

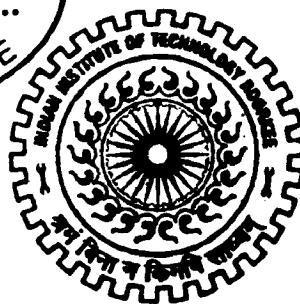
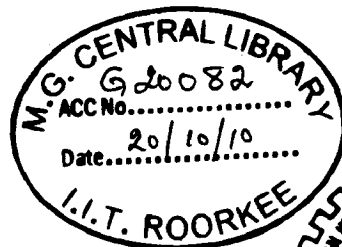
CLIENT-SIDE DEFENSE AGAINST PHISHING WITH PAGESAFE

A DISSERTATION

*Submitted in partial fulfillment of the
requirements for the award of the degree*
of
MASTER OF TECHNOLOGY
in
COMPUTER SCIENCE AND ENGINEERING

By

PANKAJ KUMAR SENGAR



**DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY ROORKEE
ROORKEE -247 667 (INDIA)
JUNE, 2010**

Candidate's Declaration

I hereby declare that the work being presented in the dissertation report titled “**Client-Side Defense Against Phishing With PageSafe**” in partial fulfillment of the requirement for the award of the degree of Master of Technology in Computer Science and Engineering, submitted in the Department of Electronics and Computer Engineering, Indian Institute of Technology Roorkee, is an authentic record of my own work carried out under the guidance of Dr Kuldeep Singh, in the Department of Electronics and Computer Engineering, Indian Institute of Technology Roorkee. I have not submitted the matter embodied in this dissertation report for the award of any other degree.

Dated: 16-06-10

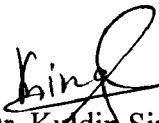
Place: IIT Roorkee.


(Pankaj Kumar Sengar)

Certificate

This is to certify that above statements made by the candidate are correct to the best of my knowledge and belief.

Dated: 16/6/10
Place: IIT Roorkee.


Dr. Kuldeep Singh, 16/6/10
Professor,
Department of Electronics
and Computer Engineering,
IIT Roorkee, Roorkee,
247667 (India)

ACKNOWLEDGMENTS

First of all and foremost, I would like to thank the Almighty, without whose grace, I would not be even here. I would like to express my deep sense of gratitude and indebtedness to my guide Dr. Kuldip Singh, for his invaluable guidance and constant encouragement throughout the dissertation. His zeal for getting the best out of his students helped me to perform above my par. I was able to complete this dissertation in time due to the constant motivation and support received from him.

I am also grateful to Mr. Sandeep Sood for helping me to clarify some basic and important concepts explored in this dissertation work. I would want to express thanks to my colleagues, Vijay Joshi, Ashish Kumar, Ankush Aggarwal and Laxmikant Sahu, for their “taken for granted” help with trivial matters, without which I am sure my work might have hit a dead end.

Last but not the least I would like to thank my family for their constant support, motivation and showing interest which sometimes lead to ray of hope in darkness.

(Pankaj Kumar Sengar)

ABSTRACT

Everyday, a number of attacks are launched with the aim of making web users believe that they are communicating with a trusted entity for the purpose of stealing account information, logon credentials, and identity information in general. These attacks, commonly known as “phishing attacks,” are most commonly initiated by sending out emails with links to spoofed websites that harvest information. Many anti-phishing schemes have recently been proposed in literature. Despite all those efforts, the threat of phishing attacks is not mitigated. Blacklist approaches where a list of phishing URLs is maintained by anti-phishing organizations, are partially effective. These schemes require the blacklist provider organizations to be much faster than phishers and effectiveness is based on the quality of blacklist otherwise phishing attack can cause damage. Another approach is to preserve secret information but to keep their private information could be irritating works for users. The effectiveness of private information preserving approaches is totally dependent on users. Solutions based on automatic classification have problems of false negatives and false positives.

This dissertation proposes PageSafe – an anti-phishing tool that prevents accesses to phishing sites through URL validation and also detects DNS poisoning attacks. PageSafe also examines the anomalies in web pages and uses a machine learning approach for automatic classification. PageSafe does not preserve any secret information and requires very less input from user. PageSafe performs automatic classification but by taking advantage of user assistance and external repositories, hence the number of false positives is reduced by a significant value. PageSafe is based on an approach opposite to blacklist approach removing the race between phishers and anti-phishing organizations. PageSafe maintains a whitelist of URLs with the mapping of corresponding IPs. This list is referenced first for resolving IP of a URL to protect user from DNS poisoning attacks. With PageSafe users help to decide whether or not a web page is legitimate. This report also presents an analysis on effectiveness of PageSafe based on an experiment done on a set of phishing pages and compares PageSafe with other available browser toolbars.

Table of Contents

CANDIDATE’S DECLARATION	i
CERTIFICATE	i
ACKNOWLEDGMENTS	ii
ABSTRACT	iii
TABLE OF CONTENTS	iv
LIST OF FIGURES	vii
LIST OF TABLES	viii
Chapter 1: Phishing Attack	1
1.1 Introduction	1
1.2 History Background.....	1
1.3 Current Status.....	1
1.4 Phishing Success Reasons.....	3
1.5 Motivation.....	3
1.6 Problem Statement	4
1.7 Organization of the Report	5
Chapter 2: Phishing Attack Vectors	6
2.1 Man-In-Middle Attacks.....	6
2.1.1 Transparent Proxies.....	6
2.1.2 DNS Cache Poisoning.....	6
2.1.3 Browser Proxy Configuration.....	6
2.2 URL Obfuscation Attacks.....	7
2.2.1 Bad Domains Names.....	7
2.2.2 Friendly Login URLs.	7
2.2.3 Third-Party Shortened URL.....	8
2.2.4 Host Name Obfuscation.....	8
2.2.5 URL Obfuscation.....	9
2.3 Cross-Site Scripting Attacks.....	9
2.4 Preset Session Attack	11
2.5 Hidden Attacks	12

2.5.1 Hidden Frames	12
2.5.2 Overriding Page Contents.....	12
2.5.3 Graphical Substitution.....	13
2.6 Observing Customer Data.....	13
2.6.1 Key Logging.....	13
2.6.2 Screen Grabbing.....	14
2.7 Client Side Vulnerabilities.....	14
2.8 Pharming.....	15
2.9 Spear Phishing.....	15
2.10 In-Session Phishing.....	16
Chapter 3: Defense Mechanisms.....	17
3.1 Client-Side.....	17
3.1.1 Desktop Protecting Agent.....	17
3.1.2 Email Sophistication.....	17
3.1.3 Browser Capabilities.....	18
3.1.4 Digitally Signed Emails.....	19
3.1.5 Customer Vigilance.....	20
3.2 Server-Side	20
3.2.1 Customer Awareness.....	20
3.2.2 Validating Official Communication.....	21
3.2.3 Custom Web Application Security.....	21
3.2.4 Strong Token-Based Authentication.....	22
3.3 Enterprise.....	23
3.3.1 Digitally Signed Email.....	23
3.3.2 Domain Monitoring.....	23
3.3.3 Mail Server Authentication.....	24
3.3.4 Gateway Services.....	25
3.3.5 Managed Services.....	25
Chapter 4: Design Details.....	27
4.1 PageSafe Model	27
4.2 Neural Network	27

4.3 Main Functionality	29
Chapter 5: Implementation Details.....	34
5.1 Background.....	34
5.2 PageSafe Working.....	34
5.3 Description Of Classes.....	34
5.4 Results.....	38
Chapter 6: Conclusions and Future Work.....	40
6.1 Conclusion.....	40
6.2 Future Work.....	40
REFERENCES.....	41

List of Figures

Figure 2.1: Cross Site Scripting Attack.....	10
Figure 2.2: Preset Session Attack.....	11
Figure 3.1: Strong Token Based Authentication.....	23
Figure 3.2: Mail Server Authentication-DNS querying of MX records.....	24
Figure 4.1: PageSafe Model.....	28
Figure 4.2: PageSafe Flow-Chart.....	33
Figure 5.1: Confusion matrix.....	40

List of Tables

Table 1.1: Anti-Phishing Working Group Report 2009.....	2
Table 6.1: Comparison With Available Browser Toolbars.....	40

Chapter 1: Phishing Attack

1.1 Introduction

The aim of phishing attacks is to steal user's secret information such as passwords, credit card number and pin etc. Phishing attackers use various tricks to convince user to visit bogus sites. Social engineering, such as phishing emails, or/and technical subterfuge, such as Trojan horses can be used for attack. Phishing sites mislead users to believe that they are legitimate sites and ask for the secret information.

1.2 History Background

The word "phishing" originally comes from the analogy that early Internet criminals used emails to "phish" for passwords and financial data from a sea of Internet user. The use of "ph" in the terminology is linked to popular hacker naming conventions such as "Phreaks" who were early hackers involved in the hacking of telephone systems.

The term came into light in the 1996 timeframe by hackers who were stealing America Online (AOL) accounts by scamming passwords from unsuspecting AOL users. The popularized first mention on the Internet of phishing was made in alt.2600 hacker newsgroup in January, 1996, however the term may have been used even earlier in the popular hacker newsletter "2600".

By 1996, hacked accounts were called "phish", and by 1997 phish were actively being traded between hackers as a form of electronic currency. There are instances whereby Phishers would routinely trade 10 working AOL phish for a piece of hacking software or stolen copyrighted applications and games.

The term Phishing covers not only obtaining user account details, but now includes access to all personal and financial data. Form emails, now it has been expanded into fake websites, installation of Trojan horse key-loggers and screen captures, and man-in-the-middle data proxies delivered through any electronic communication channel.

1.3 Current Status

The statistics show that phishing remained highly localized in certain Internet namespaces, and that some anti-phishing measures had noticeable impacts. Phishing is a damaging phenomenon

which results in many millions of dollars in losses. Phishing has always been attractive to criminals because it has low start-up costs and few barriers to entry.

According to Anti-phishing Working Group Report 2009 [1], by mid-2009, phishing is dominated by one player as never before the "Avalanche phishing operation". "Avalanche" is the name given to the world's most prolific phishing gang, and to the infrastructure it uses to host phishing sites. This criminal enterprise perfected a system for deploying mass-produced phishing sites, and for distributing malware that gives the gang additional capabilities for theft. Avalanche was responsible for two-thirds (66%) of all phishing attacks launched in the second half of 2009. There were at least 126,697 phishing attacks. This is more than double the 55,698 attacks we recorded in first half of 2009.

