2009

Techniques for detecting zero day phishing websites

Michael Blasi
Iowa State University

Techniques for detecting zero day phishing websites

by

Michael Blasi

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Co Majors: Information Assurance; Computer Engineering

Program of Study Committee:
Yong Guan, Major Professor
Doug Jacobson
Steffen Schmidt

Iowa State University

Ames, Iowa

2009

TABLE OF CONTENTS

ABSTRACT

Phishing is a means of obtaining confidential information through fraudulent web sites that appear to be legitimate. There are many phishing detection techniques available, but current practices leave much to be desired. A central problem is that web browsers rely on a black list of known phishing sites, but some phishing sites have a lifespan as short as a few hours. A faster recognition system is needed by the web browser to identify zero day phishing sites which are new phishing sites that have not yet been discovered.

This research improves upon techniques used by popular anti-phishing software and introduces a new method of detecting fraudulent web pages using cascading style sheets (CSS). Current phishing detection techniques are examined and a new detection method is implemented and evaluated against hundreds of known phishing sites.

CHAPTER 1. OVERVIEW OF PHISHING

Phishing is the electronic means of obtaining personal information by disguising oneself as a legitimate online entity. More specifically phishing is the process of creating a fraudulent email or website that appears to originate from a legitimate source. Phishers, who are authors of phishing websites, create a fraudulent website in hopes that visitors will divulge sensitive information such as account numbers, usernames, passwords, pins, social security numbers, etc.

Introduction

Typically there are three main phases to the phishing cycle. First, the phisher creates a phishing website and then goes phishing by sending out numerous emails to unsuspecting users. The phisher tries to convince the reader of the email to visit the link included in the email. When the user "bites" on the phish, the link in the email directs the user to the phishing site which appears legitimate and similar or identical to the legitimate target site. The phish is successful when the user enters confidential information on the phishing page and it is leaked to the phisher. Afterwards the phisher tries to exploit the confidential information by transferring money, opening accounts, or making purchases using the captured information. Or the phisher merely acts as a middleman and sells the information to other criminals.

The Phish

The phishing life cycle typically begins with a mass email that attempts to convince the reader to visit the included website link. This phase of phishing is much like fishing.

Instead of using a fishing lure and line to catch a fish, a phisher sends out many emails in hopes that a few readers will “bite” at the email lure by visiting the included link to the phishing website.

Typically the email looks legitimate and will include a company logo of a popular financial institution and a return address of the legitimate company. The link in the email will also appear legitimate at first glance. The phisher wants the lure to be as authentic as possible so that the victim will “bite”.

Usually the phishing email will try to convince the reader to visit the included website in order to update certain information or avoid account termination. Many techniques are used, but most try to convince the reader that urgent action is needed and prey upon emotion such as urgency, account interruption, or account termination.

The Bite

The bite occurs when the victim clicks on the link in the email and is directed to the phishing website. The phishing website typically looks identical or very similar to the legitimate site it is attempting to impersonate. It is critical that the website looks legitimate so that the user does not suspect that the page is fraudulent. Often the legitimate page is simply copied and hosted elsewhere so the phishing page contains all of the correct styles and content of the legitimate site. Logos, keywords, and even security notices are commonly found on phishing sites to convince the user that the site is legitimate. Once the user visits the phishing website and is assured that the page is legitimate because it resembles the legitimate site, then the phisher can request personal information. It is a critical step for the phisher to first build the trust so the user thinks that the page is legitimate. If there are misspellings, outdated images, or other suspicious content then the user may think twice about entering sensitive information.

The Catch

Once a user has visited the phishing page and is convinced that the page is familiar and legitimate, then the phisher requests confidential information from the user. Often there is a user login and password box that requests a username and password from the user. Sometimes a phishing page will ask for other confidential information such as account numbers, pins, social security numbers, date of birth, etc. Once the user divulges this information it is typically stored in a database on the phishing server, emailed to a phisher's email address, or sent to a chat room. After the information is submitted the user will typically receive an error message, return to the phishing login box with the impression that nothing happened, or be redirected to the legitimate website. It will appear to the user that nothing happened even though the information has been leaked. Obviously the phisher doesn't want the user to know that they have just divulged their confidential information.

The Damage

Phishers harvest confidential information and then either try to exploit it by transferring funds, making purchases, etc. or they sell the information to third party criminals to exploit [2]. Underground internet chat rooms are common meeting areas where phishers can sell confidential information to interested parties.

Prosecution

Typically phishing sites are active for only a short period of time before being discovered and shutdown. Most sites are active for as short as a few hours to as long as a few days [12]. Usually the site will be reported and confirmed phishing and then the Internet Service Provider will delete the phishing site. However it is often difficult to remove the site or prosecute the phisher if the site is hosted in a foreign country because of differing laws and jurisdiction. Often financial institutions will refund the lost money from customers because it is easier and less costly than trying to find and prosecute the criminals.

CHAPTER 2. TAXONOMY OF PHISHING

There are many techniques to lure victims to phishing websites by using creative emails and playing upon emotions of email readers. This research however isn't concerned about how the victim arrived upon the phishing page, but focuses on detecting the page as phishing once the page is loaded. Therefore this taxonomy of different phishing techniques is focused on the types and tricks used in the phishing website itself.

Introduction

While most phishing websites try to imitate the legitimate site as best as possible there are other sites that merely try to obtain as much information as possible. Some phishing sites could also be classified as malicious sites as well because they attempt to exploit browser vulnerabilities or install malicious code.

Phishing targets

Phishers naturally want to target a site that has a high reward with little risk. Most phishing attacks target financial institutions such as banks, brokers, credit card companies, etc. Obviously these high value targets have the biggest reward because money can be transferred from an account or a fraudulent credit card purchase or transfer can be made. Other high value targets include eBay and PayPal. Phishers are increasingly targeting even smaller banks such as credit unions because typically these banks don't have the resources to try and retaliate against phishing attacks. The risk is lower when targeting a smaller bank. Other less damaging targets include online email accounts and social networking sites.

This research focuses on financial institutions and includes eBay and PayPal since they are highly targeted companies for phishers.

Mimic the legitimate site

It should be obvious that the majority of phishing sites try to mimic the legitimate site as best as possible. After all the phisher is attempting to impersonate the legitimate site so that the victim is confident enough to divulge sensitive information. If the victim isn't comfortable or familiar with the site layout, then he may become suspicious of the phishing site.

Most phishing sites do a good job of appearing legitimate by copying the page layout, fonts, styles, logos, and even security information of the legitimate site. In fact many of the links on the phishing site will actually link to the legitimate site which helps the phishing site appear even more legitimate.

Many phishing sites use or copy the style sheet of the legitimate site so that all the page layout, fonts, styles, etc. match the site. Detecting phishing sites that use legitimate style sheets is the main contribution of this research and is further discussed in chapter 7.

Mimic the URL

In addition to mimicking the actual content of the websites some phishing sites try to mimic the actual URL of the phishing site. For example one could replace a W with two V's or a lowercase L with a number 1. Phishing sites often try to use a URL that mimics the legitimate URL or includes the legitimate URL in the phishing URL somewhere [5].

Update personal information

Some phishing sites don't try to imitate the legitimate site very well. In fact the phishing page may only have a logo or security seal that matches one on the legitimate site. However these pages can be very dangerous because they will request several pieces of sensitive information. One example is a phishing page that asks the user to update their information such as name, address, phone number, account number, credit card number, etc. These phishing sites can be more difficult to detect because the page doesn't mimic the legitimate page except for a logo or keyword. Mainly the page contains many input boxes for a victim to enter personal information. However some of these types of phishing pages still use the legitimate sites' style sheet so they can be detected with the style sheet detection discussed in chapter 7.

Obscure URL

Some phishing sites have been known to try and hide the actual URL of the phishing site and replace it with the legitimate URL [19]. Exploiting browser vulnerabilities or using Javascript to overwrite the address bar with something different is an advanced phishing technique to hide the actual phishing URL. When the user looks at the browser address bar he will see the legitimate URL even though the page is a phishing site.

DNS poisoning

Another advanced phishing technique is to poison the DNS table on the local machine. This is a combination of malware and phishing because typically malware is installed to overwrite the computer's DNS table. The DNS table converts words such as www.ebay.com into numbers for the computer to use in the form of an IP address. If an attacker poisons the DNS table then when the victim visits a site such as www.ebay.com then the phishing site will be loaded instead because the phishing site's IP address is associated with the domain www.ebay.com. This is an advanced technique and if there is malware installed on the machine, then the attacker probably has access to personal files on the machine. Phishing sites are probably the least of ones worries if the DNS table is poisoned.

Phishing lifecycle

Phishing websites have a very short life span. Phishing sites can be active as little as a few hours or as long as several days. Typically most phishing sites are only active a few days before being detected and shutdown [12]. Because of their short life span a blacklist can easily miss new phishing sites.

Economy of phishing

A phisher doesn't necessarily need much technical expertise to be successful. There are tools available to send mass emails and create phishing pages. In fact there is a marketplace for compromised hosts where a phishing page can easily be hosted. There are programs to send harvested confidential information from phishing pages to an email account or chat room user. Chat rooms are often used to buy and sell harvested credentials. The phisher may only act as a middleman and sell the confidential information to a criminal who will exploit that information through identity fraud or other means. "The phishing economy is a decentralized and self-organized social network of merchants and consumers governed by laws of supply and demand." [2]

CHAPTER 3. LITERATURE REVIEW AND EXISTING SOLUTIONS

Phishing is a growing problem on the internet today for both consumers and businesses. One of the most common approaches for an attacker is to create a copycat website in order to capture personal information from consumers. A malicious website may look identical to an online bank or other financial institution in order to capture passwords, social security numbers, account numbers, and other confidential information. A victim may not identify the malicious site until after the confidential information has been leaked. While the phishing life cycle typically begins with a fraudulent email, this research focuses on detection methods using the client's web browser.

Introduction

Currently most web browsers detect phishing sites in a similar manner to detecting viruses. A black list of known phishing websites is stored in a database and uploaded to the web browser during a regular interval much like a list of known virus signatures is updated on antivirus software. There are, however, other techniques in use as well as many proposed solutions including domain checks, URL examination, page content, community input, and algorithms. Recent anti-phishing literature is examined and then techniques are categorized.

Blacklists

A blacklist in the context of phishing is a list of untrusted URLs or more simply a list of banned websites that are known to have malicious intentions. For example if a Mozilla Firefox version 2 user browses to a known phishing site, Firefox realizes that the site URL or

IP address matches the blacklist and displays a warning to the user [14]. While this is a good start to addressing the problem there are two main assumptions with this method.

The first assumption is that the browser will automatically update itself during regular intervals before the user visits the phishing site. A user could turn off the updates altogether or set the interval length too long. Or the user could unfortunately be one of the first persons to visit the phishing site before it has been reported to the blacklist.

The second assumption is that the URL will not change. Attackers often change IP addresses and URLs to avoid detection and prosecution. One should expect a phishing site to change IP locations and ISPs (Internet Service Providers) regularly [12]. While maintaining a known black list of phishing sites is a good first step in phishing detection, it should only be the first part of the solution.

Owner information

This category of detection attempts to determine information about the owner of the webpage. A domain name must be registered with a central authority and certain information about the owner of the domain is available to the public. Most of this information is found during a "whois" query. One can simply enter the domain name on www.whois.net and the registration information will appear. There are also interfaces available so software programs like anti-phishing toolbars can query the database and receive the results. The results are then analyzed or displayed via the toolbar. Some of the information used for anti-phishing is origin country, age of registration, and owner. However, often this information may not correspond to the actual company or owner of the domain because the domain may be leased or the owner may have specified that the information remain private.

Suspicious URLs

Many phishing URLs have patterns that raise the likelihood of a potential phishing site. Using IP addresses instead of words such as http://218.1.0.128 is still a popular way to disguise the domain of a phishing site. A URL containing the '@' symbol will ignore everything to the left of the '@' symbol. For example a URL of "http://www.ebay.com@http://www.phishing-site.com" will navigate to the phishing site instead of eBay. Additional checks include excessive dots in the URL, redirection servers, misspelling, non-standard port numbers, and keywords. Current web browsers such as Microsoft Internet Explorer 7 [13] detect some of these suspicious URL techniques and automatically prevent the user from visiting the suspicious site.

