, it is a technique on the World Wide Web for making a webpage available under many URLs.

5) Email: There is a function on PHP called mail or email and it take our information which we enter in the forms like "MasterCard number, etc. "and send them when we press the pay button throw e-mail to the phishers e-mail. Phisher can insert PHP code inside Html code and use this function to send our information.

6) iframe: It is HTML tag code and used to embedding another webpage into current webpage. It creates a frame or window on a webpage so that another page can load inside this frame. Phishers use the iframe and make it invisible i.e. without frame borders, when the user goes to website, he/she cannot know that there is another page is also loading in the iframe window. It is a big problem which all people do not know it, it is like small website open in current webpage for example: we can open

www.google.com in my page www.mona.com by using iframe so when the people enter our website they will see the secured website is opened but it is not in the page it open throw iframe . Example: http://www.phisher.com/index.php?search="><iframe src=http://google.com ></iframe> // Replace http://google.com by the phishing page.

7) *Script*: It is PHP files and there is some of phisher used scripts to send personal information or PC information to them, and some scripts send viruses or load from external websites. Scripts tag use to put any external file in the page like jquery or CSS and if it is with start and end tag, it is legal because this is the correct and standard script tag. Example: <script type="text/javascript" src="includes/jscripts/jquery.min.js"></script>, it is now load file to make the page appearance good. When there are tags like this <script> and between them any codes (not links) it is suspicious tags because this script code is javascript or any other languages which may be used to send personal information or PC information to phishers. So if we find <script> tags and there end tags </script> it is a legal tags, otherwise it is a phishing character.

8) Popup window: Phishers use popup windows to gain personal information. Often, these popups may ask to update, validate or confirm account information and it is likes official organizations websites. If the user enters his information in the popup windows, the phisher then steals this private information. There is two ways of popup window ,one of them is used to confirm or to tell us something and this window have a special way to write it in html or JavaScript like this < … onClick="window.open('mona.html')"> and it is by html and legal window . The other way is illegal popup window because it is a javaScript file used in like this: "Open Popup" onClick="javascript:popUp('mona.html')"> . It is illegal because it is open a new page from another website as registration page or ask the user information. It is a new full page and it open automatic when the user open a page or click on any link so it is illegal popup window.

B. System design and implementation:

i. System overview:

Detect the phishing websites by checking the webpage source code, we extract some phishing characteristics out of the W3C standards to evaluate the security of the websites, and check each character in the webpage source code, if we find a phishing character, we will decrease from the initial secure weight. Finally we calculate the security percentage based on the final weight, the high percentage indicates secure website and others indicates the website is most likely to be a phishing website.

The environment built by using Microsoft Visual Studio 2010 and use C# programming language. Figure 1 shows the phishing detection layout.

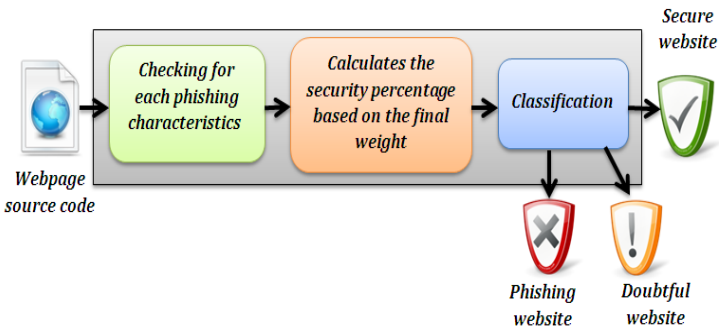

Figure 1: phishing detection layout

ii. System description and work flow:

1- First:

- Browse button opens an open file dialog to choose the source code file to check the webpage security; figure 2 is the program main window.
- Starts a file stream that opens the file in read mode.