	2H2009	1H2009	2H2008	1H2008
Attacks	126,697	55,698	56,959	47,324
Phishing domain names	28,775	30,131	30,454	26,678
TLDs used	173	171	170	155
IP-based phish (unique IPs)	2,031	3,563	2,809	3,389
Maliciously registered domains	6,372	4,382	5,591	
IDN domains	12	13	10	52

Table 1.1: Anti-Phishing Working Group Report-2009

The average uptimes of phishing attacks have fallen steadily from 19 hours 30 minutes in first half of 2008 to 11 hours 44 minutes in second half of 2009. The longer a phishing attack remains active, the more money the victims and target institutions lose, and the more money the Phisher can make. The great majority of phishing continues to be concentrated in just a few namespaces. 76% of all phishing attacks occurred in just four TLDs: .COM, .EU, .NET, and .UK. Out of 28,775 domains used for phishing, 6,372 were maliciously registered domain registered by phisher. The remaining 22,403 domains used for phishing were "compromised" or hacked domains. Phishers still do not tend to abuse Internationalized Domain Names (IDNs).

1.4 Phishing Success Reasons

Successful phishers not only present a high credibility web presence to their victims but they create a so impressive presence also, which causes the victim to fail to recognize security measures installed in web browsers. An analysis on a phishing database addresses the question of why phishing works. The answer to this question is organized along three dimensions: lack of knowledge, visual deception, and lack of attention [2].

- Many users do not understand the meaning or the syntax of domain names and email headers. They cannot distinguish legitimate versus fraudulent URLs (e.g., they may think www.sbi-security.com belongs to www.sbi.com or cannot differentiate between www.paypal.com and www.paypai.com).
- Many users do not know that a closed padlock icon in the browser indicates that the page they are viewing was delivered securely by SSL. They do not know how to check the digital certificate.
- People have very little awareness about phishing. Some have misconceptions about which website features indicate security.
- User can be fooled by using an image of a legitimate hyperlink. The image itself serves as a hyperlink to a different rogue site.
- A common phishing technique is to place an illegitimate browser window on top of, or next to, a legitimate window. If they have the same look and feel, users may mistakenly believe that both windows are from the same source.
- If images and logos are copied perfectly, sometimes the only cues that are available to the user are the tone of the language, misspellings or other signs of unprofessional designer and the type and quantity of requested personal information.
- Security is often a secondary goal. When users are focused on their primary tasks, they may not notice security indicators or read warning messages.

1.5 Motivation

Today organizations are moving towards electronic transactions. The reason behind this is ease of business and global reach. Now a days each aspect of business, from order placement to

payment, is conducted through Internet in form of electronic transactions. Phishing is an obstacle to these electronic transactions.

Data suggest that some phishing attacks have convinced up to 5% of their recipients to provide sensitive information to spoofed websites [2]. About two million users gave information to spoofed websites resulting in direct losses of \$1.2 billion for U.S. banks and card issuers in 2003 [3]. Phishing attacks are increasing rapidly every year. The total attacks launched in 2009 were 182,395 while in 2008 the number was 104,283 [1].

1.6 Problem Statement

Phishing attacks are rapidly growing in number. Though a number of anti-phishing browser toolbars are available but unfortunately no one fully solves phishing attacks problem. For example, Antiphish [8] stores mapping of secret information with mapping to corresponding domain, SpoofGuard [9] examines domain name, images and links on web pages and raises alarm if the site has high probability of phishing, Microsoft and Google integrated [12] the blacklisted phishing domains into browser and browser warns user against these URLs, PwdHash [16] authenticates a user with domain specific password, DSS (Dynamic Security Skins) [17] provides trusted password window and uses SRP (secure remote password) protocol for authentication, PhishGuard [18] maintains trustlist containing mapping of trusted domains and corresponding IP addresses, ItrustPage [19] validates a URL through external information repositories. The problems with current toolbars are:

1. Most of the toolbars fail against DNS poisoning where DNS cache on local host is corrupted by installing some malicious software.
2. Few rely on automation but these have problems of false negatives (identifying a real website as phishing site) and false positives (identifying a phishing website as real website).
3. Most of the toolbars rely on users input even in some toolbars security is totally dependent on user inputs.
4. Some toolbars store secret information but it is not good to store secret information which is memorized by user.
5. Few depend on blacklist of phishing URLs but problem with this approach is that there is always a race between phisher and blacklist provider organization.

6. Few works in combination with server side where changes at server side are required.

This report presents PageSafe – an anti-phishing toolbar which works at client side without any change at server-side. PageSafe does not store any secret information like password etc. and requires very less input from user. Instead of querying with blacklist server, it validates URL by using external information repositories. PageSafe performs automatic classification but by taking advantages of user input so that the number of false positives are reduced by a significant value.

1.7 Organization of report

The report is organized as:

Chapter 2 discusses the phishing attack vectors, i.e. the various tricks used to launch phishing attack.

Chapter 3 explains the various security measures that can be adopted by users and organizations against phishing attack.

Chapter 4 describes the design principles. It discusses the various steps in working of proposed anti-phishing tool – PageSafe.

Chapter 5 explains technology used for implementation and details of implementation. It also presents performance analysis of PageSafe.

Chapter 6 concludes the dissertation work. It describes limitations and suggests future work for PageSafe.

Chapter 2: Phishing Attack Vectors

A number of methods are used by phisher to trick customer into doing something with their supplied page content. There are an ever increasing number of ways to do this. The common methods are explained below [3]:

2.1 Man-in-the-middle Attacks

In this type of attack, the attacker locates themselves between the customer and the real web-based application, and proxies all communications between the systems. This form of attack is successful for both HTTP and HTTPS communications. The customer connects to the attacker's server as if it was the real site, while the attacker's server makes a simultaneous connection to the real site. For man-in-the-middle attacks to be successful, the attacker must be able to direct the customer to their proxy server instead of the real server. This may be carried out through a number of methods:

2.1.1 Transparent Proxies

These proxies are located on route to the real server (e.g. corporate gateway or intermediary ISP), a transparent proxy service can intercept all data by forcing all outbound HTTP and HTTPS traffic through itself. In this transparent operation no configuration changes are required at the customer end.

2.1.2 DNS Cache Poisoning

"DNS Cache Poisoning" may be used to change normal traffic routing by injecting false IP addresses for key domain names so that all traffic destined for the legitimate URL goes to IP address of the attackers proxy server IP address.

2.1.3 Browser Proxy Configuration

By changing the browser proxy configuration options, an attacker can force all web traffic through to their nominated proxy server. This method is not transparent to the customer, and the customer may easily review their web browser settings to identify an offending proxy server. In

many cases browser proxy configuration changes setting up the attack will have been carried out in advance of receipt of the Phishing message.

2.2 URL Obfuscation Attacks

Phishing email convinces recipient to follow a hyperlink (URL) to the attacker's server, without them realizing that they have been duped. Unfortunately phishers have access to an increasingly large number of methods for obfuscating the final destination of the customer's web request. The most common methods of URL obfuscation include:

2.2.1 Bad Domain Names

The most common obfuscation method is through the registration and use of bad domain names. Consider the financial institute SBI with the registered domain sbi.com and the associated customer transactional site

`http://privatebanking.sbi.com`. The Phisher could set up a server using any of the following names to help obfuscate the real destination host:

- `http://privatebanking.sbi.com.ch`
- `http://sbi.privatebanking.com`
- `http://privatebanking.sbl.com`
- `http://privatebanking.sbi.members.com`

It is important to note that as domain registration organizations move to internationalize their services, it is possible to register domain names in other languages and their specific character sets. For example, the Cyrillic "o" looks identical to the standard ASCII "o" but can be used for different domain registration purposes. Even the standard ASCII character set allows for ambiguities such as upper-case "I" and lower-case "l".

2.2.2 Friendly Login URLs

Many common web browser implementations allow for complex URLs that can include authentication information such as a login name and password. In general the format is `URI://username:password@hostname/path`.

Phishers may substitute the username and password fields for details associated with the target organization. For example the following URL sets the username = mybank.com, password = e-banking and the destination hostname is evilsite.com.

`http://mybank.com:ebanking@evilsite.com/phishing/fakepage.htm`

This friendly login URL can successfully trick many customers into thinking that they are actually visiting the legitimate MyBank page. Because of its success, many current browser versions have dropped support for this URL encoding method.

2.2.3 Third-Party Shortened URL's

Due to the length and complexity of many web-based application URLs third-party organizations have sprung up offering free services designed to provide shorter URL's. Through a combination of social engineering and deliberately broken longs or incorrect URL's, Phishers may use these free services to obfuscate the true destination. Common free services include `http://smallurl.com` and `http://tinyurl.com`.

2.2.4 Host Name Obfuscation

Web browser communicates over the Internet by resolving URL to an IP address, such as 209.134.161.35, with the help of DNS servers. A Phisher may wish to use the IP address as part of a URL to obfuscate the host and possibly hide the destination from the end user. For example the following URL:

`http://mybank.com:ebanking@evilsite.com/phishing/fakepage.htm` could be obfuscated such as:

`http://mybank.com:ebanking@210.134.161.35/login.htm`

Different IP representations within an URL can be used to obscure the host destination. There other ways to encode the address other than the classic dotted-decimal format include:

- **Dword** - meaning double word because it consists essentially of two binary "words" of 16 bits; but it is expressed in decimal (base 10),
- **Octal** - address expressed in base 8, and
- **Hexadecimal** - address expressed in base 16.