Page content information

Phishing sites must obtain input from the user in order to be effective. Searching the webpage for input fields such as username and password information will help determine if the site is potentially dangerous. If no input fields are detected then the site is most likely harmless. Most phishing sites prompt the user to update sensitive information by logging in with a password or verifying personal information such as a credit card number or social security number. Solutions can detect these input fields on a website and warn a user if the site is suspected to be a phishing site. The page can also be examined for other information such as image logos that match known legitimate sites but do not appear on the legitimate site's domain.

Community Information

The most popular anti-phishing detection mechanism relies on a blacklist of known phishing sites that is maintained by an anti-phishing community. If the user attempts to visit a URL on the blacklist then the user is prevented from visiting the site or warned that the site is a known phishing site. Some solutions use a whitelist which is the opposite of a blacklist and contains a list of known trusted sites. Other techniques rely on a community of users to mark the site as phishing or not and display the results of the currently viewed site to the user. A site's popularity may also be indicated by community information. Google's Pagerank [7] and Netcraft's anti-phishing toolbar [15] both rank sites based on popularity and risk. Most anti-phishing solutions use some type of community information such as a blacklist or whitelist of URLs from trusted sources.

Algorithms

Some well-known solutions to other problems such as spam have been extended to the area of anti-phishing. One of these algorithms is TF-IDF which uses unique keywords to identify a specific page. This technique is often used in text mining or search engines to find relevant pages. The TF-IDF algorithm is applied to the website and keywords are identified. Those keywords are then sent to a search engine such as Google and the top URLs are identified [21]. If the suspected website is located in the top search results then the site is considered legitimate. Otherwise the site is labeled as phishing because most likely the phishing site will not have a high ranking on the search engine results. This method relies on two assumptions. First the phishing page must look and act nearly identical to the legitimate page thereby returning identical TF-IDF keywords. Second the search engine must accurately rank legitimate sites higher than phishing sites.

Another algorithm is the Bayesian filter which was originally developed to detect spam email. Researchers at the University of Iowa are implementing a Bayesian filter based solution to detect phishing sites [10].

One solution from researchers in Hong Kong involves an algorithm that determines the visual similarity of two web pages [11]. This approach examines the visual similarity of one web page and compares it to the visual characteristics of a legitimate page stored in a whitelist.

Password hashing

Other solutions include checking for outgoing passwords by using password hashing techniques or hashing all outgoing passwords with the salt of the domain name so the passwords are unique to each domain [18]. In this example even if a user visits a phishing page and divulges a password the password will not work on the legitimate page because it is hashed using the domain of the legitimate page as a salt to the hash.

Comparison of current solutions

Current solutions were evaluated and a matrix formed that shows which techniques each solution implements. The results are in Table 1.

Table 1

Source	Tool	O	L	P	U	B	R	D	W	C	N	F	I	K	Y	A	T	X
1	CallingID	X	X	X	X	X												
1	Cloudmark						X											
1	Earthlink	X	X		X	X		X										
1	Ebay					X			X									
1	Firefox 2					X												X
1	Trustwatch					X			X									
1	IE 7					X												X
1	Netcraft	X			X			X		X								
1	Netscape 8.1					X												
1	Spoofguard							X		X	X	X	X					
2	B-APT								X						X			
3	Cantina		X									X	X	X		X		

Key

O. origin country
L. registration length
P. popularity
U. user reports-
B. blacklist
R. user ratings- users report whether a site is good or bad, each user also has a rating
D. owner information or domain
W. whitelist
C. suspicious characters in URL
N. non-standard port numbers
F. input-password fields
I. images
K. suspicious links
Y. Bayesian filter
A. IP address
T. TF-IDF
X. Unknown heuristics

Sources

1. Phinding phish
2. B-APT
3. Cantina

B-APT

Researchers at the University of Iowa are developing an anti-phishing toolbar named B-APT to combat phishing. The toolbar uses a Bayesian filter which is commonly used in spam email filtering. The main benefit of a Bayesian filter is the ability to detect never before seen items. B-APT uses a whitelist that is "a comprehensive list of US financial institution and e-commerce sites, along with email providers and other major sites with login pages" [10]. The company name and URL are stored together as tuples in the whitelist. When a user visits a phishing site that contains the same company name and tuple the program detects that it isn't the legitimate site and asks the user if he wants to be redirected to the legitimate site. Using a Bayesian filter is a promising solution for zero day phishing detection as it can detect new phishing sites and doesn't rely on a blacklist.

SpoofGuard

SpoofGuard [4] is an anti-phishing toolbar developed by researchers at Stanford University. SpoofGuard incorporates both stateless and stateful evaluations of a web site. Stateless evaluations include detecting suspicious URLs, images, links, and passwords. Many of the URL detection characteristics used in this research in chapter 5 were adopted from SpoofGuard. These include IP address detection, hexadecimal characters, unusual port numbers, and suspicious characters.

SpoofGuard contains an image database of popular phishing targets to raise a phishing score if the image is found. It also examines host names against those in the browser history. SpoofGuard also uses a hashed password file to determine if the user enters a username and password on an illegitimate site. It also detects hashed images in this manner.

A Framework for Detection and Measurement of Phishing Attacks

Researchers from John Hopkins University and Google Inc. have demonstrated that one can often tell if a page is phishing based entirely upon the URL [6]. They claim an accuracy rate of 97.3%. However they also use Google's PageRank in their analysis of the URL which uses information collected from Google. PageRank is a database of web pages that ranks the importance of a page based upon hostname and the age of the site [7]. Phishing pages are generally short lived and not very popular so naturally they have a low page rank while legitimate sites are established and popular so they have a high page rank.

Many similar traits from SpoofGuard are found in this research including IP addresses, hexadecimal, and misspelling domain names. This paper identified the top keywords and companies found in phishing sites listed in Chapter 5. However, there are some phishing sites that appear to have normal URLs such as those found in Table 6. These would most likely only be detected by the low PageRank score which is more of a proprietary page ranking system by Google than a URL check because it uses the PageRank database of websites.

Cantina

Researchers at Carnegie Mellon have created an anti-phishing solution based upon the TF-IDF algorithm often used in information retrieval and text mining. "TF-IDF yields a weight that measures how important a word is to a document in a corpus. The importance increases proportionally to the number of times a word appears in the document, but is offset by the frequency of the word in the corpus.” [21] Essentially Cantina creates a signature of a webpage by taking the five terms with the highest TF-IDF weights and feeds them to a search engine. If the domain name of the current site matches the domain name of the top N results from the search engine then the site is considered legitimate. Otherwise it is considered phishing. Cantina also implements many detection methods from Spoofguard such as suspicious URL and IP address.

CHAPTER 4. A FRAMEWORK FOR DETECTING ZERO DAY PHISHING WEBSITES

Introduction

The challenges, goals, and problem statements of a phishing detection solution are discussed in this chapter.

Challenges

Acceptable algorithm completion time

The solution should not significantly affect the availability of viewing a webpage. Websites should load quickly and with minimal delay. The user should not notice a difference between a website loaded with the phishing detection turned on compared to the same website loaded with the phishing detection disabled.

Database size

In theory the larger the whitelist database the greater percentage of phishing web sites would be detected. However with a large whitelist database the algorithm will take longer to run and eventually a point will be reached where the user is waiting for the algorithm to finish even after the page is loaded.

The web is a large domain

The World Wide Web is a very large domain so implementing a solution that covers such a large domain is very difficult. It is important to first focus on phishing websites that target financial institutions and then move towards other targets.

CSS changes

CSS files and content will most likely change from time to time. There needs to be an update feature for items in the whitelist to check if the CSS content has changed and if so update the whitelist database.

Problem statements

URL check

The URL check is a check on the URL of the website. If the URL appears suspicious then it can be marked as phishing, but what makes a URL suspicious? Several ideas were borrowed from Spoofguard and additional checks were added for this research because of the trends found within the phishing websites. One of the problems is that some URLs will appear suspicious even though they are not. For example if a website hosted on a free web hosting company contains one of the top phishing target domain names such as ebay then it will have two URL detections and be marked as phishing. It may not actually be a phishing site, but the URL is suspicious enough to detect it. There are so many different scenarios for each specific URL that it is difficult to provide a compromise between high true positive detection and low false positives. One way is to set a threshold high enough so that most non-phishing sites won't be detected while the phishing sites will still be detected. More specifically the URL detection will focus on those attributes more common in phishing sites than non-phishing sites.

Page content check

The page content check searches the HTML of the webpage for suspicious items and determines if a site is phishing, but what makes a webpage suspicious? First of all, the page has to request input from the user which leads to the check for the HTML input tag. Another method discussed in other research is to search for company logos or images because almost all phishing sites use them. However this method was not implemented because there are many affiliate sites that may contain the same image or logo. Another check is for keywords. Again this wasn't implemented to reduce the false positives of affiliate sites or sites that just use the keywords to discuss the company. The "saved from" comment is a great way to check if a page is merely copied locally. Javascript poses a difficult problem because one can use Javascript to create items on a webpage and the usual HTML content such as keywords and input fields may not appear in the HTML source code of the site.

CSS check

CSS has proven to be a great method of detecting phishing sites, but there are obvious weaknesses. First the site has to be using an external style sheet. While the majority of phishing sites use CSS there are still a few that do not. Without a CSS file one cannot currently apply the CSS detection method. However one way to address this problem is to look at the inline styles included in the HTML content and compare the styles to those in the whitelist. This method wasn't implemented, but is addressed further in the Future Work chapter. Another problem with CSS is multiple sites may use the same CSS file location such as eBay and PayPal. These sites use the same CSS location for different country sites so this will trigger a false alarm in the detection. Another problem is some companies offload their bandwidth to a third party for downloading images and CSS files. These third party companies may host CSS files from many different companies thus causing a false detection in the CSS algorithm.

Goals of this research

Detecting zero day phishing web sites

One of the primary goals of this research is to detect zero day phishing web sites. Current practices do a decent job of detecting known phishing sites, but there is a time delay before those phishing sites are added to the blacklist. A user that visits a new phishing site is vulnerable until that site is added to the blacklist. Using heuristics like the URL and CSS detection algorithms is one way to detect new phishing sites.

Unobtrusiveness

Any anti-phishing software should be unobtrusive to the user. Ideally the software works behind the scenes and verifies that a website is legit or warns the user of a suspicious site. However, it is very difficult to correctly identify every website. Therefore the user must somehow interact with the filter in an easy to use manner. If the software is too obtrusive and blocking sites the user wants to visit then the user will just become frustrated and disable the filter. Therefore it is important to balance security with availability.

Integration with common web browsers

The software should integrate easily with commonly used web browsers. Ideally the anti-phishing filter should be included with the web browser and enabled by default. The common user will not seek out and download a phishing filter so providing a default setting that is enabled by default will assist users with little technical or phishing knowledge.

Low false positive rate

Ideally the software should have a low false positive rate. This means that legitimate sites should not be detected as fraudulent sites. This is probably the most volatile variable because a false positive depends a lot on the type of algorithm used to identify potentially fraudulent sites.

High true positive rate

The software should detect the majority of fraudulent sites. Fraudulent sites that are not identified should amount to fewer than five percent of total fraudulent sites visited.

High true negative rate

The software should detect legitimate sites and correctly verify that the site is legitimate. A correct and up to date white list will substantially contribute to a higher true negative rate which is correctly identifying legitimate sites.

CHAPTER 5. URL DETECTION

The Uniform Resource Locator (URL) detection algorithm is based entirely on the URL of the website. This is a standalone test completed on the client machine that does not rely on information from outside sources such as an anti-phishing database or content from the webpage itself. Because many phishing websites try to impersonate the URL of the target or include suspicious items, this is a good start for detecting phishing sites that are not in a blacklist.