Figure 2: Phishing detection program main window

2- Second:

- Checks the webpage security throw the check button.
- Reads every line in the source code individually.

a) Initializing the counter by a secure website weight.
b) Checks for each phishing characteristics in the webpage source code and we have 8 phishing character. They are: different domain, external link for images, suspicious URLs, domain tag, iframe, email, suspicious script and popup window.
c) Phishing characteristics classification :

Table 1: Phishing characteristics classification

Phishing characteristics	Phishing characteristics risk
Https	Medium
Images	Low
Suspicious URLs	High
Domain	Medium

Email	High
Iframe	Low
Script	High
Popup window	Low

d) If the program finds a phishing character, it will decrease from the counter based on the phishing character risk.

e) The program will check in this source code for this things :

1. First it will search for all images in this website so if there any images work from another website or have links from another place it will consider it a phishing character [7] , so all images should be in the website folder like this <img src="Logo.PNG" border="0" width="1349" height="706">.
2. The program will check the login or submit button and should make the button action on the website like: login.php. If the button action link to any IP likes this: 103.838.39.0/login.php or email or script it will consider a phishing character.
3. Also check for iframe, domain, script tags and popup window if it found it considered a phishing character.
4. If there are more than one phishing characteristic of the same type such as more than one popup window, it will be considered as one.

f) After the program considers all phishing characteristics, it makes a security percentage based on the final weight to tell us if this website will be secure or not.

- If the website takes 80 % of 100 % or above, the website is secured.
- If the website take between (79%- 50 %), the website is doubtful.
- If the website takes under 50%, the website is phishing.

3- Finally:

- Shows the window that indicates if this webpage is secured or doubtful or Phishing based on the above security percentage.
- Shows the percentage that was previously calculated in the window.
- The program workflow as follows in figure 3:

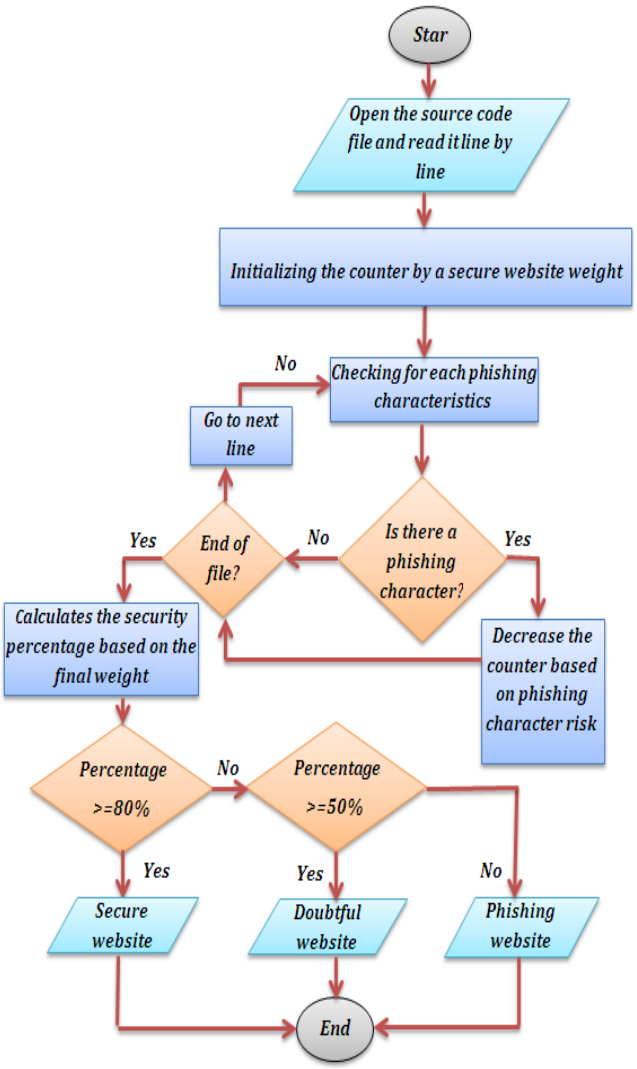

Figure 3: Phishing detection program flow chart

5. TESTING AND RESULTS

i. Results screenshots:

- ***The First window:*** The webpage is secured when the security percentage is 80% or higher, figure 4 is an example of testing results.

Figure 4: The program checking result window

(When the security percentage 80% or more)

- ***The Second window:*** The webpage is doubtful when the security percentage is less than 80% and higher than 50%, figure 5 is an example of testing results.

Figure 5: The program checking result window

(When the security percentage less than 80% and higher than 50%)

- ***The Third window:*** The webpage is phishing when the security percentage is less than 50%, figure 6 is an example of testing results.