For example, consider the URL `http://www.evilsite.com/`, resolving to 210.134.161.35. This can be interpreted as:

- **Decimal** – `http://210.134.161.35/`

- Dword – [http:// 3532038435/](http://3532038435/)
- Octal – <http://0322.0206.0241.0043/>
- Hexadecimal – <http://0xD2.0x86.0xA1.0x23/> or even <http://0xD286A123/>

2.2.5 URL Obfuscation

Internet software such as web browsers and email clients support local languages, most software supports alternate encoding systems for data. A Phisher can obfuscate the true nature of a supplied URL using one (or a mix) of these encoding schemes. Typical encoding schemes include:

- **Escape Encoding:** It is the accepted method of representing characters within a URL that may need special syntax handling to be correctly interpreted. This is done using a triplet sequence consists of the percentage character “%” followed by the two hexadecimal digits representing the octet code of the original character. For example, the ASCII character set represents a space with hexadecimal 20. Thus its URL-encoded representation is %20.
- **Unicode Encoding:** Unicode Encoding is a method of referencing and storing characters with multiple bytes by providing a unique reference number for every character no matter what the language or platform. For example %u0020 represents a space.
- **Inappropriate UTF-8 Encoding:** Unicode UTF-8 preserves the full ASCII character range. This great flexibility provides many opportunities for disguising standard characters in longer escape-encoded sequences. For example, the full stop character “.” may be represented as %2E, or %C0%AE, or %E0%80%AE, or %F0%80%80%AE, or %F8%80%80%80%AE, or even %FX%80%80%80%80%AE.
- **Multiple Encoding:** Many applications still incorrectly parse escape-encoded data multiple times. Consequently, Phishers may further obfuscate the URL information by encoding characters multiple times. For example, the back-slash “\” character may be encoded as %25 originally, but could be extended to: %255C, or %35C, or %%35%63, or %25%35%63, etc.

2.3 Cross-site Scripting Attacks

Cross-site scripting attacks (commonly referred to as CSS or XSS) make use of custom URL or code injection into a valid web-based application URL or imbedded data field. In general, these

CSS techniques are the result of poor web-application development processes. Phishers must make use of URL formatted attacks. Typical formats for CSS injection into valid URL's include:

- **Redirection URL:**

`http://mybank.com/ebanking?URL=http://evilsite.com/phishing/fakepage.htm`

- **Inline embedding of scripting content:**

`http://mybank.com/ebanking?page=1&client=<SCRIPT>evilcode...`

- **Loading scripting code from attacker server:**

`http://mybank.com/ebanking?page=1&response=evilsite.com%21evilcode.js&go=2.`

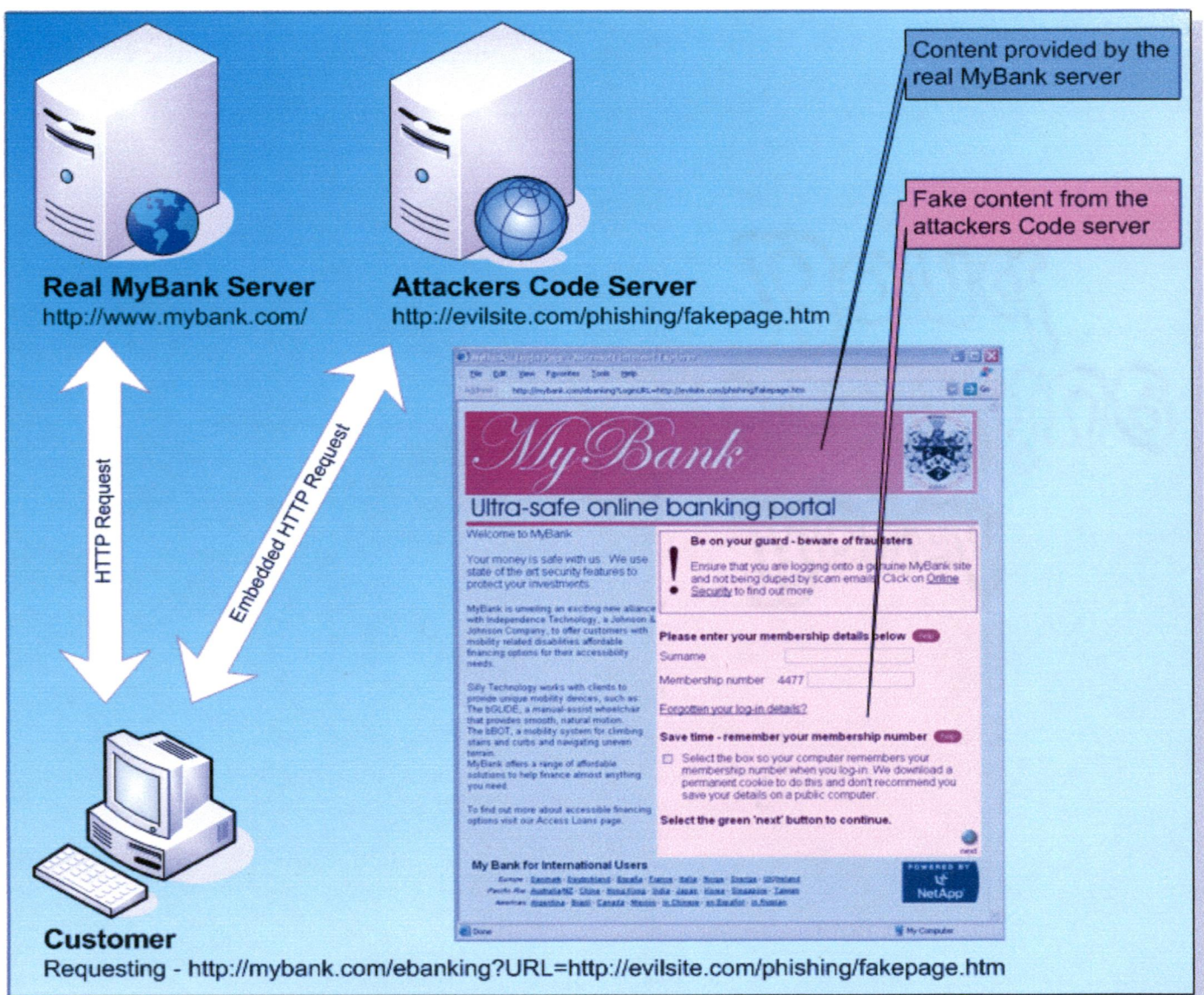


Figure 2.1: Cross Site Scripting Attack

2.4 Preset Session Attack

Since both HTTP and HTTPS are stateless protocols, web-based applications use cookies to track users through its pages and also manage access to resources that require authentication. These cookies contain Session Identifiers (SessionID's). Many web-based applications implement poor state management systems and will allow client connections to define a SessionID. The web application will track the user around the application using the preset SessionID, but will usually require the user to authenticate before allowing them access to "restricted" page content.

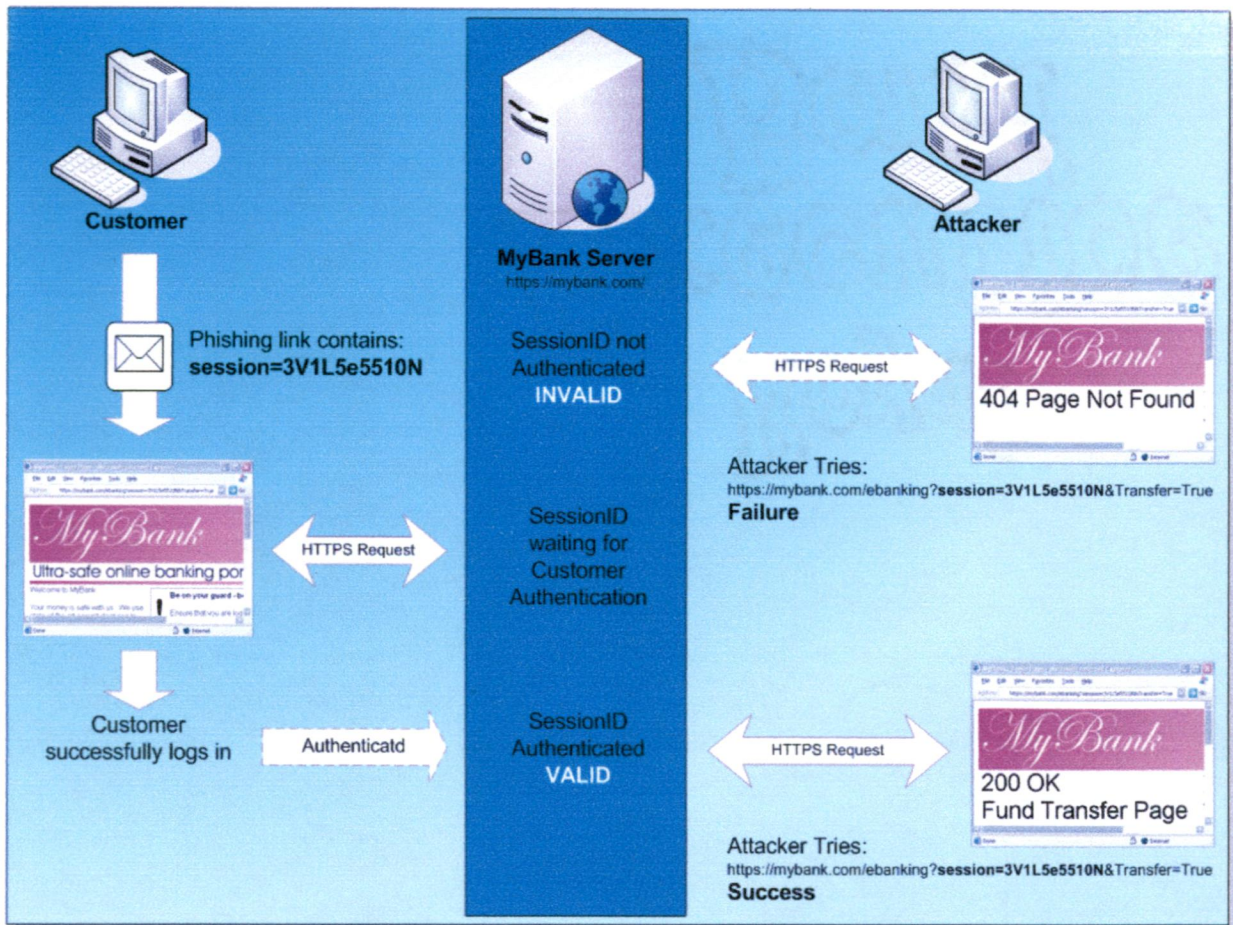


Figure 2.2: Preset session attack

In this class of attack the phishing message contains a web link to the real application server, but also contains a predefined SessionID field. The attackers system constantly polls the application server for a restricted page using the preset SessionID. Until a valid user authenticates against

this SessionID, the attacker will receive errors from the web-application server. The Phisher must wait until a message recipient follows the link and authenticates themselves using the SessionID. Once authenticated, the application server will allow any connection using the authorized SessionID to access restricted content. Therefore, the attacker can use the preset SessionID to access a restricted page and carryout his attack.