Introduction

URL detection is a good first step for detecting a phishing site, but it still leaves much to be desired. Because phishing websites try to impersonate legitimate sites, they often obfuscate the URL so that it either appears legitimate or contains unreadable characters. Some of the methods used include using IP addresses, hexadecimal characters, and unusual port numbers. It's important to note that these techniques are also sometimes used by legitimate sites. Therefore a phishing detection program shouldn't mark a site as phishing based entirely on one of these techniques being detected. A good approach is to add to a total phishing score if one detection method is triggered. URL detection works best when it is combined with other detection techniques. Many of these techniques are common phishing detection techniques borrowed from other research while some new techniques such as keywords and URL redirection have been added.

IP address

Every alphanumeric URL has a numeric only address associated with it. This numeric value is called an IP (Internet Protocol) address and is a sequence of numbers that uniquely identifies a computer on a network. Phishing URLs often contain IP addresses to hide the actual URL of the website. For example a website URL may be extremely long and look suspicious such as something like this "http://www.freewebhostingcompany.com/markswebsite/todaysphishingpage.html" but the URL that contains the IP address is typically shorter and more standard such as this "http://66.135.200.145". IP addresses are used by phishers to conceal the actual domain name of the website being visited. Sometimes IP addresses are combined with actual keywords or legitimate URLs that actually aren't used during URL navigation such as this URL "http://www.ebay.com@100.101.102.3". URL detection methods can look for an IP address in the URL and add to a phishing score if one is found. However one should note that legitimate websites sometimes use IP addresses especially for internal private devices that aren't accessible to the public. Network devices such as routers, servers, and networked printers are often accessed using an IP address in a web browser.

Hexadecimal characters

Much like the URL can be represented with a numeric only value, each character on the keyboard can be represented with a numerical value that the computer understands. This numeric decimal value can easily be converted into hexadecimal base 16. Web browsers can understand hexadecimal values and they can be used in URLs by preceding the hexadecimal value with a '%' symbol. For example the value %20 is the hexadecimal equivalent of the space character on the keyboard. Existing work has focused on detecting hexadecimal values, but this research found that many legitimate sites were detected because often legitimate sites use hexadecimal notation for punctuation and symbols such as the question mark, period, apostrophe, comma, spacebar, etc. Therefore the detection should only be positive when the hexadecimal value is a valid letter or number. Typically phishing sites use hexadecimal values to disguise the actual letters and numbers in the URL.

Unusual port number

When a computer connects to another computer the service uses a specific IP address and a port number. One analogy is the IP address is like a physical address of a building and a port number is the department or individual in the building. Spoofguard identified several standard port numbers as 21, 70, 80, 443, 1080. These correspond to common services used in web browsers such as FTP, Gopher, web, secure web, and SOCKS. If a suspicious unknown port number is used the phishing score is increased because attackers often use different port numbers to bypass security detection programs that may monitor a specific port number.

Suspicious character

Spoofguard identified two characters common in phishing URLs the '@' and '-' characters. By far the most common and dangerous character used in phishing URLs is the '@' character. This character is used by web browsers to automatically pass a username to a secure site. The username proceeds the '@' symbol and the destination URL follows the '@' symbol. The problem is that if the website isn't setup to handle a secure connection, the web browser will navigate to the destination URL without any error message. Phishers exploit this weakness by filling the username field with a legitimate URL while the destination URL is a phishing site. For example the URL "http://www.bankofamerica.com@phishingsite.com" will navigate to the destination URL which is "phishingsite.com" and will attempt to login using "www.bankofamerica.com" as the username. Obviously this is a great way to disguise the actual URL of the website and combined with an IP address one can really hide the phishing site while the URL appears to be legitimate.

Spoofguard identified a dash '-' as the other suspicious character. However it was determined through experimentation that the dash is not a good indicator of a phishing site. Many legitimate URLs use a dash while few legitimate sites use a '@' symbol. Therefore the dash was removed from the list of suspicious characters leaving only the '@' symbol at this time.

Keyword check

Phishing websites use keywords in the URL to lure victims to the phishing sites and provide a false sense of security. Researchers at Google and Johns Hopkins University have identified several common keywords found in phishing URLs. They are: secure, account, webscr, login, signin, banking and confirm [6]. The phishing score increases if one of the keywords in this list is found in the URL.

Company check

Phishing websites want to appear as legitimate as possible so they often contain the name of the company they are targeting. Researchers from Google and Johns Hopkins University identified the most prevalent phishing targets. Most of these targeted sites are included in the company check. The list includes eBay, Paypal, Volksbank, Wells Fargo, Bank of America, Private Banking, HSBC, Chase, Amazon, Banamex, and Barclays [6]. The corresponding domain names of these companies were determined and the company keyword list includes: ebay, paypal, volksbank, wellsfargo, bankofamerica, privatebanking, hsbc, chase, amazon, banamex, and Barclays. The total phishing score increases if one of the keywords in this list is found in the URL.

Redirection

Sometimes phishers want to hide the URL completely by using a URL redirection service. This type of service will shorten the URL into a series of letters or numbers. Services like tinyurl.com and notlong.com allow anyone to enter a URL and the service will essentially create a shortcut to the URL such as “http://www.tinyurl.com/12345”. Obviously this can be used for both good purposes to shorten a URL and malicious purposes to hide the actual URL.

Free web hosting

One trend discovered during this research was the large percentage of free web hosting companies used by phishing websites. Free web hosting sites offer web space to anyone for free and are very attractive to phishers. Because there is no cost as a third party is hosting the actual computer equipment, the phisher can remain anonymous, and when the site is discovered and shut down the phisher can just move on to a different hosting service.

The URL is checked for a free hosting domain name such as lycos.com, altavista.org, ripway.com, tripod.com, and freewebs.com. These are the common web hosting providers found in the phishing URLs. The phishing score increases by one if a free web hosting company is found in the URL. Obviously free web hosting services are used for non-malicious purposes as well so it's important to not mark the site as phishing based solely on the free web hosting check.

Anonymizer

Anonymizers can conceal the actual URL and provide a way for users to surf a website anonymously. They are often used by privacy conscious web users who don't want any personal information sent to a website or want to conceal their own IP address. Phishers can use an anonymizer or web proxy to act as a middleman between the user and the phishing site. Generally this is used to conceal the actual phishing site's URL as the anonymizer's URL will include a specific parameter that translates to the actual URL that is displayed. The list of common anonymizers and proxies found in phishing websites is found in Table 2. If an anonymizer or proxy is found the phishing score increases by one.

Table 2

Anonymizers and web proxies found in phishing sites
www.behidden.com
Web.archive.org
www.proxco.info
www.schoolproxy7
Freesurf11.info
Supahproxy.com
www.the-cloak.com
www.hidemyass.com
www.nijacloak.com

CHAPTER 6. PAGE CONTENT DETECTION

Page content detection relies on detecting certain properties of the actual content found in the webpage source code. Page content detection is advantageous over URL detection because there is a lot more information to examine. However, page content detection is more time consuming than URL detection because there is more information to process and search.

Introduction

Many phishing detection solutions use page content detection in some form. Common techniques such as checking for a user input field are used by Spoofguard. Other techniques use images or keywords to identify phishing sites. Another interesting signature based approach checks for the "saved from" comment that is added to a website's HTML code when the content is saved locally to a file. The new CSS detection algorithm proposed in this research is also a content based detection approach to anti-phishing.

Input field

The goal of a phishing website is to obtain sensitive confidential information from the user such as a username and password, social security number, credit card number, bank account number, or pin number. A user must enter some information into the phishing website in order for the information to leak. The most common way of entering information into a website is through an HTML input field. The input field is used in HTML code to produce a textbox where users can enter information. It begins with the phrase "<input" in HTML so any webpage with this tag should be further examined. It is the most commonly

used and widely accepted method of receiving input from the user. If the input tag is not present in the website then it is most likely not a phishing threat so one can examine the page for the input tag and decide whether to proceed with other detection methods.

Images

One common way phishing pages appear legitimate is by using images and logos from the legitimate target page. Most, if not all, legitimate companies have a unique copyrighted company logo that appears on their website. This unique logo assures the user that they have visited the company's website and provides a unique signature for the website. Users expect to see the familiar logo when visiting a company's website and often assume that the webpage is legitimate because the logo is present. However, this logo is very easy to copy and reproduce in the digital world. Phishing websites can easily use an image link to link to the original image on the legitimate site or the image can be downloaded and stored on the phishing site.

One method of phishing detection using images is to create a database of images with their associated domain names and if the image is found outside the domain then the page is suspected as phishing. In practice however there are often many legitimate affiliate sites that may contain a logo of a partner company along with a link to that company. These affiliate sites would be suspected of phishing because they are using a logo that is not their own. Because images aren't necessarily attached to a specific website it is difficult to use them to detect phishing websites.

Keywords

Because phishing sites attempt to steal personal information, certain keywords are often used in the webpage. Keywords such as username, password, login, pin, account number, name, address, etc. are used to prompt the user to enter sensitive information. A phishing detection program could search for these keywords and add to a phishing score if they are present. However many legitimate sites also ask for this information so the challenge lies in accurately sorting the good sites from the bad ones.

Saved from comment

One of the easiest ways to copy a website and store it locally is to open the site in a web browser and choose the "File->Save As" option. This allows one to store a local copy of the current webpage in view. Often additional information is added to the top of the webpage in the form of a comment. A comment is a message inside the HTML code that is not visible when the webpage is viewed, but is visible if one examines the HTML source code. Comments are usually reserved for web programmers to make notes about the HTML code, but in some cases web browsers automatically add a comment to the webpage if it is stored locally. The comment will often begin with the phrase "saved from url" such as the following comment from a Carfax vehicle history report that was saved locally.

```
<!-- saved from url=(0057)http://www.carfax.com/cfm/FSBO.cfm?report-->
```

CSS

Cascading Style Sheets (CSS) offer a unique signature of a specific website. A style sheet is a file on a web server that defines the styles, fonts, spacing, and other information about how the website will appear in the web browser. The assumption is that each legitimate website will have a unique style and therefore a unique style sheet. If the phishing detection program identifies a style sheet that doesn’t belong to a particular site, then that site will be marked as phishing. CSS detection is a new and unique contribution to phishing detection and is further discussed in Chapter 6.

CHAPTER 7. CSS DETECTION

Websites use cascading style sheets (CSS) to easily change and modify the style, appearance, and look of a website without modifying the content of the page. Essentially, a cascading style sheet contains all the information about fonts, styles, colors, background information, layout, placement, and anything else about how the page looks and feels. The majority of websites has adopted external CSS files as a standard and uses one or more unique style sheets when displaying page content to the web browser. The style sheet file is an excellent signature for a website because each website has a unique look and feel so the style sheet is unique to the owner of the site.

Introduction

Standard external style sheets are text files that end in the extension ".css". Nearly all of the legitimate and phishing webpages examined use an external style sheet. Only 1% of eBay phishing pages and 12% of banking phishing pages did not use a style sheet. This makes CSS an excellent choice for a phishing detection program. A whitelist is needed to store the legitimate website information. There are two methods of CSS detection: CSS link detection and CSS content detection.

Assumptions

The first assumption for using CSS as a phishing detection program is that a phishing website will use CSS while mimicking a legitimate site. Over 95% of the phishing sites examined use CSS with only a handful not using CSS at all. In addition, many of the phishing websites use a direct link to the actual CSS file hosted on the legitimate site which makes detection very easy and fast.

The second assumption with the CSS phishing detection method is that a legitimate website will own a unique CSS file hosted on a domain that they own or lease. Upon examination of the list of 500 legitimate websites by 3Sharp [1], one can conclude that this assumption is correct in the majority of cases. Over 90% are using a CSS file in their own domain or sub domain and the other 10% are hosting the CSS file on an affiliating domain such as a sister company. Also some of the CSS files contain copyright information right in the text of the file so copyright infringement is additional evidence that each legitimate website must use a unique CSS file.