Figure 6: The program checking result window

(When the security percentage less than 50%)

ii. Test cases :

- I build two source code files for the same page and add phishing characteristics to these files. The first source code file in figure 7 has a popup window and the second in figure 8 has three phishing characteristics, they are: an email in the form button action, external link for the image, domain tag.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<title>Secure Login</title>
<meta name="keywords" content="Paypal,Secure,Money,transfer">
<script type="text/javascript">
function popUp(URL) {
day = new Date();
id = day.getTime();
eval("page" + id + " = window.open(URL, '" + id + "', 'toolbar=0,scrollbars=0,location=0,statusbar=1,menubar=0,resiza
}
function getCookie(c_name)
{
var i,x,y,ARRcookies=document.cookie.split(";");
for (i=0;i<ARRcookies.length;i++)
 {
  x=ARRcookies[i].substr(0,ARRcookies[i].indexOf("="));
  y=ARRcookies[i].substr(ARRcookies[i].indexOf("=")+1);
  x=x.replace(/^\s+|\s+$/g,"");
  if (x==c_name)
    {
    return unescape(y);
    } } }
function setCookie(c_name,value,exdays) {
var exdate=new Date();
exdate.setDate(exdate.getDate() + exdays);
var c_value=escape(value) + ((exdays==null) ? "" : "; expires="+exdate.toUTCString());
document.cookie=c_name + "=" + c_value; }
function checkCookie()
{
var username=getCookie("username");
if (username!=null && username!="")
 {
 alert("Welcome again " + username);
 }
else
 {
 username=prompt("Please enter your name:","");
 if (username!=null && username!="")
   {
   setCookie("username",username,365);
   }
 }
}
</script>
<style type="text/css">
 p.c2 {text-align: center}
 div.c1 {text-align: center}
</style>
</head>
<body>
<div class="c1"><img src="logo.png"></div>
<form name="ThisForm" id="ThisForm">
<p class="c2"><br>
Card Type: <select name="CardType">
<option value="MasterCard">MasterCard</option>
<option value="VisaCard">Visa</option>
</select></p>
<p class="c2">Card Number: <input name="CardNumber" size="22" maxlength="19"></p>
<p class="c2"></p>
<form>
<p class="c2">Security Number : <input type="password" name="pwd"></p>
</form>
<div class="c1"><br> Expiration Date: Month <select name="ExpMon">
<option value="1" selected>1</option>
<option value="2">2</option>
<option value="3">3</option>
<option value="4">4</option>
<option value="5">5</option>
<option value="6">6</option>
<option value="7">7</option>
<option value="8">8</option>
<option value="9">9</option>
<option value="10">10</option>
<option value="11">11</option>
<option value="12">12</option>
</select>
<div class="c1">Year <input name="ExpYear" size="4" maxlength="4">  (Range: 1997~2020)<br>
<div class="c1"><input type="button" value="Check" onclick="CheckCardNumber(this.form)"><br></div>
</div>
</div>
</form>
<BODY onLoad="javascript:popUp('http:www.paybal.com')">
</body>
</html>
```

Figure 7: First source code file

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<title>Secure Login</title>
<meta name="keywords" content="Paypal,Secure,Money,transfer"></meta>
<script type="text/javascript">
<domain>
function getCookie(c_name)
{
var i,x,y,ARRcookies=document.cookie.split(";");
for (i=0;i<ARRcookies.length;i++)
  {
  x=ARRcookies[i].substr(0,ARRcookies[i].indexOf("="));
  y=ARRcookies[i].substr(ARRcookies[i].indexOf("=")+1);
  x=x.replace(/^\s+|\s+$/g,"");
  if (x==c_name)
    {
    return unescape(y);
    } } }
function setCookie(c_name,value,exdays)
{
var exdate=new Date();
exdate.setDate(exdate.getDate() + exdays);
var c_value=escape(value) + ((exdays==null) ? "" : "; expires="+exdate.toUTCString());
document.cookie=c_name + "=" + c_value;}
function checkCookie()
{
var username=getCookie("username");
if (username!=null && username!="")
  {
  alert("Welcome again " + username);
  }
else
  {
  username=prompt("Please enter your name:","");
  if (username!=null && username!="")
    {
    setCookie("username",username,365);
    }
  }
}
</script>
<style type="text/css">
p.c2 {text-align: center}
div.c1 {text-align: center}
</style>
</head>
<body>
<div class="c1"><img src="https://www.paypal.com/logo.png" alt=logo></div>
<form name="ThisForm" id="ThisForm">
<p class="c2"><br>
Card Type: <select name="CardType">
<option value="MasterCard">MasterCard</option>
<option value="VisaCard">Visa</option>
</select></p>
<p class="c2">Card Number: <input name="CardNumber" size="22" maxlength="19"></p>
<p class="c2"></p>
<form>
<p class="c2">Security Number : <input type="password" name="pwd"></p>
</form>
<div class="c1"><br>Expiration Date: Month <select name="ExpMon">
<option value="1" selected>1</option>
<option value="2">2</option>
<option value="3">3</option>
<option value="4">4</option>
<option value="5">5</option>
<option value="6">6</option>
<option value="7">7</option>
<option value="8">8</option>
<option value="9">9</option>
<option value="10">10</option>
<option value="11">11</option>
<option value="12">12</option>
</select>
<div class="c1">Year <input name="ExpYear" size="4" maxlength="4">  (Range: 1997~2020)<br></div>
<div class="c1"><input type="button" value="Check" onclick="CheckCardNumber"email"><"hacker@xx.com"<br></d
</td>
</div>
</form>
<img src="http://www.iuiyyo.COM/LOGO.PNG>
</body>
</html>
```