2.5 Hidden Attacks

HTML, DHTML and other scriptable code can be used to manipulate the display of the rendered information by web browsers. The attackers use these techniques to disguise fake content (in particular the source of the page content) as coming from the real site while fake copy of the site is hosted on the attackers own systems. The most common vectors include:

2.5.1 Hidden Frames

Hidden frame can be used to deliver additional content (e.g. overriding page content or graphical substitution), forcing the browser to display a fake padlock for non-SSL connection or to execute screen-grabbing and key-logging observation code. In the following example, two frames are defined. The first frame contains the legitimate site URL information, while the second frame – occupying 0% of the browser interface –references the Phisher’s chosen content.

```
<frameset rows="100%,*" framespacing="0">  
<frame name="real" src="http://mybank.com/" scrolling="auto">  
<frame name="hiddenContent" src="http://evilsite.com/bad.htm" scrolling="auto">  
</frameset>
```

2.5.2 Overriding Page Content

Several methods exist to override displayed content. One of the most popular methods of inserting fake content within a page is to use the DHTML function - DIV. The DIV function allows an attacker to place content into a “virtual container” that can be positioned to hide underlying content. An attacker can build a complete page on top of the real page.

2.5.3 Graphical Substitution

Web Browsers provide visual clues such as URL presented within the browsers URL field, the secure padlock representing an HTTPS encrypted connection, and the Zone of the page source. Browser scripting languages (such as JavaScript, VBScript and Java) can be used to overcome these clues by locating specially created graphics over these key areas with fake information. Therefore the attacker may prepare images for a range of common browsers and code their page in such a way that the appropriate images are always used. Graphical substitution combined with additional scripting code can be used to fake other browser functionality. Examples include:

- Implementing “right-click” functionality and menu access,
- Presenting false popup messages just as the real browser or web application would,
- Displaying fake SSL certificate details when reviewing page properties or security settings through the use of images.

2.6 Observing Customer Data

Phishers can employ key-loggers and screen-grabbers to observe confidential customer data as it is entered into a web-based application. This information is collected locally and typically retrieved through by attacker through the following different methods:

- Continuous streaming of data through sender/receiver pair. To do this, the attacker must often keep a connection open to the customer’s computer.
- Local collection and batching of information for upload to the attacker’s server. This may be done through protocols such as FTP, HTTP, SMTP, etc.
- Backdoor collection software allows the attacker to connect remotely to the customer’s machine and pull back the data as and when required.

2.6.1 Key-Logging

Key loggers observe and record all key presses by the customer when they enter their authentication information into the web-based application login pages. Key-loggers may be pre-compiled objects that will observe all key presses – regardless of application or context or they may be written in client-side scripting code to observe key presses within the context of the web browser.

2.6.2 Screen-Grabbing

Screen grabbers are designed to take a screen shot of data that has been entered into a web-based application. This functionality is used to overcome some of the more secure financial applications that have special features build-in to prevent against standard key-logging attacks. For example, in a recent Phishing attempt against Barclays, the attack used screen grabbing techniques to capture an image of the second-tier login process designed to prevent key-logging attempts.

2.7 Client-side Vulnerabilities

The web browsers, customers use to surf the web, are often vulnerable to a myriad of attacks. The more functionality built into the browser, the more likely there exists a vulnerability that can be exploited by an attacker. This, combined with the ability to install add-ons (such as Flash, RealPlayer and other embedded applications) means that there are many opportunities for attack.

Example : Microsoft Internet Explorer URL Mishandling

By inserting some special character in URL, a user would be redirected to the attackers server, but characters after would not be displayed in the browser URL field because of limited size. For example: 'http://www.mybank.com%01@evilsite.com/phishing/fakepage.htm'.

Example : RealPlayer/RealOne Browser Extension Heap Corruption

All popular web browsers offer support for RealPlayer and the automatic playing of media. By crafting a malformed .RA, .RM, .RV or .RMJ file it possible to cause heap corruption that can lead to execution of an attacker's arbitrary code. This code will run in the security context of the logged on user.

```
<OBJECT ID="RealOneActiveXObject" WIDTH=0 HEIGHT=0
CLASSID="CLSID:FDC7A535-4070-4B92-A0EA-D9994BCC0DC5"></OBJECT>
// Play a clip and show new status display
function clipPlay() {
    window.parent.external.PlayClip(
        "rtsp://evilsite.com/hackme.rm",
        "Title=Glorious Day|Artist name=Me Alone")
}
```

2.8 Pharming

Pharming [5] attacks redirect you to a hacker's site even when you type the URL of a real site into your browser. Pharming does not require that a user clicks on an email message. Pharming is a scamming practice in which malicious code is installed on a personal computer or server, misdirecting users to fraudulent web sites without their knowledge or consent. Pharming is also called "phishing without a lure". Pharmers typically redirect users to a spoofed website by poisoning company's hosts files or DNS (changing the mapping of a legitimate URL to fake site IP) so that requests for certain URLs return a bogus address and subsequent communications are then directed to a fake site. Other types of Pharming attacks involve Trojan horses, worms or other technologies that attack the browser address bar, thus redirecting the user to a fraudulent website when the user types in a legitimate address.

2.9 Spear Phishing

Spear phishing is an e-mail spoofing fraud attempt that targets a specific organization, seeking unauthorized access to confidential data. As with the e-mail messages used in regular phishing, spear phishing messages appear to come from a trusted source. These attacks use personal information about the victim to make the email appear more legitimate. The attacker finds a Web site for a targeted organization that supplies contact information for employees and other relevant data about the company. Using available details to make the message seem authentic, the perpetrator drafts an e-mail appearing to come from an individual who might reasonably request confidential information, such as a network administrator.

According to an article in the New York Times [6], spear phishing attempts are not typically initiated by "random hackers" but are more likely to be conducted by "sophisticated groups out for financial gain, trade secrets or military information."

Typically, a spear phisher requests user names and passwords or asks recipients to click on a link that will result in the user downloading spyware or other malicious programming. If a single employee falls for the spear phisher's ploy, the attacker can masquerade as that individual and gain access to sensitive data.

2.10 In Session Phishing

In-session phishing [7] is next generation of phishing attacks that has recently come in focus. In this phishing the specific focus is on what we call “in-session” attacks. In-session phishing attack takes place when the victim is logged in an online banking application and therefore is much more likely to succeed. The scenario for the attack is user logs in their online banking application to perform some tasks. User leaves a browser window open for some time and navigates to another website for some other work leaving this browser window open. After some time a popup appears, allegedly from the banking website, which asks the user to retype their username and password because the session has expired, or complete a customer satisfaction survey, or participate in a promotion, etc. As the user has recently logged in the banking website, he will not suspect this popup as fraudulent and follows the pop-up.

For this type of attack the attacker injects code into a compromised website which does not change the appearance of the website and does not download malware to the user’s PC. Therefore it is very hard to detect. This code only searches for online banking websites that visitors are currently logged onto, and present them with a popup that claims to be from the banking website they are logged on to. A method for finding the user who are logged in is discussed by hackers in 2006. In this method if the offensive website code is capable of loading the image (i.e. those images that accessible to logged-in users only),this confirms the user is logged on.

A vulnerability of JavaScript engine in most of browsers - Internet Explorer, Firefox, Safari, and Chrome – allows a website to check whether a user is currently logged onto another website. This vulnerability comes because of a specific JavaScript function which is use by majority of websites, including financial institutions, online retailers. When this function is called it leaves a temporary footprint on the computer and any other website can identify this footprint. Websites that use this function can be easily traced. The compromised website maintains a list of websites for checking logged on users. There is no limit of URL that a compromised website can check for logged in users. This only answers a question to browser whether the user in logged in or not and browser replies.

Chapter 3: Defense Mechanisms

Defense mechanisms against phishing can be deployed at three logical layers [4]:

1. The Client-side – this includes the user's PC.
2. The Server-side – this includes the businesses Internet visible systems and custom applications.
3. Enterprise Level – distributed technologies and third-party management services

3.1 Client-side

The client-side should be seen as representing the forefront of anti-phishing security. There are many solutions available for use at client-side to counter phishing attack.

3.1.1 Desktop Protection Agents

Desktop systems should be configured to use multiple desktop protection agents that are capable of performing the following services such as Anti-Virus, Firewall, IDS, anti-spam, anti-phish etc. Specific to phishing attack vectors, these solutions should provide the following functionality:

- The ability to detect and block installation of malicious software (such as Trojan horses, key-loggers, screen-grabbers and creating backdoors) through email attachments, file downloads, dynamic HTML and scripted content.
- The ability to identify common Spam delivery techniques.
- The ability to pull down the latest anti-virus and anti-spam signatures and apply them to the intercepting protection software as a daily activity.
- The ability to detect and block unauthorized out-bound and inbound connections from installed software or active processes and automatic block of delivery of sensitive information through these connections.
- The ability to detect and block common Spyware installations.

3.1.2 Email Sophistication

Email applications provide users an increasing level of functionality and sophistication. Most of this functionality is not required for day-to-day use particularly for Internet communication services. This unnecessary embedded (and often default) functionality is exploited by Phishing attacks. Most popular applications allow users to turn off the most dangerous functionality.

- **HTML-based Email:** HTML base emails provide ability to obfuscate the true destination of links, the ability to embed scripting elements and the automatic rendering of embedded (or linked) multimedia elements. HTML functionality must be disabled in all email client applications. Instead plain-text email representation should be used, and ideally the chosen font should be fixed. However, users should be prepared to receive some emails that appear to be “gobbldy-gook” due to textual formatting issues and probable HTML code inclusions. Some popular email clients will automatically remove the HTML code. Users should not use other email rendering options (such as Rich-text or Microsoft Word editors) as there are known security flaws with these formats which could also be exploited by Phishers.
- **Attachment Blocking:** Email applications must be capable of blocking “dangerous” attachments and preventing users from quickly executing or viewing attached content. Some popular email applications (such as Microsoft Outlook) maintain a list of “dangerous” attachment formats, and prevent users from opening them. Ideally, users should not be able to directly access email attachments (i.e. saving it locally) from within the email application. This applies to all attachment types (including Microsoft Word documents, multimedia files and binary files) as many of these file formats can contain malicious code capable of compromising the associated rendering application. In addition, by saving the file locally, local antivirus solutions are better able to inspect the file for viruses or other malicious content.