Another assumption is that the phisher will not alter the CSS file enough to prevent a CSS detection. The goal of the phisher is to reproduce a website that looks identical to the legitimate site in order to trick a user into believing that it is the legitimate site. Because a phisher needs to reproduce as many details as possible, one can conclude that nearly every style, font type, layout, etc. of the webpage should match the legitimate site. The easiest way to reproduce the look and feel of a website is to use that site's style sheet as this is where all the style information is contained.

The final assumption is that the user will add his financial websites or other confidential sites to the whitelist. The other alternative is that the whitelist will contain a comprehensive list of popular targets. If the legitimate site is not listed in the whitelist, then the CSS detection algorithm will not detect phishing sites targeting that specific site because the CSS detection relies on the CSS information of the legitimate site stored in the whitelist database. A mix of user education and a preloaded database of popular targets are needed for the CSS detection algorithm to function properly.

Whitelist

A whitelist in the context of phishing detection is simply a list of trusted websites. For CSS detection to work properly, the list contains more than just the URL of the trusted website. Each entry in the whitelist database contains six strings: the URL of the trusted site, the domain of the site, the title of the site, the CSS filename, the CSS domain, and the CSS content of the file.

URL of the trusted site

The URL of the trusted site is used to periodically update the CSS information in the database. This is the URL of the site such as “https:\\signin.ebay.com”.

Domain of the trusted site

The domain of the trusted site is the domain of the URL such as “signin.ebay.com” and is used to determine whether the current page displayed in the browser is on the whitelist or not.

Title of the trusted site

The title of the trusted site is the page title of the site such as “Welcome to eBay” and can be used during CSS content detection to speed up detection by matching potential phishing site titles with titles in the whitelist database.

CSS filename

The CSS filename is the filename of the CSS file such as “paypal.css” and can also be used during CSS content detection to speed up detection by matching potential phishing site CSS filenames with filenames in the whitelist database.

CSS domain

The CSS domain is the domain of the location of the CSS file such as “secureinclude.ebaystatic.com”. Often the domain is the same as the site domain, but in other cases such as eBay, the CSS file is hosted on a different domain. Storing the CSS domain is essential because if a match is found on a website not in the whitelist, then it is most likely a phishing site linking to the actual CSS file location of the legitimate site.

CSS content

The CSS content is the actual text contained in the CSS file that contains all of the style information. The CSS content is used to compare with the CSS content of a possible phishing site in order to determine if there is a match with a legitimate site.

CSS Link Detection

The easiest method for a phisher to obtain the style sheet of the targeted website is to provide a link to the legitimate site’s CSS file. Many websites already include a full or relative link to the stylesheet file therefore if the phisher simply copies the website content and does not modify it, then the CSS link is already present.

Style sheet standards

The World Wide Web Consortium lists two standards for linking to an external style sheet within an HTML document. These standards provide that basis for the CSS link detection.

"The most standard and best-supported method is to create a link to that document using the <link> tag in the <head> of the document" [16] as shown here:

```
<HEAD>
<LINK REL="STYLESHEET" HREF=" /pathname/stylesheet.css"
TYPE="text/css">
</HEAD>
```

An alternative to linking is to import external style sheets into the <style> element using the @import function:

```
<STYLE TYPE="text/css">
        @import url(http://pathname/stylesheet.css);
</STYLE>
```

Link detection

After reviewing the style sheet standards one can easily conclude how to detect the CSS links in the HTML page content. Once the page source is downloaded the detection algorithm searches for either the phrase "rel="stylesheet" or the tag "@import". If one of these tags is found, the algorithm parses through the text to find the "HREF" or "url" keyword in order to extract the full path to the style sheet. The domain of the current style sheet path is then compared with the CSS domain list and if a match is found the page is marked as phishing.

One may conclude that it is easier to search the page source for ".css" because it is the standard extension for a CSS file, however some phishing sites were found to contain style sheets with the extension changed to something other than ".css" and the browser still loads the style sheet just fine. Therefore it is important to search for the above mentioned standard tags because the CSS file will still load without the proper ".css" extension.

Detection avoidance

Phishing sites use many interesting techniques to cover their tracks and avoid detection methods. Some of the phishing sites examined used an anonymous web proxy such as behidden.com to load the style sheet. An anonymous web proxy acts as a third party between the phishing site and the legitimate site to hide the actual identity of the phishing site from the legitimate site. The legitimate site receives a request from the anonymous web proxy and the proxy is the middleman that forwards the request to the phishing site. This technique hides the identity of the phishing site from the legitimate site. For example, if a legitimate website operator noticed many CSS requests coming from outside of the legitimate domain then the operator may become suspicious that this request is from a phishing site and try to identify the site as phishing. Using a web proxy hides the true identity of the phishing site from the legitimate site because it is actually the web proxy that makes the request. Because the URL of the CSS link is attached as a parameter to the URL of the web proxy the CSS link detection will fail if an anonymizer is used. However there is no practical reason any legitimate site should use an anonymizer to retrieve a CSS file so one can merely mark the site as phishing if an anonymizer is found.

Another method of CSS retrieval to avoid detection is using a legacy location of the CSS file. Some phishing sites use a legacy version of the CSS file location. If the domain of the legacy CSS file is the same domain as the current CSS file then the detection algorithm will still work. However, if the older legacy CSS file is located on another domain then the detection will fail. One easy method to obtain a legacy CSS file is to use an internet archive website such as The Wayback Machine [8]. This site hosts versions of websites from years ago so one can look back in time to a previous version of a website. This provides an excellent opportunity to grab a CSS file from a legitimate site without visiting the legitimate site.

CSS Content Detection

The content of the CSS file is text that defines the styles of the webpage. Therefore it is assumed to be unique to the page or domain that links to it. One way a phisher can avoid the CSS link detection is to download the CSS file and host it locally on the phishing site. The link detection fails because the domain of the CSS file will not be in the whitelist like it would be if the phisher linked to the legitimate CSS file instead of hosting it locally. For example one could download eBay's CSS file and host it at http://www.phishingsite.com/ebay.css instead of linking to eBay's site. Content detection is a necessary second step in the CSS detection algorithm.

There are several ways one could approach detecting similar page content. The ideal detection method would compare two strings for similarity. One approach to the problem is to use the Levenstein Distance Algorithm [9] which computes a score based on how similar or different two strings are. However, this approach proved too time consuming and ineffective for CSS content detection. Therefore a solution was devised that balances speed with accuracy.

The CSS content detection algorithm first parses the CSS content into an array of strings separated by each new line in the CSS content file. Because many lines contain only a '{' or a '}' character for coding purposes, the lines with less than two characters are excluded. One at a time each item in the CSS content database is searched for the lines previously parsed in the suspicious CSS content file. If more than half of the lines are found in one particular CSS content item in the whitelist the search algorithm stops and the page is marked as phishing. This method has proven to be fast and accurate; however it is still a compromise because a phisher could merely add a character to each line in the CSS file to thwart this detection scheme.

CHAPTER 8. EXPERIMENTS AND RESULTS

Introduction

Phishing sites are difficult to conduct experiments on because of their short lived nature and the risks associated with opening harmful websites. Because phishing sites are short lived a web crawling program was created to download the sites when they were still active so they could be reviewed at a later time. Phishtank [17] was chosen as the source of phishing sites because of its wide variety of sites and ease of use with the web crawler program. In order to be sure that the phishing data is accurate one must manually verify that a site is actually a phishing site by examining the suspected page.

Experimenting with phishing sites can be dangerous because some of the suspected phishing sites contained malware that would install unwanted programs on a computer. Often anti-virus software would prevent downloading and viewing certain sites that were already downloaded because they contained malware. Sometimes it was necessary to turn off the anti-virus software in order to examine certain sites. Twice during the experiments the testing computer opened a phishing site that installed malware and disrupted the testing computer for several hours until it was properly removed.

URL experiments

The purpose of this experiment was to determine what percentage of phishing and legitimate URLs would be detected using the URL detection alone. The phishing URLs were confirmed phishing URLs harvested from Phishtank in October of 2008 to use in this experiment. A web crawler detected new submissions to Phishtank that were confirmed phishing sites from Phishtank users. The web crawler downloaded the website and stored it locally along with the URL of the website. The good URLs were selected from a list of legitimate URLs composed by 3Sharp [1]. A program was written in C# .NET to determine if the URL was a possible phishing URL by using all the detection methods discussed in Chapter 4 except for the anonymizers and free web hosting detection methods.

The results in Table 3 show that only 80% of the good URLs were not detected as phishing which translates to a high false positive rate of 20%. The phishing URLs had 28% escape detection which translates to a true positive rate of 72%. These results show that detection by URL alone leaves much to be desired.

Table 3

Detection method	Good URLs	Percentage	Phish URLs	Percentage
URL count	503		10503	
IP address	3	0.6%	686	6.5%
suspicious characters	47	9.3%	3851	36.7%
suspicious port	1	0.2%	79	0.8%
keyword in URL	48	9.5%	5353	51.0%
hex encoded	20	4.0%	125	1.2%
company in URL	7	1.4%	3496	33.3%
tiny URL	0	0.0%	6	0.1%
0 detections	401	79.7%	2973	28.3%
1 detection	80	15.9%	3521	33.5%
2 detections	20	4.0%	2131	20.3%
3 detections	2	0.4%	1717	16.3%
4 detections	0	0.0%	149	1.4%
5 detections	0	0.0%	12	0.1%
6 detections	0	0.0%	0	0.0%
7 detections	0	0.0%	0	0.0%

True Positive	71.7%
False Negative	28.3%
True Negative	79.7%
False Positive	20.3%

Modify the suspicious character detection

After examining the results from Table 3 one quickly realizes that the two major categories where the good URLs are being falsely detected are suspicious characters and keywords in the URL. The keyword detection is expected as keywords are commonly found in both legitimate and phishing URLs. The decision was made to remove the dash '–' from the suspicious character detection as legitimate sites also use this character. After removing the dash there was a drastic decrease in suspicious character detection from both sets of URLs as seen in Table 4. The false positive detection improved by 8% while the true positive detection decreased by 8%.

Table 4

Detection method	**Good URLs**	**Percentage**	**Phish URLs**	**Percentage**
URL count	503		10503	
IP address	3	0.6%	686	6.5%
suspicious characters	1	0.2%	200	1.9%
suspicious port	1	0.2%	79	0.8%
keyword in URL	48	9.5%	5353	51.0%
hex encoded	20	4.0%	125	1.2%
company in URL	7	1.4%	3496	33.3%
tiny URL	0	0.0%	6	0.1%
0 detections	442	87.9%	3810	36.3%
1 detection	44	8.7%	3946	37.6%
2 detections	15	3.0%	2296	21.9%
3 detections	2	0.4%	405	3.9%
4 detections	0	0.0%	44	0.4%
5 detections	0	0.0%	2	0.0%
6 detections	0	0.0%	0	0.0%
7 detections	0	0.0%	0	0.0%

True Positive	63.7%
False Negative	36.3%
True Negative	87.9%
False Positive	12.1%

Examining the false positives and false negatives

The next step was to manually examine the list of undetected phishing sites as well as the list of legitimate sites that caused detection. The legitimate sites were mostly keyword detections such as login or eBay, but there were a few IP addresses and hex detections in the URLs of legitimate sites. Table 5 shows a sample of legitimate URLs that were detected. The false negatives or phishing sites not detected were also examined. Table 6 shows a sample of undetected phishing URLs. Although generalizations can be made such as all URLs that contain an IP address or suspicious characters are phishing, this is obviously not the case all of the time as seen in Table 5. However one can increase a phishing score for these types of detections and combine them with other detection methods to determine the likelihood of a phishing site.

Table 5

Sample of legitimate URLs detected
http://212.174.200.74/airdev/login/loginindex.jsp
http://elearning.uws.edu.au/webct/ticket/ticketlogin
http://www.ittakes2.com/login.php
http://hub.benl.ebay.be/buy

Table 6

Sample of undetected phishing URLs
http://www.martinclothes.com/curl.php
http://pifflesnespl.com/
http://support.tabor.edu/ws/
http://p41pal.t35.com/
http://h1.ripway.com/pptesting/
http://funytrd.freehostia.com/upload/

Hosting sites may add to phishing score

Manually examining the undetected phishing sites revealed a common occurrence. Many of the phishing sites were hosted by a free web hosting service. If a URL is hosted on a free web hosting service then the phishing score could be increased to indicate that the sight is potentially suspicious.