Figure 8: Second source code file

1- First source code file:

It is secure website; the program found only a popup window, figure 9 shows the result window.

Figure 9: First source code file checking results.

2- Second source code file:

It is a phishing website, the program found three phishing characteristics, they are:

- The button action is an email.
- Image source is an external link.
- Domain tag.

Figure 10 shows the result window.

Figure 10: Second source code file checking results

iii. Discussing the finding: table 2 shows the phishing characteristics was found in the webpage source code files.

Table 2: Detailed results about the two source code files

phishing characteristics	File 1	File 2
1. Different domain	No	No
2. External images	No	Yes
3. Suspicious URLs	No	No
4. Domain tag	No	Yes
5. Iframe	No	No
6. Email	No	Yes
7. Suspicious Script	No	No
8. Popup window	Yes	No
Rate	90 %	34 %
Website class	Secure website	Phishing website

CONCLUSION AND FUTURE WORK

In this paper we proposed a phishing detection approach that classifies the webpage security by checking the webpage source code, we extract some phishing characteristics out of the W3C standards to evaluate the security of the websites, and checked the webpage source code, if we find a phishing character, and we will decrease from the initial secure weight. Finally we calculated the security percentage based on the final weight, the high percentage indicates secure website and others indicates the website is most likely to be a phishing website. We checked two webpage source codes for legitimate and phishing websites and compare the security percentages between them, and we found the phishing website is less security percentage than the legitimate website. In Future work we can add other checks in the program and check more source codes contains many languages in it like PHP, CSS, asp, java, Perl, etc. Also, we can develop a browser plug-in to check the webpages and informs the user if there any possible attack.

REFERENCES

[1] http://www.oit.umn.edu/safe-computing/topics/phishing-scams/index.htm

[2] M. Dunlop, S. Groat, and D. Shelly," GoldPhish: Using Images for Content-Based Phishing Analysis", in the Fifth International Conference on Internet Monitoring and Protection, 2010.

[3] http://www.w3.org/TR/xhtml1/

[4] http://www.webstandards.org/learn/faq/

[5] M. Aburrous, M.A. Hossain, F. Thabatah and K. Dahal, "Intelligent phishing website detection system using fuzzy techniques", in 3rd International Conference on Information and Communication Technologies: From Theory to Applications (ICTTA), pp. 1-6, 2008.

[6] N. Chou, R. Ledesma, Y. Teraguchi, D. Boneh, and J. C. Mitchell, "Client–side defense against web–based identity theft", In Proceedings of 11th Annual Network and Distributed System Security Symposium, 2004.

[7] Y. Pan and X. Ding, “Anomaly Based Web Phishing Page Detection", Proceedings of the 22nd Annual Computer Security Applications Conference (ACSAC'06), Computer Society, 2006.

[8] http://www.oit.umn.edu/safe-computing/topics/phishing-scams/index.htm

[9] http://www.w3.org/TR/WAI-WEBCONTENT/