3.1.3 Browser Capabilities

Web browsers also offer extended functionality that can be exploited by phisher (often to a higher degree than email clients). Much of the sophistication is devoted to being a “jack of all trades”, and no single user can be expected to require the use of all this functionality. In particular, if the purpose of the web browser is to only browse Internet web services, a sophisticated web browser is not required. To help prevent many Phishing attack vectors, web browser users should:

- Disable all window pop-up functionality
- Disable Java runtime support
- Disable ActiveX support
- Disable all multimedia and auto-play/auto-execute extensions

- Prevent the storage of non-secure cookies
- Ensure that any downloads cannot be automatically run from the browser, and must instead be downloaded into a directory for anti-virus inspection

There are number of browser plug-ins available that can be added to the browsers toolbar and provide an active monitoring facility. These toolbars verify that the requested server host is not currently on a list of known phishing scams.

3.1.4 Digitally Signed Email

Digital signatures can be used to verify the integrity of the messages content – thereby identifying whether the message content has been altered during transit. A signed message consists of original message attached with its encrypted hash by sender's private key with date and time. It can be verified by receiver using sender's public key. Almost all popular email client applications support the signing and verification of signed email messages. It is recommended that users have a personal public/private key pair, enable automatic signing of emails by default and verify all signatures on received emails and be careful of unsigned or invalid signed messages. Therefore it is still possible for a Phisher to send forth an email with a spoofed address and digitally sign it with a key that they own.

S/MIME and PGP: There are currently two popular methods for providing digital signing. These are S/MIME and PGP. Most major Internet mail application vendor's ship products capable of using and understanding S/MIME, PGP/MIME, and OpenPGP signed mail. Although they offer similar services to email users, the two methods have very different formats. Users of one protocol can also communicate users of the other protocol. Key points for S/MIME and PGP are:

- S/MIME was originally developed by RSA Data Security, Inc. It is based on the PKCS #7 data format for the messages, and the X.509v3 format for certificates.
- PGP/MIME is based on PGP, which was developed by many individuals. The message and certificate formats were created from scratch and use simple binary encoding. OpenPGP is also based on PGP.
- S/MIME, PGP/MIME, and OpenPGP use MIME to structure their messages. They rely on the multipart/signed MIME type that is described in RFC 1847 for moving signed messages over the Internet.

3.1.5 Customer Vigilance: Customer’s vigilance plays an important role against phishing attack which involves inspecting content that is presented to them and verifying its authenticity. General vigilance includes:

- If you get an email that warns you, with little or no notice, that an account of yours will be shut down unless you reconfirm billing information, do not reply or click on the link in the email. Instead, contact the company cited in the email using a telephone number or Web site address you know to be genuine.
- Never respond to HTML email with embedded submission forms. Any information submitted via the email (even if it is legitimate) will be sent in clear text and could be observed.
- Avoid emailing personal and financial information and also check padlock icon on browser status bar before submitting financial information through a Web site.
- For sites that indicate they are secure, check the SSL certificate and ensure that it has been issued by a trusted certificate authority. SSL certificate information can be obtained by double-clicking on the “lock” icon or by right-clicking on a page and selecting properties.
- Review credit card and bank account statements as soon as you receive them to determine whether there are any unauthorized charges. If your statement is late by more than a couple of days, call your credit card company or bank to confirm your billing address and account balances.



3.2 Server Side

The interaction with server must be specified and implemented as a protocol that is either independent from other protocols or enhances an already existing protocol, such as the SSL/TLS protocol. By carrying out this work from the server-side, organizations can help to protect against a complex threat. The application at server side must provide:

3.2.1 Customer Awareness

It is important that organizations constantly spread awareness about dangers of phishing attack and appropriate actions to be taken. The key steps in helping to ensure customer awareness and vigilance include:

- Notifications on login pages about how the organization communicates with their customers.

Customers reaching the page should be prompted to think about the legitimacy of the email (or other communication) that drove them to the page.

- Provide clear links for customers to report phishing scams, or other possible fraudulent emails sent in the organizations name.
- Provide advice on how to verify the integrity, i.e. how to check the security settings, SSL, “padlock” and certificate signature of the page, decipher the URL line.

3.2.2 Validating Official Communications

There are a number of techniques an organization can apply to official communications but also these must take care of user’s technical ability and value of transactions. These techniques include:

- Use personalized emails for the specific recipient such as using of the customer’s name instead of common addressing (such as dear sir), or reference some other piece (not complete) of unique information shared between the customer.
- Clearly referencing the subject and date of the previous email with a new mail and providing sequence number to each mail.
- Use of digital certificates to sign messages.
- Providing a portal on the corporate website to allow customer to copy/paste their received message contents or URL to an interactive form for the verification of authenticity of the message and validation of URL.

3.2.3 Custom Web Application Security

Many popular Phishing attack vectors can be removed by applying robust content checking functions and implementing a few “personalization” security additions. For example, cross-site scripting vulnerabilities, resulting in highly successful attacks, can be detected by content checking.

Content Validation:

- Never directly trust data submitted by a user or other application components and always sanitize data before processing or storing it.
- Never present submitted data directly back to an application user without sanitizing it

- Ensure that all dangerous characters as constituting an executable language are replaced with their appropriate HTML safe versions. This decoding process may have to be carried out many times – until all encoded sequences have been removed.

Session Handling: Custom applications are used to implement session handling because of stateless nature of HTTP and HTTPS. To overcome a Preset Session attack, these applications must ensure:

- Never accept session information within a URL.
- Ensure that SessionID's have expiry time limits and that they are checked before use with each client request.
- Submission of an invalid SessionID (i.e. expired, revoked, or extended beyond it's life), should result in a server side redirection to the login page with a new SessionID.
- Never keep a SessionID that was initially provided over HTTP after the customer has logged in over a secure connection (i.e. HTTPS) and after authenticating, the customer should always be issued a new SessionID.

URL Qualification: For some applications redirection is necessary. For such application greater care must be taken in qualifying the nature of the link before redirection. These application must ensure:

- Do not reference redirection URL's or alternative file paths directly within the browsers URL path (e.g. <http://mybank.com/redirect.aspx?URL=secure.mybank.com>)
- Always maintain a valid “approved” list of redirection URL's.
- Never allow customers to supply their own URL's.
- Never allow IP addresses to be used in URL information.

3.2.4 Strong Token-based Authentication

Using single-use time-based time passwords is another way for achieving strong authentication. These passwords are generated by external systems. A password cannot be repeated for authentication. These systems, often referred to as token based authentication systems, may be based on physical devices (such as key-fobs or calculators) or software. Customers of the legitimate web-based application may use a physical token such as a smartcard or calculator to provide a single-use or time-dependant password.

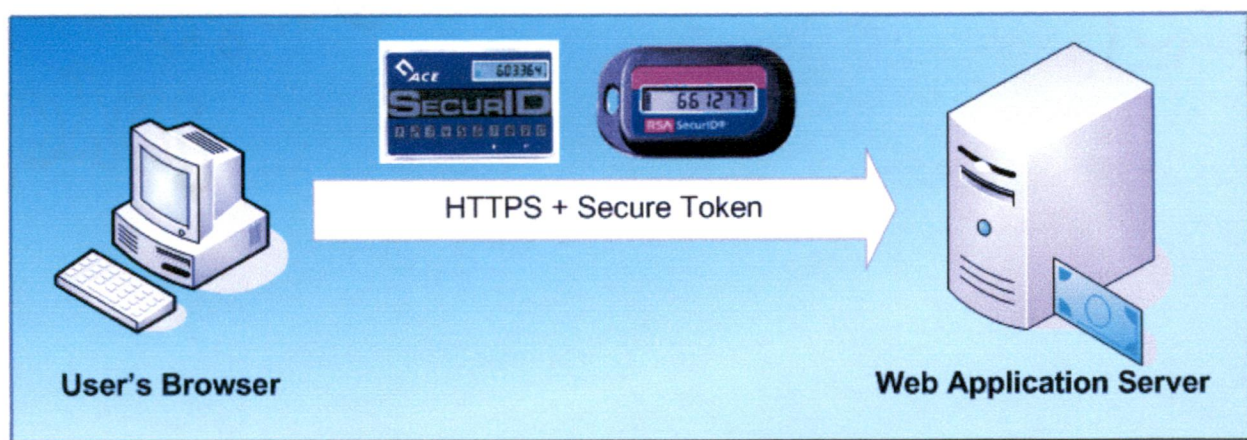


Figure 3.1: Strong Token-Based Authentication

By reducing the likelihood of authentication details being shared between multiple organizations, there is less opportunities for an attacker to achieve an identity theft.

3.3 Enterprise

There is no single silver bullet solution exists at enterprise level against phishing. Enterprise security solutions work in combination with client-side and server-side security mechanisms, offering considerable defense against phishing and other threats. Key steps to anti-phishing enterprise-level security include:

3.3.1 Digitally Signed Email

One tool available to senders of legitimate emails to aid the recipient in this process is to digitally sign their messages, allowing the recipient to establish a level of comfort that the message actually came from the indicated sender. An enterprise must configure their receiving email servers to automatically validate digitally signed emails before they reach the recipient. In addition, the enterprise email server can be configured to always sign outbound email.

3.3.2 Domain Monitoring

Organizations must carefully monitor the expiry and renewal of existing corporate domains and the registration of similarly named domains. There are numerous agencies that allow the registration of domains previously owned by an organization that have not been renewed. Many organizations own multiple domains that must be renewed in a timely fashion. Failure will result in a loss of service, (i.e. domain name lookup) or may be purchased by a third-party. It is a

simple process for someone to register a domain name through any domain registrar, anywhere in the world. A phisher can register domain names that may infringe upon an organizations trademark or used to trick customers into believing that they have reached a legitimate host. For example, assuming the organizations name is “Paypal” and their normal website is www.paypal.com, the organization should keep a watchful eye out for hyphenated names (i.e www.pay-pal.com),country specific (i.e. www.paypal.com.au), legitimate possibilities (i.e. www.secure-paypal.com),or mixed wording (i.e. www.paypalglobal.com) etc. Now commercial services available that help organizations monitor the domain name service and alert when potentially threatening new domains are registered.