Some of the popular hosting services found are listed in Table 7.

Table 7

Common free web hosting sites detected
altervista.org
ripway.com
lycos
freewebs.com

Financial institutions are common phishing targets

As expected many different financial institutions are common targets for phishing. The banks listed in Table 8 were commonly found in the list of undetected phishing URLs and could be added to the company keyword list to increase true positive detection.

Table 8

Common bank domains detected
Wachovia
chase
colonialbank
bankofamerica
abbeynational
suntrust
rbccentura
capitalone
citibank
tdbanknorth
wellsfargo
firstbanks

eBay phishing experiment October 2008

Several thousand phishing sites were harvested from Phishtank in October 2008 and their contents were downloaded and stored locally for review. This experiment focuses on eBay because it is one of the most targeted phishing sites. A program was written to search the HTML content of the phishing sites for “ebay” and then sort them for further review. Each match found to contain “ebay” was manually reviewed by a human to verify that it was indeed a phishing site imitating eBay. Many sites were discarded from this experiment because they contained the word “ebay” and were sorted by the program, but after manual review they were found not to be eBay phishing sites. A total of 376 confirmed eBay phishing sites were used for this experiment.

The sites were further sorted by the type of CSS found in the page content to determine how effective the detection algorithm correctly identified the different categories of CSS detection. The categories were current eBay CSS file, legacy eBay CSS file, anonymous CSS file, and no CSS found. The current eBay CSS file category contained the phishing sites that had a CSS reference to the current path of eBay's CSS file found on the domain “secureinclude.ebaystatic.com”. The majority of the eBay phishing sites was linking to eBay's current CSS file and fell into this category. The legacy eBay CSS file category contained the eBay phishing sites using a legacy or alternate path of eBay's CSS file found in the domain “include.ebaystatic.com”. The anonymous category contained the phishing sites using an anonymizer to retrieve the CSS file. Finally if a site didn't use CSS at all then it fell into the no CSS category.

A program was written to loop through each downloaded phishing site and determine if it was considered phishing or not by applying the URL and CSS detection algorithms discussed in previous sections. The results are in Table 9.

Table 9

CSS Type	**Detected**	**Undetected**
eBay	358	0
Legacy eBay	8	0
Anonymous	5	0
No CSS	1	4
Total	**372**	**4**

As seen in Table 9 every single phishing site that linked to eBay's website was detected as phishing. What is surprising is that every single phishing website using an alternate path of eBay's CSS file was also detected. This is most likely due to the CSS content detection algorithm that noticed the CSS file was very similar to the eBay CSS file stored in the whitelist database. All of the phishing sites using an anonymous web browser to access the CSS file were also detected.

As expected, the detection program missed 80% of the phishing sites not using CSS. The good news is that only about 1% of the eBay phishing sites were not using CSS which corresponds to the generalization found in this research that more than 95% of both legitimate and phishing sites use CSS. The bad news is that if the phishing site doesn't use CSS then it will most likely be undetected. This is the main weakness of the CSS detection algorithm and suggestions for improvement are discussed further in the Future Work chapter.

Overall the phishing detection algorithms performed excellent on the eBay phishing sites with a true positive rate of 99% and a false negative rate of 1%.

PayPal phishing experiment October 2008

PayPal is consistently listed in the top targets of phishing sites along with eBay so an experiment similar to the eBay experiment was conducted with PayPal phishing sites. A program was written to search the HTML content of the phishing sites for “paypal” and then sort them for further review. Again each match found to contain “paypal” was manually reviewed by a human to verify that it was indeed a phishing site imitating PayPal. Many sites were discarded from this experiment because they contained the word “paypal” and were sorted by the program, but after manual review they were found not to be PayPal phishing sites. A total of 647 confirmed PayPal phishing sites were used for this experiment.

The sites were sorted into two categories: CSS phishing sites and non-CSS phishing sites. A program was written to loop through each downloaded phishing site and determine if it was considered phishing or not by applying the URL and CSS detection algorithms discussed in previous sections. The results are in Table 10.

Table 10

Category	**Detected**	**Undetected**
CSS	627	0
No CSS	4	16
Total	**631**	**16**

The results are very similar to the eBay phishing sites with all of the phishing sites that used CSS being detected and many of the phishing sites not using CSS being undetected. The detection missed 75% of the non CSS sites, but overall the PayPal detection rate was very good with a true positive rate of 97.5% and a false negative rate of 2.5%.

eBay local CSS phishing experiment February 2009

The experiments used on the phishing websites downloaded in 2008 excluded the few phishing sites that used local CSS files because at the time of the download the local CSS detection hadn't been created and therefore there was no need to download the local CSS files of the phishing sites. However as the CSS detection algorithm evolved to include a check on local CSS files there was a need to examine phishing sites using local CSS files to evaluate the effectiveness of the local CSS detection algorithm. Therefore an additional twenty four eBay phishing sites were downloaded in February 2009 along with their local CSS files if applicable. Again these sites were manually confirmed to be phishing sites targeting eBay.

Of the twenty four phishing sites downloaded eight of them used a local CSS file and all of the local CSS files were detected as phishing. There were three categories similar to the eBay categories in the October 2008 experiments with the addition of the local CSS category. The eBay phishing sites that used a local CSS file rather than a linked CSS file were sorted into the local CSS category. The results are in Table 11.

Table 11

Category	**Detected**	**Undetected**
CSS link	15	0
Local CSS	8	0
No CSS	1	0
Total	**24**	**0**

As shown in the table the local CSS detection along with the URL detection worked well detecting 100% of the local CSS files targeting eBay. Combining all categories of the eBay experiments from 2008 and 2009 yields a respectable true positive rate of 99% and a false negative rate of 1%.

However an additional experiment was conducted to test the effectiveness of using just the local CSS detection by itself. For this experiment the URL detection was turned off so the only detection algorithm available was the local CSS file detection. The purpose of this experiment was to determine the effectiveness of just using the local CSS detection algorithm. Without the URL detection, only six of the eight phishing sites using a local CSS file were detected which leaves room for improvement with a undetected or false negative rate of 25%. The other two phishing sites were previously detected through the URL detection algorithm. This provides further evidence that many layers of detection techniques working together will yield the best results.

PayPal local CSS phishing experiment February 2009

A similar experiment to the eBay 2009 experiment was setup to evaluate the local CSS detection on PayPal phishing sites. An additional thirty one PayPal phishing sites were downloaded in February 2009 along with their local CSS files if applicable. Again these sites were manually confirmed to be phishing sites targeting PayPal.

Of the thirty one sites downloaded six of them used a local CSS file and all of the local CSS files were detected as phishing. There were three categories similar to the PayPal categories in the October 2008 experiments with the addition of the local CSS category. The PayPal phishing sites that used a local CSS file rather than a linked CSS file were sorted into the local CSS category. The results are in Table 12.

Table 12

Category	**Detected**	**Undetected**
CSS link	23	0
Local CSS	6	0
No CSS	2	0
Total	**31**	**0**

As shown in the table the local CSS detection along with the URL detection worked well detecting 100% of the local CSS files targeting PayPal. Combining all categories of the PayPal experiments yields a respectable true positive rate of 98% and a false negative rate of 2%.

However an additional experiment was conducted to test the effectiveness of using just the local CSS detection by itself. For this experiment the URL detection was turned off so the only detection algorithm available was the local CSS file detection. The purpose of this experiment was to determine the effectiveness of just using the local CSS detection algorithm. Without the URL detection, only four of the six phishing sites using a local CSS file were detected which leaves room for improvement with an undetected or false negative rate of 33%. The other two phishing sites were previously detected through the URL detection algorithm. Again this provides further evidence that many layers of detection techniques working together will yield the best results.

Banking phishing experiment October 2008

A program was written to search the HTML content of the phishing sites for the keyword “bank” and then sort them for further review. Again each match found to contain “bank” was manually reviewed by a human to verify that it was indeed a phishing site imitating a financial institution. A total of 96 confirmed banking phishing sites were used for this experiment.

The sites were sorted into two categories: CSS phishing sites and non-CSS phishing sites. The banks found in the phishing sites were added to the whitelist much like a user would add a specific bank to the whitelist. A program was written to loop through each downloaded phishing site and determine if it was considered phishing or not by applying the URL and CSS detection algorithms discussed in previous sections. The results are in Table 13.

Table 13

Category	**Detected**	**Undetected**
CSS	93	3
No CSS	1	14
Total	**94**	**17**

For the banking sites using a CSS link there was a true detection rate of 97% and 3% of the CSS sites went undetected. This was mainly due to these specific banks not being added to the whitelist or the CSS file was located in a sub domain of a URL with a country code. The program was unable to add one site to the whitelist specifically due to the banking site's security features. The site would close the connection when the program tried to download information such as the CSS file. Further investigation is needed into this implementation problem, but one can conclude that certain sites are preventing the toolbar from adding the site to the whitelist.

The other undetected CSS site was using a CSS file found in the sub domain of the CSS domain stored in the whitelist. Because of country code extensions the sub domain parsing becomes difficult with a country code extension on the end. If these implementation issues are corrected the results would most likely change to 100% detection for the banking sites using CSS.

The banking phishing sites had a much larger percentage of non-CSS sites than the eBay and PayPal phishing sites. Because of this statistic there were a larger percentage of the banking sites undetected by the program. Only one out of the fifteen sites with no CSS was detected.

Overall the program performed well on the CSS sites as expected and poorly on the non-CSS sites. The two categories combined yields a detection rate of 85% which is much better than a blacklist at detecting new phishing sites but still leaves room for improvement of detecting the non-CSS phishing sites.

Banking local CSS phishing experiment February 2009

A similar experiment to the eBay and PayPal 2009 experiments was setup to evaluate the local CSS detection on banking phishing sites. An additional forty eight banking phishing sites were downloaded in February 2009 along with their local CSS files if applicable. Again these sites were manually confirmed to be phishing sites targeting online banking sites.

There were three categories similar to the banking categories in the October 2008 experiments with the addition of the local CSS category. The banking phishing sites that used a local CSS file rather than a linked CSS file were sorted into the local CSS category. The results are in Table 14.

Table 14

Category	**Detected**	**Undetected**
CSS link	21	2
Local CSS	8	12
No CSS	0	5
Total	**29**	**19**

The results of this experiment were quite surprising. The program caught most of the banking sites using a CSS link to the legitimate site's CSS location. The two undetected sites were from a banking site that could not be added to the whitelist so the program didn't have the bank's information in the whitelist.

The surprising results come from the results of the local CSS category. The program had a high rate of false negatives in this category. In other words there were many phishing sites that were not detected by the program. Upon further investigation the CSS files of the sites in this category were significantly different than the CSS files of the legitimate target banking sites. For example some CSS files had certain words missing, extra return characters, and some were possibly legacy CSS files. All were able to defeat the CSS content checking algorithm which leaves much room for improvement.

Improving the CSS content checking algorithm will improve the detection rate of the local CSS category. Currently the CSS content checking algorithm separates each item in the CSS content by checking for a return character followed by a new line character. In this research this is the most commonly used method used by web designers on the legitimate sites which is why it was chosen as the rule to parse different lines of CSS content. However as one examines the results of Table 14 one quickly realizes that the CSS content checking algorithm needs to be improved so that it recognizes changes in the CSS file.

Combining the two banking experiments yields a true positive rate of 77% and a false negative rate of 23%. Nearly all of the false negatives either don't use CSS or have modified the CSS file which prevented the program from detecting those sites.