3.3.3 Mail Server Authentication

The sender’s mail server is authenticated by the receiving mail server by reverse resolution of Domain information to a specific IP address or range. If the senders IP address is not an authorized address for the email domain, the email is dropped by the receiving mail server.

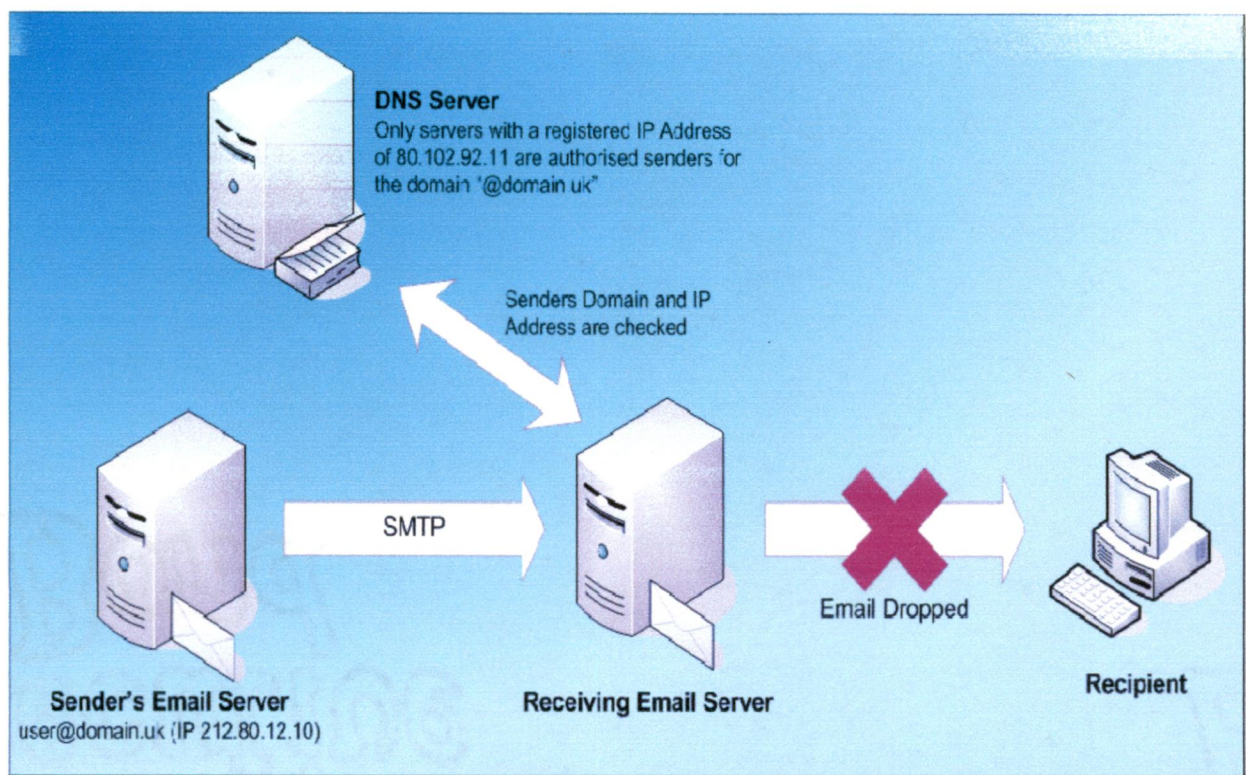


Figure 3.2: Mail Server Authentication-DNS querying of MX records

Another solution is to use Secure SMTP, email transport is conducted over an encrypted SSL/TLS link. When the sender mail server connects to the recipient mail server, certificates are exchanged and an encrypted link is established. Invalid and missing certificates will prevent a secure connection and do not allow delivery of emails.

3.3.4 Gateway Services

A gateway protection services, located on route to Internet, can monitor and control both inbound and outbound communications. These services can be used to identify malicious Phishing content; whether it is in email or other communication streams.

Typical enterprise-level gateway services include:

- Gateway Anti-Virus Scanning is used to detect viruses, malicious scripting code and binary attachments that contain Trojan horse software.
- Gateway Anti-Spam Filtering is a rule-based inspection of email content for key phrases (such as Viagra) and bad words, typically used to identify common spam, but also capable of stopping many forms of phishing attack that are designed to look like legitimate spam.
- Gateway Content Filtering performs inspection of many types of communication methods (e.g. email, IM, AOL, HTTP, FTP) for bad content or requests.
- Proxy Services provide protection against inbound attacks through the use of network address translation. Good protection against common information leakage of internal network configurations.

3.3.5 Managed Services

Managed services are becoming an essential component of preventing e-mail security risks. Managed services have the ability to analyze email messages delivered at a global level, and identify common threads between malicious emails. Security needs vary for different-size companies. Rather than trusting network security to an internal staff, hiring a security firm to manage the process every day is a growing trend. For instance, an organization may only receive 5 or 6 carefully disguised phishing emails with minor content changes not enough to trigger an anti-spam response while the managed service provider has spotted several thousand of the same style emails which triggers the anti-spam/anti-phishing blocking processes. When dealing with phishing and spam, email volume is a key component in identifying malicious activities.

Managed security software allows the security firm to check corporate computer systems and then keep them intruder-free through remote-access security checks and daily traffic monitoring. Managed security services also provide active web monitoring. Managed service providers may deploy agent-based 'bots to monitor URL's and web content from remote sites, actively searching for all instances of an organizations logo, trademark, or unique web content. The subscribing organization institution provides a "white list" of authorized users of logo, trademark, and unique web content to the service provider. When the 'bots detect unauthorized deployments or instances of the logos, trademarks, or other web content, remediation actions may be taken by the subscriber.

Chapter 4: Design Details

4.1 PageSafe Model

PageSafe performs automatic classification but does not completely rely on automation to detect phishing. Instead, PageSafe asks for user input and also examines anomalies in web page to perform automatic classification to decide on the legitimacy of a web page. It uses external information repositories on the Internet to help the user with decision-making. PageSafe prevents accesses to phishing sites and warns against DNS poisoning attacks. PageSafe maintains a Whitelist—a list of domains with mapping to corresponding IP addresses. Whitelist is encrypted by a master password to protect it from corruption through malicious softwares. It maintains a dynamic whitelist containing domains with mapping to corresponding IP addresses. It considers that phishing sites are short lived and only allows those sites that are not short lived. This section presents the PageSafe model for preventing accesses to Phishing sites as well as Pharming detection. PageSafe uses artificial neural network approach for automatic classification after identifying anomalies in a web page.

4.1.1 Neural Networks

ANNs develop their own solutions from examples for a class of problems. Artificial neural network consists of a collection of processing elements (neurons) that are highly interconnected and transform a set of inputs to a set of desired outputs. The neurons process information parallelly and collectively within the structure of the network. The result of the transformation is determined by the characteristics of the elements and the weights associated with the interconnections among them. A neural network conducts an analysis of the information and provides a probability estimate that it matches with the data it has been trained to recognize. The neural network gains the experience initially by training the system with both the input and output of the desired problem. The network configuration is refined until satisfactory results are obtained. Artificial neural networks have different types of architectures, which consequently require different types of algorithms. In this thesis work Scaled Conjugate Gradient Backpropagation (traincsg) algorithm is used for training neural network.

Scaled Conjugate Gradient Backpropagation Algorithm: The scaled conjugate gradient algorithm [20] is an implementation of avoiding the complicated line search procedure of

conventional conjugate gradient algorithm (CGA). According to the SCGA, the Hessian matrix is approximated by:

$$E''(w_k)p_k = \frac{E'(w_k + \sigma_k p_k) - E'(w_k)}{\sigma_k} + \lambda_k p_k$$

where E' and E'' are the first and second derivative information of global error function $E(w_k)$. The other terms p_k , σ_k and λ_k represent the weights, search direction, parameter controlling the change in weight for second derivative approximation and parameter for regulating the indefiniteness of the Hessian.

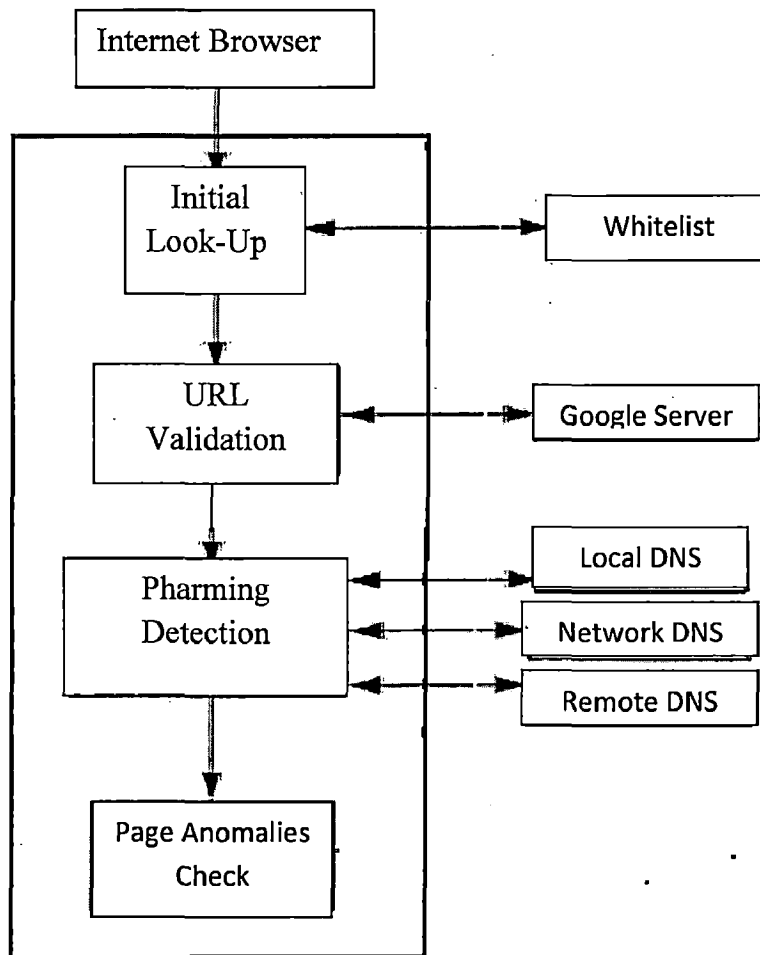


Figure 4.1: PageSafe Model

4.1.2 Main Functionality

PageSafe has four modules-Initial look up, URL validation, Pharming detection and Page anomalies check. When a user requests a URL, these modules becomes active and performs different functions:

Initial Look-Up Module: The Initial look-up module looks up the domains of URL in whitelist and picks up the corresponding IP from the whitelist if domain is found in whitelist. URL is passed to URL validation module if URL is not found on whitelist.