Legitimate site assumptions

For this experiment the legitimate sites from 3Sharp were evaluated to determine if two basic assumptions are correct. First the sites were examined to see if they used an anonymizer to retrieve the CSS file. As expected this is a phishing technique to avoid tracing and none of the legitimate sites used an anonymizer to retrieve the CSS. The majority of legitimate sites used a local or linked CSS file on the same domain or sub domain of the URL. Less than ten percent used a CSS file on a different domain that was typically an affiliate website owned by the same parent company.

Legitimate sites detected

The legitimate sites were analyzed by the program to determine the number of false detections. Obviously the ideal detection algorithm will have no false positives, but an acceptable rate would be a few percent. Of the legitimate sites in the 3Sharp list, 390 were reachable by the program. Only 3 out of 390 legitimate sites were marked as phishing. This translates to an acceptably low false positive rate of less than 1%. The three sites marked as phishing are examined below.

The first site marked as phishing was an eBay site in Belgium with the URL http://hub.benl.ebay.be/buy. This site was detected because every different international eBay site uses the same centralized CSS domain at ebaystatic.com. This is possibly a problem with other large organizations and is discussed in the future work section. For now each user will need to add the specific international page to the whitelist.

The second site marked as phishing contained the keyword "login" and had an IP address in the URL. The URL appeared suspicious and matched the profile of a phishing site so it was marked as phishing. However it was actually a login page for a car dealership. There's not much one could do to prevent this type of false positive because the URL appears so suspicious. Again one can only add it to the whitelist.

The final site marked as phishing was interesting because it was detected with the CSS content check. The CSS content is provided in Figure 1. This CSS file is very small for one and it is also very generic. These two features make it easy to detect half of its content in other CSS files. This is merely an exception that will be detected because of the way the CSS content detection operates and the generic nature of this particular CSS file.

```
/* °øÅë¿ä¼Ò */
* {
background-position:0 0;
background-repeat:no-repeat;
padding:0;
margin:0;
text-align:left;
}

li {list-style:none;}

a {color:#666;text-decoration:none; vertical-align:middle;}
a:hover {text-decoration:underline;}
a img {border:none;}

img {vertical-align:middle;}

form * {vertical-align:middle;}

fieldset {border:none;}

/* °øÅë ·¹ÀÌ¾Æ¿ô */

body {
padding:0;
margin:0;
color:#666;
width:100%;
font: 12px/1.5em "±¼¸²", "µ¸¿ò", Seoul, "ÇÑ°Ã¼", sans-serif;
background-repeat:repeat-x;
}
```

Figure 1

Zero day phish

An easy method to verify that the toolbar will detect new phishing attacks is to visit www.phishtank.com and look at the recently reported phishing sites. Some of these recently reported sites are not yet in the blacklist database used by the phishing filter in Internet

Explorer so even if the phishing filter is turned on Internet Explorer will not detect many of these sites. The toolbar however was able to accurately detect these zero day phish even when Internet Explorer did not detect them.

It is difficult to set up an experiment to compare Internet Explorer with the anti-phishing toolbar developed in this research because one must manually examine the new phishing pages and see which solution detects the phishing sites. The other problem is that over time Internet Explorer will obviously be more effective because the phishing site will eventually be added to the blacklist.

Manual verification of a few phishing sites on different days and times has shown that the anti-phishing toolbar is effective at detecting new phishing sites that Internet Explorer does not detect as phishing. This is by design as the toolbar is designed to detect new phishing sites while Internet Explorer relies mostly on a blacklist of known phishing sites.

CHAPTER 9. TOOLBAR IMPLEMENTATION

A simple web browser toolbar was implemented to demonstrate one possible anti-phishing solution using the detection methods discussed in this research. A server side solution has many benefits, but was ultimately discarded in favor of a client based solution. The toolbar is designed to assist the user in identifying phishing sites while using Microsoft Internet Explorer web browser.

Introduction

A server based solution that resides on a central computer server or firewall has a lot of benefits for the IT department such as no software to install on the client machine, ease of management, ease of updates, etc. but it does have a couple of drawbacks. The first disadvantage of a server based solution is the inability to monitor secure traffic such as SSL protocol used by many financial websites. This would inhibit the software from adding secure pages to the whitelist and examining the contents of secure websites. The other drawback is the whitelist would have to contain many more entries to support a diverse group of users. Therefore a client based browser toolbar was implemented. A browser based toolbar is simply an add-on for an internet browser that typically rests underneath the address bar in the browser and can display certain information to the user such as a warning that the current site is a phishing site. Figure 2 shows the toolbar underneath the address bar displaying a status of “checking site.”

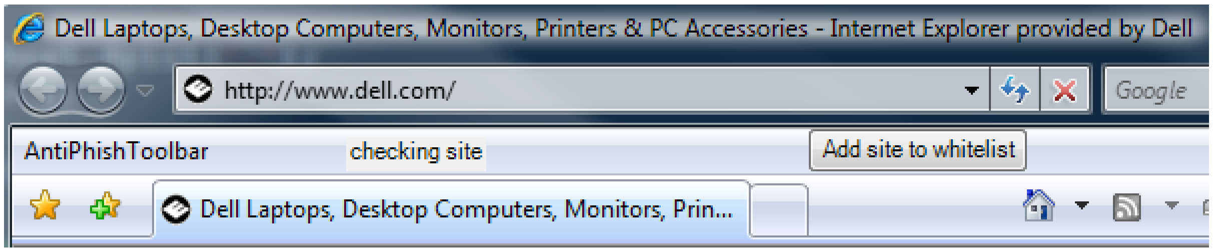

Dell Laptops, Desktop Computers, Monitors, Printers & PC Accessories - Internet Explorer provided by Dell
http://www.dell.com/
Google
AntiPhishToolbar
checking site
Add site to whitelist
Dell Laptops, Desktop Computers, Monitors, Prin...

Figure 2

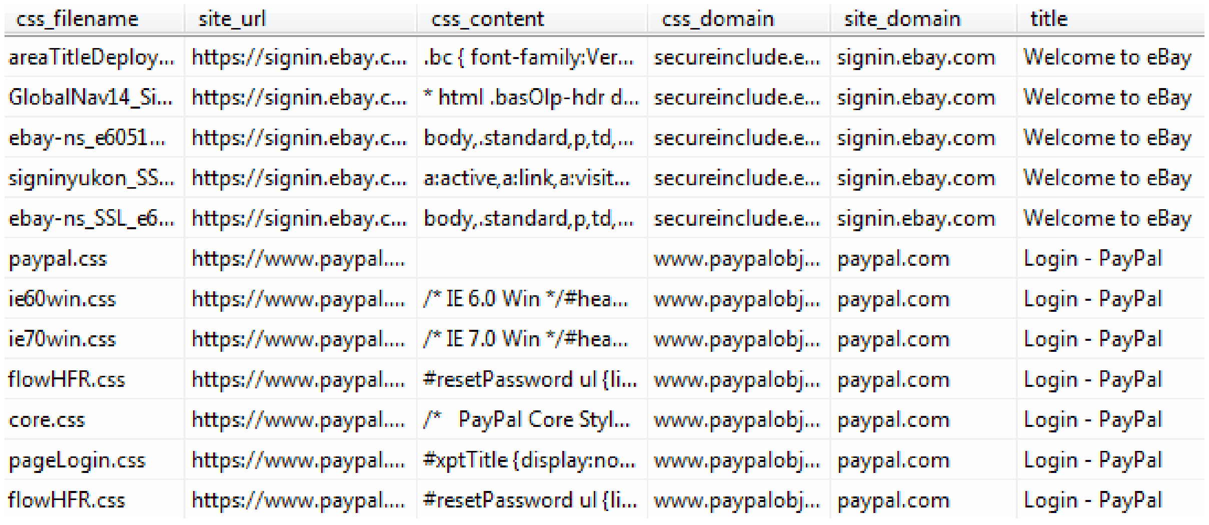
Toolbar design and requirements

The toolbar code is written and designed using Microsoft Visual C# .NET 2008. The software was chosen mainly because of the author's familiarity with C#. Naturally the C# source code and algorithms used for the phishing experiments was adopted and modified to fit the requirements of a toolbar. Add-in Express software for Internet Explorer and Visual Studio [3] assisted with the creation of a toolbar. The toolbar requires Microsoft Internet Explorer version 6 or 7, Microsoft .NET framework 3.5, and Microsoft SQL Server Compact Edition 3.5. All are available free of charge from Microsoft.

The toolbar uses a Microsoft SQL Server Compact Edition database stored locally on the client machine. This database contains the whitelist information and is loaded into memory when the toolbar is active. The toolbar is designed to complement existing anti-phishing measures in Internet Explorer so it does not contain a blacklist. The primary function of the toolbar is to detect and warn the user of new phishing sites that have not been reported and to assure the user that a site is legitimate when a whitelist webpage is accessed. The primary goal of the toolbar is to assist the browser with phishing detection while remaining inconspicuous to the user. Hence, the toolbar uses the URL detection on all sites, but the CSS detection is only effective if the target site is in the whitelist. Therefore it is critical that the user add all appropriate sites such as specific banks to the whitelist upon installation. This can easily be done by visiting the site's login page and clicking on the "add site to whitelist" button found in the toolbar in Figure 2.

Default whitelist database

The default whitelist database is the database that installs automatically when the toolbar is installed. It contains a list of legitimate websites that are common phishing targets. Each entry in the whitelist database contains six strings as discussed in Chapter 7: the URL of the trusted site, the domain of the site, the title of the site, the CSS filename, the CSS domain, and the CSS content of the file. Note that many websites contain more than one style sheet so one site may have multiple entries in the database corresponding to each style sheet on the page. Figure 3 shows the database information for eBay and Paypal.



css_filename	site_url	css_content	css_domain	site_domain	title
areaTitleDeploy...	https://signin.ebay.c...	.bc { font-family:Ver...	secureinclude.e...	signin.ebay.com	Welcome to eBay
GlobalNav14_Si...	https://signin.ebay.c...	* html .basOlp-hdr d...	secureinclude.e...	signin.ebay.com	Welcome to eBay
ebay-ns_e6051...	https://signin.ebay.c...	body,.standard,p,td,...	secureinclude.e...	signin.ebay.com	Welcome to eBay
signinyukon_SS...	https://signin.ebay.c...	a:active,a:link,a:visit...	secureinclude.e...	signin.ebay.com	Welcome to eBay
ebay-ns_SSL_e6...	https://signin.ebay.c...	body,.standard,p,td,...	secureinclude.e...	signin.ebay.com	Welcome to eBay
paypal.css	https://www.paypal....		www.paypalobj...	paypal.com	Login - PayPal
ie60win.css	https://www.paypal....	/* IE 6.0 Win */#hea...	www.paypalobj...	paypal.com	Login - PayPal
ie70win.css	https://www.paypal....	/* IE 7.0 Win */#hea...	www.paypalobj...	paypal.com	Login - PayPal
flowHFR.css	https://www.paypal....	#resetPassword ul {li...	www.paypalobj...	paypal.com	Login - PayPal
core.css	https://www.paypal....	/* PayPal Core Styl...	www.paypalobj...	paypal.com	Login - PayPal
pageLogin.css	https://www.paypal....	#xptTitle {display:no...	www.paypalobj...	paypal.com	Login - PayPal
flowHFR.css	https://www.paypal....	#resetPassword ul {li...	www.paypalobj...	paypal.com	Login - PayPal

Figure 3

Currently the default whitelist contains information from the login pages of the following sites: eBay, Paypal, Wells Fargo, Bank of America, Citibank, CapitalOne, Lloyds Bank, Regions Bank, HSBC, Alliance Bank, Internal Revenue Service, ANZ Bank, HM

Treasury, and GTE Federal Credit Union. Obviously this default list can easily be changed to accommodate users in specific regions such as the US, Canada, Europe, etc.

Toolbar operation

The toolbar performs several steps to decide which message to display to the user. The anti phishing algorithm in the toolbar is initiated when the requested webpage is downloaded completely. The toolbar first checks if the URL is in the whitelist. If the toolbar is in the whitelist then the detection algorithm stops and “whitelist” appears on the toolbar. If the website is not in the whitelist the next step is to check the URL for suspicion. If the URL is determined suspicious then the phishing warning is displayed along with “url detection” on the toolbar. Finally if the site passes the URL detection test then the CSS detection algorithm begins. If the CSS detection algorithm detects a CSS link or CSS content in the database then a phishing warning message is displayed to the user and either “css link” or “css content” is displayed in the toolbar. A flowchart of the toolbar operation is found in Figure 4.

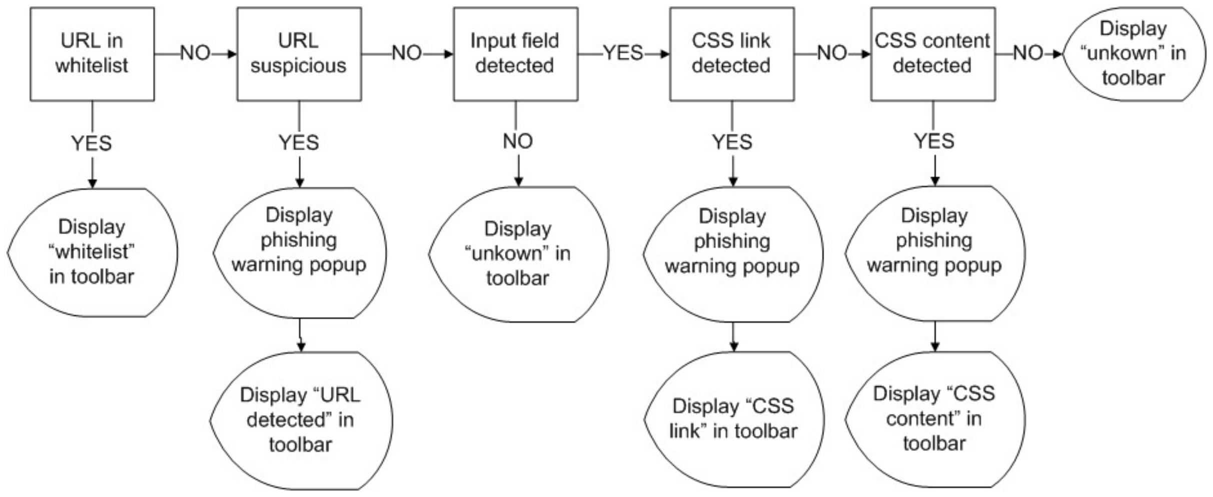

URL in whitelist
NO
URL suspicious
NO
Input field detected
YES
CSS link detected
NO
CSS content detected
NO
Display "unkown" in toolbar
YES
Display "whitelist" in toolbar
YES
Display phishing warning popup
Display "URL detected" in toolbar
NO
Display "unkown" in toolbar
YES
Display phishing warning popup
Display "CSS link" in toolbar
YES
Display phishing warning popup
Display "CSS content" in toolbar

Figure 4

Whitelist check

The first step of the toolbar is to perform a whitelist check. The URL is examined and if the hostname is found in the whitelist database then the detection algorithm stops and "whitelist" is displayed on the toolbar indicating to the user that the currently viewed site matches one in the whitelist.

Upon further experimentation it was found that the sub domains of sites in the whitelist should also be considered as legitimate sites. For example if the domain stored in the whitelist is "ebay.com" and the domain being checked is "signin.ebay.com" then the algorithm will mark the current site as "whitelist" because it is in the sub domain of a site in the whitelist. Currently sites ending in .com, .gov, .net, and .edu are checked for a sub domain. The sub domain check is limited to these extensions for now because of country code extension conflicts.

URL check

If the site is not in the whitelist then the URL check is performed. The URL check performs a series of checks on the URL of the website. Many of the URL detection techniques are implementations of previous phishing detection work such as Spoofguard [4] and the Google research paper [6]. No page content is examined during the URL check. Each detection will add one to the total score of the URL check. If the score is above a certain threshold the page is marked as phishing. The default threshold is two detections but in the future the user will have the option to change this field in the toolbar settings.

First the URL is converted to only lowercase characters to remove case sensitivity from the keyword detections. The first check on the URL is a check for hexadecimal characters. Every ASCII character has a corresponding hexadecimal equivalent number that computers can understand. These hexadecimal values are two digits and start with a '%' character in URLs. For example, one common use of hexadecimal notation in URLs is '%20' which represents a space. Both legitimate sites and phishers can use hexadecimal values in the URL for both good and malicious purposes. Often legitimate sites use hexadecimal values to replace punctuation such as a comma, question mark, colon, or space. However phishing sites often use hexadecimal values to replace letters and numbers so the actual URL of the website is hidden from the user. Because both legitimate and phishing sites can use hex values it is important to discriminate the good from the bad. This detection first searches the URL for the '%' character which indicates to the browser that the next two digits are hexadecimal values. Then it converts the hexadecimal digits to an integer value and determines if the integer value is a valid letter or number in the ASCII table. If the value is a letter or number then the URL is suspicious, the hex detection stops, and the total score for URL detection is increased by one. If the value is not a letter or number then the hex value is considered safe and the algorithm repeats the hexadecimal check until no more hexadecimal values are found in the URL. After the hexadecimal check, the URL is converted into its unescaped form. In other words if the phisher is using any escape characters such as hexadecimal values those are converted into their character form.

Secondly the URL is checked for suspicious characters. Originally this list included two characters adopted from Spoofguard, the "at symbol", '@', and the dash, '-'. However, experiments revealed that the dash character is not a successful indicator that a site is phishing and there are many legitimate sites that use the dash character. Therefore the dash character was removed from the detection. The '@' character is still a good indicator of a phishing site because the '@' character in a URL is used to send a username to a website. Phishers use the '@' symbol to trick a user into visiting a phishing site because any characters before the '@' symbol are not part of the actual URL of the site. Characters preceding the '@' are interpreted as a username by the web browser and therefore anything before the '@' character is ignored for loading a website. For example, a phishing site targeting eBay can use a URL of https://signin.ebay.com@phishingpage.com to give the appearance that the user is navigating to signin.ebay.com, while the browser actually navigates to phishingpage.com using the username "signin.ebay.com." If the @ character is found, the URL detection score is increased by one.

Next the URL is checked to see if it contains an IP address. Often phishing sites disguise the hostname of the site by using the IP address instead of the valid domain name. If a valid IP address is found in the URL then the URL detection score increases by one.

The URL is also checked for a non-standard port number. Sometimes phishing sites use a different port number than the standard port numbers. If the port number is not in the standard list of legitimate ports which includes http, https, ftp, gopher, and socks then the URL detection score increases by one. The standard port numbers are 21, 70, 80, 443, and 1080 [4].

Next the URL is checked against a list of common keywords. Google research indicates that the most common phishing keywords are: secure, account, webscr, login, signin, and confirm [6]. Banking was also added to the keyword list because of its high frequency in the experiments conducted. If a keyword is found the URL detection score increases by one.

The URL is also checked to see if it contains the company name of the top ten phishing targets found in the Google research paper. This list includes ebay, paypal, volksbank, wellsfargo, bankofamerica, privatebanking, hsbc, chase, amazon, banamax, and Barclays [6]. Fifth Third Bank was excluded because it's using the keyword "53" from its website http://www.53.com resulted in too many false detections because other websites may have the number 53 in the URL for other purposes. If a company name is found the URL detection score increases by one.

The URL is then checked to see if it has been reduced by a URL shortening tool such as those found at Tinyurl.com or Notlong.com. If "tinyurl.com" or "notlong.com" is found in the URL then the URL detection score is increased by one. Note that if a URL is shortened in this way then most likely the URL detection score will be exactly one because the parameters after "tinyurl.com" are typically random sequences of letters and numbers. It is however possible to create a custom alias such as something similar to "http://www.tinyurl.com/paypal_login" which yields a detection score of three because of the additional "paypal" and "login" keywords.

Next the URL is checked to see if it is using a common anonymizer or such as behidden.com or similar sites. The phishing score increases by one if an anonymizer is found in the URL.

Finally the URL is examined to see if the site is hosted on a free web hosting domain. Because of anonymity, ease of use, and free cost many phishing sites are hosted on free web hosting domains. The phishing score increases by one if a free web hosting domain is found. Obviously many personal websites are also hosted on free web hosting domains so it is important to combine this detection method with other URL detections.

Input field check

If the URL detection threshold is not reached then the CSS detection begins. The algorithm first searches for the HTML input field "<input" and then proceeds with the CSS detection if an input field is detected.

CSS link check

The CSS algorithm first searches for a CSS link. When one is found that is not a local link then the link's domain name is extracted and that value is searched against the list of CSS domain names in the whitelist. If a match is found, then the detection algorithm stops and the site is marked as phishing.

The CSS link check also extracts the last two parts of the domain in case there is a sub domain link being used. For example if the CSS domain "bankofamerica.com" is stored in the whitelist and the page being checked is using a CSS file from "sitekey.bankofamerica.com" the program will extract "bankofamerica.com" and recognize that the domain is in the whitelist. This was added because some phishing sites were using legacy or alternate CSS domain locations that were in a sub domain of the current CSS file's location. Again because of country code extension conflicts the extensions being checked for sub domains include .com, .gov, .net, and .edu.

CSS content check

If the CSS link detection is unsuccessful at finding a phishing site then the CSS content detection begins. Each CSS file found in the website is downloaded and then compared to the CSS content in the whitelist database. If half (50%) of the CSS file is found in one of the CSS files in the database then the detection algorithm stops and the site is marked as phishing.

Toolbar buttons

The toolbar currently has one button that allows interaction from the user. Additional buttons may be added in the future. The first button is "Add to whitelist" and allows the user to easily add the current site to the whitelist.

Add to whitelist button

When a user clicks the "Add to whitelist" button the currently viewed website is added to the whitelist. This allows the user to easily add new websites such as specific banks to the whitelist. Each CSS file in the current website is added to the whitelist as described in the Whitelist section of Chapter 7.

Toolbar notifications

The toolbar is designed to be inconspicuous and unobtrusive while still protecting the user from phishing attacks. The two methods used to communicate results to the user are a textbox in the toolbar itself and a pop-up warning if a site is suspected as phishing.

Toolbar textbox

The toolbar itself contains a textbox that displays the results of the phishing detection algorithm of the current page. There are five possible indicators displayed in the textbox of the toolbar: checking site, whitelist, URL detected, CSS link detected, and CSS content detected. An example of "checking site" being displayed is shown in Figure 2.

"Checking site" is displayed when the page content is still loading or the phishing detection algorithm is still working. On a high speed internet connection "checking site" is only displayed briefly while the browser loads the current website.

"Whitelist" is displayed when the current domain of the URL matches an entry already in the whitelist database. When a user sees "whitelist" in the toolbar, he\she can be confident that the current site is not a phishing site.

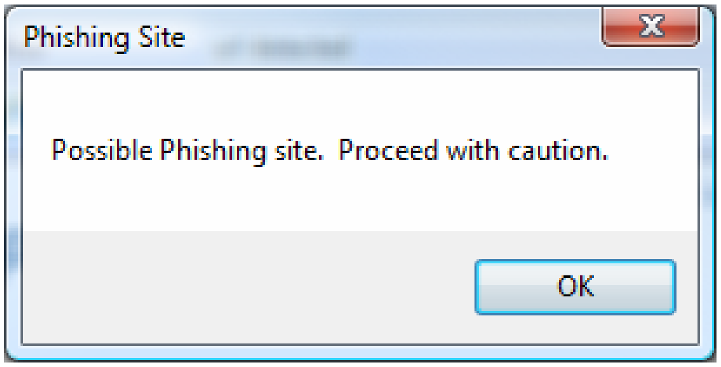
"URL detected" is displayed when the current site fails the URL detection algorithm. This informs the user that the site is marked phishing due to a suspicious URL.

"CSS link detected" is displayed when the current site fails the CSS link detection algorithm. This informs the user that the site is marked phishing due to a suspicious CSS link that matches one found in the whitelist.

"CSS content detected" is displayed when the current site fails the CSS content detection algorithm. This informs the user that the site is marked phishing because the current site's CSS content is found in the whitelist.