URL Validation Module: URL validation module issues a search query to Google for The URL. If the top 10 search results contain the URL, then it infers that the URL is legitimate. The average life time of a phishing site is 5-6 days [14]. The fact that the site appears in the top 10 search results means that the Google crawler indexed the site, and that the site is not short-lived. Sometimes, the user reaches a web page by navigating to it from the Google search page. These domains are automatically added to whitelist after performing pharming check. If URL is not found in top 10 results, PageSafe involves users to specify the search term for the web page they intend to visit. PageSafe performs a Google Search and provides the search results to users. Users can choose a URL from the search results. After URL validation it URL is passed to Pharming detection module.

Pharming Detection Module: Malicious softwares attached with emails or web pages can cause DNS poisoning or Pharming. There are 3 entities involved in resolving a URL to IP:

Local DNS: means a local host file which is firstly referenced to resolve a web address to a specific IP address.

Network DNS: means DNS server inside an organization. If the Local DNS doesn't know the corresponding IP address, the Network DNS is asked.

Remote DNS: means the DNS servers of ISPs. We use these to check the Network DNS Pharming.

Pharming detection module compares the IP addresses from Local DNS and Network DNS with the IP address form remote DNS for the requested URL. If all are not same then pharming is detected and alert is made to user. We assume Remote DNSs such as ISP DNS are highly

secured and are not contaminated. If no pharming is detected the URL is added to whitelist with mapping to its IP and web page is retrieved from the server. After retrieving the web page Page Anomalies Check module examines the page.

Page Anomalies Check Module: This module first selects high frequency words appearing in a web page and removes stop words from the set. Suppose, $S=\{s_1, s_2, s_3, \dots\}$ is a set found after removing stop words from the set of high frequency words. It performs the checks in two rounds. In first round it performs:

1. Similarity Check: The attacker uses a URL that seems similar to actual legitimate URL. This finds out a URL in whitelist with maximum similarity to requested URL. Ratcliff/Obershelp pattern matching algorithm [14] is used for matching substrings. The algorithm works by examining two strings passed to it and locating the largest group of characters in common. The algorithm uses this group of characters as an anchor between the two strings. The algorithm then places any group of characters found to the left or the right of the anchor on a stack for further examination. This procedure is repeated for all substrings on the stack until there is nothing left to examine. The algorithm calculates the score returned as the number of characters found in common divided by the total number of characters in the string. Alternatively this algorithm finds out the percentage of legitimate URL in current page URL. For example, suppose you want to compare the similarity between the word `https://www.sbi.com` and `http://www.sbi-members.com`. The largest common group of characters that the algorithm would find is `://www.sbi`. The two sub groups remaining to the left are `https` and `http`, and to the right are `.com` and `-members.com`. The algorithm places both of these string sections groups on the stack to be examined, and advances the current score to 10, equal to number of characters in common substrings. `.com` and `-members.com` are next to come off of the stack and are then examined. The algorithm finds 4 character in common `.com`. The score is advanced to 14. Next, the algorithm pulls `https` and `http` off of the stack. The largest common substring found is `http`. The algorithm advances the score to 18. There is nothing to the left now, stack is now empty and the algorithm ready to return the similarity value found. There was a score of 18 out of a total of 19. The result means that the two strings were 95 percent alike.

2. Title check: The contents of title tag in HTML appear on the top of browser window. Attacker uses the name of real site to make user to believe that he is visiting a legitimate website. The contents of title, after removing stop words, are compared with the domains in secure list. Appearance of title contents in some secure list domains causes phishing suspiciousness. If both the checks indicate the same domain in whitelist, user is alerted for phishing warning. Otherwise next round is executed.

In second round it checks anomalies in web page and applies the following rules:

1. URL Check : A web page's URL (Uniform Resource Location) is unique in the cyberspace. For a regular web site, its identity is usually part of its domain name. For a phishing site, its true URL is usually similar but different from its claimed identity. Specifically, we consider three cases:

- a. For URL address L , no s_i is a substring of L for all $0 \leq i \leq k$, or the domain name looks obscured, e.g `http://www.paypal.com@123.123.123.123`, or `http://www.paypal.com.secure.login.cmd.path.hotelielsi.com/cgi.bin/`, $F_1=1$,
- b. If one page only uses the IP address, $F_1=0$,
- c. Otherwise , $F_1=-1$.

2. Link Check: Anchors in a normal web page usually point to pages in the same domain. For phishing pages, there are possible abnormalities listed below. In the following, let A_a be the total number of anchors in page.

- a. **Nil anchor:** An anchor is called a nil anchor if it points to nowhere. Examples are ``, ``, ``, etc.
- b. **Foreign anchor:** An anchor is called foreign anchor if it points to foreign domain. The attacker does not want to create the complete website. The percentage of nil and foreign anchors in a page reflects the degree of suspiciousness. Let The higher the percentage, the more likely that page is a phishing page. $A_{n/f}$ be the number of nil and foreign anchors in web page. F_2 is assigned as follows:

$$F_2 = \begin{cases} 0 & A_a = 0 \\ A_{n/f}/A_a & A_{n/f} > 0 \\ -1 & \text{otherwise} \end{cases}$$

3. Request URL check: Web pages are object rich, containing numerous objects including images, CSS files, scripts etc., a large percent of objects are loaded from its own domain. Only a small portion of them are from foreign domains. While in phishing pages, most objects are copied or loaded from the real sites since the attackers intend to reduce their cost of faking. We observe that more request URLs in page indicates a higher probability of page being faked.

$$F_3 = \begin{cases} 0 & R_a = 0 \\ R_f/R_a & R_f > 0 \\ -1 & \text{otherwise} \end{cases}$$

where R_f is the number of request URLs to foreign domains.

4. Form handler check: The ultimate goal of phishing attacks is to steal user's private information, such as user name and password. Phishing pages often contain forms requesting user inputs. It shows where the data is to be sent. However, the handler of such a form in a phishing page usually refers to the real site or simply is void or to some foreign domain. We set $F_4 = 1$, if there is an occurrence of any void handler (e.g. `<form action="#">`, `<form action="about: blank">`, `<form action="javascript:true">`), or any handler referring to a foreign domain. If there is no handler in page, $F_4=0$ and $F_4= -1$ for other cases.

5. SSL Check: In a SSL transaction, the web client usually requests the server to present a public key certificate. For a legitimate web site, the presented certificate contains identity relevant information, e.g. the Distinguished Name (DN). Moreover, a certificate for web usage usually defines its serving URL explicitly. A phishing site may choose to use the same certificate as its victim's one. Otherwise, its own certificate would not match the identity it attempts to impersonate. So, $F_5 = 1$ if one of the claimed identities does not appear in the certificate attached to page or the URL specified in the certificate is different from L; $F_5 = 0$ if the SSL is not applied; and $F_5 = -1$ for other cases.

6. Hidden fields: Hidden fields can be used to steel information. These fields are not visible to user but can be used to hide information such as username, passwords etc using java scripts. A large number of hidden fields in a page lead suspiciousness of page being fake. F_6 is the number of hidden fields in a webpage.

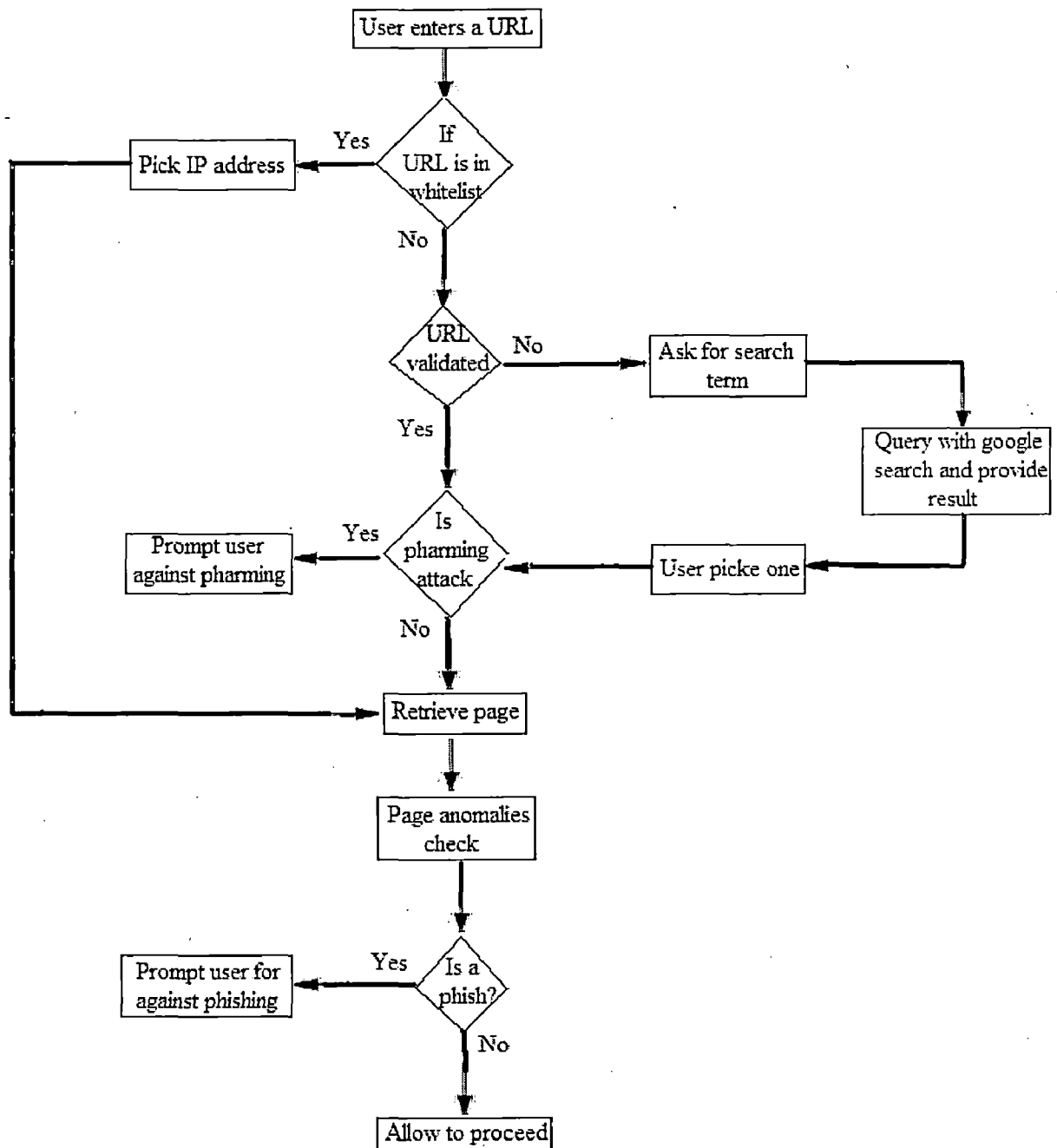


Figure 4.2: PageSafe FlowChart

After calculating values of all rules, it applies these values to a trained neural network. This neural network was trained on a set of web pages containing phishing as well as legitimate web pages. User is allowed to proceed or prompted against phishing attack based on decision.