Popup warning

If one of the detection algorithms marks the site as phishing then the toolbar creates a popup warning to warn the user of the suspicious site. The warning shown in Figure 5 informs the user that the site is possibly a phishing site and to proceed with caution.



Possible Phishing site. Proceed with caution.

Figure 5

CHAPTER. 10 FUTURE WORK AND SUMMARY

Phishing detection methods are rapidly changing to keep up with new techniques used by phishers. Combating phishing is an ongoing battle that will probably never end much like the ongoing battle with spam emails.

Introduction

Because of the broad nature of the subject, this research provides only one implementation method of phishing detection which is a toolbar to detect phishing websites using Internet Explorer. There is still much work to be done in both the detection algorithms and the implementation of the toolbar. The toolbar implementation was designed mainly as an example of one implementation of the detection algorithms and requires more research, features, and testing before becoming a commercially acceptable product.

Updating the whitelist

Occasionally the whitelist will need to be updated so that it contains the latest changes from the websites in the whitelist. While drastic changes to the CSS file content are unexpected, one can expect web designers to change the CSS file content, file name, or file location. A weekly update of the whitelist should be sufficient to keep up-to-date with whitelist changes. Currently the whitelist for the toolbar does not automatically update or include an option to manually update.

Deleting items from whitelist

Sometimes one may accidentally add an item to the whitelist or wish to remove an item that is no longer needed. Currently there is no “remove from whitelist” feature on the toolbar. Ideally the toolbar would list the current items in the whitelist and allow the user to select the item to remove. The item would then be removed from the whitelist database. A smaller whitelist is ideal because the CSS detection algorithm will run faster if it has fewer entries to check in the whitelist.

Ideal default whitelist

Selecting an ideal default whitelist to include during installation of the toolbar is a challenge in its own right. One must try to cover a wide area of phishing targets while keeping the whitelist small enough so the detection algorithm can efficiently search the items in the database.

Updated free web hosting list

There are hundreds of free web hosting companies offering web space for anyone to take advantage of. Currently only a handful of free web hosting companies that were found in the phishing sites are included in the list used by the toolbar. The top free web hosting companies should be added to this list along with a periodic update that would add new popular web hosting services to the free web hosts list.

Updated anonymizer list

Similar to the free web host list, the list of anonymizers should periodically be updated. New anonymous web proxies should be added to the list of anonymizers used by the toolbar.

Current events

Phishing schemes commonly correlate with current events happening across the globe. Events such as an economic stimulus package or mergers of two banks create opportunities for phishers to prey upon unsuspecting people that have heard news about the phishing schemes. In a way the media increases the validity of the phishing site because people know that there is a legitimate change or current event happening. Ideally a centralized server would provide updates to the toolbar and add appropriate sites based on current events. For example the toolbar could automatically check for updates daily and if the President passed an economic stimulus package then the server would inform the toolbar to add the Internal Revenue Service (IRS) website to the whitelist. This would help provide phishing protection against malicious sites claiming that they are from the IRS.

Embedded style sheet detection

One of the weaknesses of the CSS detection algorithm is dependency on an external style sheet file. While the overwhelming majority of sites use an external style sheet, some sites use inline style sheets embedded directly in the HTML source code. The current CSS detection algorithm relies on an external style sheet so it will not detect styles embedded directly in the website source code. In the future the CSS detection algorithm should be able to sort through the HTML code and extract the style information in a similar manner that it currently extracts the style information from a style sheet file. The ability to detect styles in the HTML source code will decrease the number of undetected phishing sites (false negatives) while increasing the number of correctly identified phishing sites (true positive).

Suspicious items in hostname

If the hostname contains a suspicious item such as “ebay.com” but is not on the domain ebay.com then it could trigger an alarm as many phishing sites try to impersonate good domain names.

Frame detection

Currently the toolbar doesn’t detect content in page frames. While most sites don’t use frames and this isn’t a problem, there are some sites that use frames and would be undetected. Frames should be included in future detection algorithms.

Different country false detection

Currently the toolbar will detect other country sites that use the same CSS file as one in the whitelist as being phishing. For example www.paypal.de which is the PayPal site in Germany is detected as phishing because it is not in the whitelist, but it uses the same style sheet as www.paypal.com which is in the whitelist. One must currently add different country sites of the same company to the whitelist so there are no false positives. What’s interesting is that most of the links such as login, sign up, etc. on the PayPal site in German actually uses the domain www.paypal.com so it’s mainly the homepage that ends in .de that will trigger a false detection.

Sub domain check

Checking for a sub domain can be tricky because of the different country codes on the end of the URL. For example most of the time one would like to extract the last two parts of the domain in the URL such as “ebay.com” from the URL http://signin.ebay.com. However when country codes are added to the end of a URL one might need to extract the last three parts of the domain such as the Australian eBay site http://www.ebay.com.au/. For this reason only the following URL extensions are checked for subdomains: com, gov, edu, and net. Because the web is such a large domain future work in the sub domain check is needed.

The other problem with sub domains is if a free web hosting company is added to the whitelist then the sub domain of that company may be used to host various different websites. Obviously one doesn't want to include in the whitelist all websites hosted by a free web hosting company.

CSS content checking algorithm

The algorithm used to compare the CSS content of an unknown website with the CSS content of the sites in the whitelist needs further research and study. Right now it is a compromise between high similarity and speed. Using a return character and new line character to separate the sections into longer strings cuts down on the time needed to search the whitelist. However, it also opens the possibility of an attacker to apply subtle changes to the CSS file to avoid detection. Further research into efficient and fast string comparison algorithms is needed.

Summary

While generalizations about URLs can be made, it is difficult to conclude if a website is legit or phishing merely by the URL contents alone. One can however add to a phishing score if certain features are detected that are more likely found in phishing URLs rather than legitimate URLs. Also URLs can be checked against server information that contains a blacklist or ranks URLs based on popularity or search engine scores. The main problem with detecting a phishing site with a URL only method is that many phishing URLs appear normal.

URL detection alone is a good start because many phishing sites share certain URL characters or use free web hosting sites, but there are also problems with URL detection such as a phishing URL that doesn't appear suspicious or setting the detection threshold so low that it falsely detects many legitimate sites. Combining URL detection with another detection mechanism such as CSS detection is a good combination. The CSS detection algorithm worked nearly perfectly on eBay and PayPal phishing sites that used an external CSS file. The algorithm performed very well with detecting CSS links from banking sites as well. The biggest drawback to the CSS detection algorithm is its poor detection rate against sites that aren't using a CSS file. This is expected though because the algorithm relies on a CSS file. The other area that needs improvement is the CSS content checking for local CSS files. Although the algorithm worked great on the eBay and PayPal local CSS files, results from the banking experiments suggest that a smarter comparison algorithm is needed to compare a local CSS file's content to one listed in the whitelist.

Other benefits of the CSS algorithm include a low false positive rate against legitimate sites and the ability to detect new phishing before web browsers such as Microsoft Internet Explorer 7 detects them. Detecting zero day phishing attacks is the main goal of the CSS algorithm and these experiments show that although some improvements are needed, it does a good job at reaching that goal.

Combating phishing is a never-ending battle like most security related problems. New detection methods will be introduced and then the phishers will find ways around the detection methods. There is still much research needed in the area of phishing detection, but using keywords in the URL and detecting fraudulent sites based on CSS is a step in the right direction of detecting new phishing sites.

BIBLIOGRAPHY

[1] 3Sharp, LLC. Gone phishing: evaluating anti-phishing tools for Windows. September 2006.

[2] Abad, C. The economy of phishing: A survey of the operations of the phishing market. *First Monday,* volume 10, number 9 (September 2005). http://firstmonday.org/issues/issue10_9/abad/index.html

[3] Add-In Express, LTD. Add-in Express™ for Internet Explorer and Microsoft .net. http://www.add-in-express.com/docs/internet-explorer-toolbars.php. March 2009.

[4] Chou, N., Ledesma R., Teraguchi, Y., Boneh, D., and Mitchell, J. Client-side defense against web-based identity theft. In *11th Annual Network and Distributed System Security Symposium (NDSS '04),* February 2004.

[5] Dhamija, R., Tygar, J. D., and Hearst, M. 2006. Why phishing works. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Montréal, Québec, Canada, April 22 - 27, 2006).

[6] Garera, S., Provos, N., Rubin, A.D., Chew, M. A Frameword for Detection and Measurement of Phishing Attacks. In *Proceedings of the 2007 ACM workshop on Recurring malcode*, pages 1-8, 2007.

[7] Google. Corporate overview: technology overview. http://www.google.com/corporate/tech.html. May 2009.

[8] Internet Archive. Wayback Machine. http://www.archive.org/index.php October 2008.

[9] Johansen, Lasse. Levenshtein Distance Algorithm: C# Implementation. http://www.merriampark.com/ldcsharp.htm

[10] Likarish, P., Jung, E., Dunbar, D., Hansen T., and Hourcade, J.P. B-APT: Bayesian Anti-Phishing Toolbar. In *Communications, 2008. ICC '08. IEEE International Conference on Communications,* pages 1745-1749, 2008.

[11] Liu, W., Deng, X., Huang, G., and Fu, A. Y. An Antiphishing Strategy Based on Visual Similarity Assessment. In *IEEE Internet Computing 10, 2* (Mar. 2006), 58-65.

[12] McGrath, D. K. and Gupta, M. 2008. Behind phishing: an examination of phisher modi operandi. In *Proceedings of the 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats* (San Francisco, California, April 15, 2008).

[13] Microsoft. Microsoft phishing filter FAQ. https://phishingfilter.microsoft.com/PhishingFilterFaq.aspx October 2008.

[14] Mozilla. Firefox phishing and malware protection. http://www.mozilla.com/en-US/firefox/phishing-protection/ May 2009.

[15] Netcraft, LTD. Netcraft anti-phishing toolbar. http://toolbar.netcraft.com/. October 2008.

[16] Niederst, Jennifer. Web design in a nutshell, second edition. O'Reilly Media Inc. 2001. pp. 294-295.

[17] Phishtank. http://www.phishtank.com. October 2008 and February – March 2009.

[18] Ross, B., Jackson, C., Miyake, N., Boneh, D., and Mitchell, J. C. 2005. Stronger password authentication using browser extensions. In *Proceedings of the 14th Conference on USENIX Security Symposium - Volume 14* (Baltimore, MD, July 31 - August 05, 2005).

[19] TippingPoint. Phishing detection and prevention. http://www.planb-security.net/wp/503167-001_PhishingDetectionandPrevention.pdf.

[20] Zhang Yue, Egelman, S. and Hong J. Phinding Phish: Evaluating Anti-Phishing Tools. In *Proceedings of the 14th Annual Network & Distributed System Security Symposium (NDSS 2007)*, February 2007.

[21] Zhang, Y., Hong, J., Cranor, L. Cantina: A Content-Based Approach to Detecting Phishing Web Sites. In *Proceedings of the 16th international conference on World Wide Web*, pages 639 – 648, 2007.

ACKNOWLEDGEMENTS

I would like to thank those who helped me with various aspects of conducting research and the writing of this thesis. First and foremost, I thank Dr. Yong Guan for his expertise, support, and encouragement to tackle such a broad and difficult problem as phishing. He convinced me to extend a phishing research paper into a thesis topic. I also thank Dr. Jacobson and Dr. Schmidt for taking time out of their busy schedules to review this thesis and take part in the committee process. I would also like to thank doctoral candidate Yanlin Pang for her web based programming skills and expertise. Her help with downloading and sorting the phishing pages saved me numerous hours of work. I'm also thankful to Ames Engineering for taking me on board and having the confidence that I could quickly learn C# and contribute to their projects. Without my work experience there I would have been unfamiliar with Visual Studio and the Visual C# programming language which was used extensively in this research. My wife has been tremendously patient with me as I've spent several months researching and writing and I am grateful for her understanding and support. Finally I want to thank my mother for always believing in me and supporting me in whatever good interest I wanted to pursue.