Chapter 5: Implementation Details

5.1 Background

PageSafe is written in C++ and interfaces with the Internet Explorer browser through Microsoft's COM interface. For the development of PageSafe Microsoft's Platform SDK and the Microsoft WTL classes have been used. It was also needed to set up Visual Studio to access these libraries and header files.

5.2 PageSafe Working

There are three buttons. The first, the Settings Button, brings up the Settings dialog (displays Whitelist with edit options). The second, the Status Button, displays the status (RED color for phishing) and also brings up a status message. The third, the Reset Button clears the whitelist. Whenever the browser navigates to a new page, PageSafe starts performing the various checks in two rounds. A new site is flagged if any check fails. If a site is flagged, a red light will be displayed and status message is appeared. If you select, in the Settings dialog, to be stopped before visiting a suspicious site, a flagged site will trigger a pop-up window that will warn you either before or after the browser navigates to a new page, depending on whether the site can be flagged with only the first round of checks, which occurs before navigation. The first round of checks occurs before the browser attempts to navigate to a new page. At this point, the only information available to the browser (and the PageSafe Toolbar) is the URL that it will attempt to navigate to. The two checks executed in the first round are the Pharming Detection & URL Validation. If the checks in first round fail, you will be warned before the page is loaded and be given the option not to navigate to the attempted page. Second round of checks, is executed after requested web page has been loaded to browser. In this round PageSafe performs Similarity check, Title check, URL Check, Link check, Requesting URL check, Form Handler check, SSL check, Hidden fields check.

5.3 Description of classes

AlertBar

This is the primary class required to implement an Internet Explorer toolbar. AlertBar is a COM component (the only COM component in PageSafe) that extends the IDeskBand interface,

among others; this interface exposes the necessary methods. AlertBar is where most of the interesting work is done, and will be discussed in further detail below, after the rest of the classes involved in PageSafe are discussed. Two window classes must be implemented in order to define the appearance and user interaction of the bar. Both implement the CWindowImpl interface to do this. Upon initialization of AlertBar, a new instance of ReflectionWnd is created and registered, which creates a new instance of UWToolBar.

ReflectionWnd

This window class implements a transparent window that sits on top of the toolbar and takes user messages (like mouse clicks) and reflects them to the UWToolBar, which defines the buttons that respond to these messages. When this class is created by AlertBar, it creates and registers a new UWToolBar. From AlertBar, accessing UWToolBar is done through this class. ReflectionWnd is able to pass messages to its instance of UWToolBar by registering it in a message map constructed with macros defined in the Microsoft ATL.

UWToolBar

This window class defines the buttons and appearance of the toolbar. It receives messages from the ReflectionWnd that created it. While the toolbar is running, UWToolBar maintains and updates whitelist. AlertBar, which uses this data, must access UWToolBar each time it wants to retrieve this data. Additionally, UWToolBar exports methods to set status of the page. That is, other classes, particularly AlertBar, can change the Status Button as well as the message that appears on the Status Dialog. Upon initialization of UWToolBar, two buttons are created and registered such that they fire IDM_OPTIONS and IDM_STATUSBUTTON events, which are meant to trigger the Settings (whitelist) and Page Status, respectively. When a user clicks on one of these buttons, the message is received by ReflectionWnd and is passed to UWToolBar, which fires OnCommand. OnCommand interprets the command and, if the event is an IDM_OPTIONS or an IDM_STATUSBUTTON, the appropriate dialog is initialized and fired. In the case where the Settings Dialog is fired, a new ConfigDlg is initialized displaying whitelist with edit options, and the UWToolBar updates the whitelist based on the result that ConfigDlg returns when the dialog terminates. Similarly, the Status Dialog is initialized with a string that represents the status of the current page, passed to UWToolBar from AlertBar whenever the browser navigates to a new page.

TestDlg

This class exists only to handle the Settings Dialog. When UWToolBar opens a new Settings Dialog, which occurs each time the Settings button is pressed, it creates a new instance of TestDlg and sets the initial state of the dialog box by initializing TestDlg's instance variables. While the Settings Dialog is running, TestDlg intercepts all the events it fires, like clicking on a radio button, and continually stores the state of the dialog. When the user terminates the dialog, UWToolBar extracts the values of the instance variables from ConfigDlg and saves them.

StatusDlg

This class exists only to handle the Status Dialog. It is very similar to ConfigDlg, but much simpler. This dialog only displays the warnings associated with the current page, so StatusDlg needs to interact with it only upon initialization to set the status message.

AlertBar

With the previous classes in place, AlertBar can retrieve the current state of the user settings and set the status of the current page. When the toolbar is not running, the user settings are saved in the registry. They are loaded in LoadRegVals() and saved in SaveRegVals() upon initialization and termination of the toolbar. Additionally, AlertBar can make calls to its instance of ReflectionWnd to fire pop-up windows when necessary. AlertBar also implements several COM interfaces using ATL macros.

IObjectWithSite

The SetSite method in this implementation is used to initialize AlertBar and register the browser with it. This method is called by the containing browser at the beginning of its execution and passes it a pointer to the containing object, the web browser. This is a convenient place to put the initialization of AlertBar, which includes loading user settings from the registry and passing the browser object to the ReflectionWnd and UWToolBar, which are also created here. SetSite is also called immediately before the web browser terminates AlertBar, with a null value passed in. In this case, this is used as a destructor, which un-registers the UWToolBar.

DWebBrowserEvents2

This class has two event handlers BeforeNavigate2 and DocumentComplete:

BeforeNavigate2

This event fires before navigation in the given object (pDisp). This also gives AlertBar the URL that the browser is attempting to navigate to and allows AlertBar to cancel the browser's navigation. This is where AlertBar performs the first round checks, as these checks need only the URL of the attempted navigation.

AlertBar will not perform the first round checks if the domain of the attempted URL is already in the whitelist. In this case, the variable `outerFrameIsInwhitelist` is set to `TRUE` so that this page will also be ignored after navigation is complete. Otherwise, the first round checks are done in the function `FirstAlert`, and a warning string is created to notify the user of any problems that the first round checks detected. The three checks done here are Initial look-up, Pharming check and URL validation Check. The Initial look-up iterates through the whitelist and compares the current URL to each of them. The Pharming check simply compares IP for a URL retrieved from Local DNS, Remote DNS and Network DNS. URL validation check searches a URL in Google Search. If activated first round checks, that is, those that detect problems, then a pop-up warning message is fired, with the warning string that describes all the possible problems detected so far, where the user is given the option to cancel navigation. If the user decides to do so, the value of `Cancel`, passed in by the browser, is set to `TRUE` to cancel navigation.

DocumentComplete

This event fires when the given object (pDisp) has been completely loaded and initialized. This means that every page, image and script of the document has loaded. This is very similar to `BeforeNavigate2`, but there is no option to cancel navigation because navigation has already occurred when this has fired. AlertBar performs the second round checks here. These include the Request URL Check, Link Check and SSL Check, Hidden fields check, Form handler check. These checks each retrieve the browser object from the given pDisp. From this, they retrieve the current document. Each check queries the document for a different set of information. The URL check Checks the current URL of the page. The Link Check simply retrieves a collection of all the links on the page and iterates through them. Similarly, the SSL Check retrieves certificate is verified or not. Request URL Check retrieves a collection of request URLs from the current document. Form handler check finds out that is information is submitted to foreign domain. Hidden fields check counts the number of hidden fields in page. Once the checks are done, results of all checks are applied to trained neural network and the status message is updated. The

decision is supplied to the warning light in UWToolbar's Status Button and, will pop up the current status of the page as a warning message, through ReflectionWnd. Here user can choose to stop or proceed based on the results and also if user wants to add current URL to whitelist of domains.

5.4 Results

PageSafe, can successfully detects all phishing attacks taking place through DNS poisoning or pharming attacks. Phishing sites are short lived. PageSafe only allows those URLs that have been indexed by Google i.e. these URL related websites are not short lived. By doing this, a large percentage of total phishing sites, is removed. But a phishing website can be hosted on a compromised legitimate server indexed by Google. To cope up with phishing websites hosted on a legitimate domain, PageSafe performs automatic classification of web pages using artificial neural network.

A sample of 101 websites including phishing as well as legitimate websites is taken from www.phishtank.com. The performance of rules is analyzed using MATLAB. The sample is used as: 71 sites in training, 20 sites for validation and 10 sites for testing. Percentage errors in training, validation and testing are 5.6, 0 and 0 respectively.

After training the neural network a new dataset of 60 sites was taken containing phishing as well as legitimate sites. The accuracy achieved was 97% i.e. only 3% sites are wrongly classified. Figure 6.1 shows the confusion matrix which is achieved when this sample is tested by trained neural network using MATLAB.

PageSafe warns when a user tries to visit a website which is not indexed by Google i.e. only allows those websites that are not short lived. Phishing sites are short lived hence by allowing only Google indexed web sites reduces the probability of a website being a phish. PageSafe performs automatic classification only for those phishing websites that are hosted on legitimate domains which are lesser in number hence false positives are reduced by a significant value. Table 6.1 compares PageSafe with other available browser toolbars.

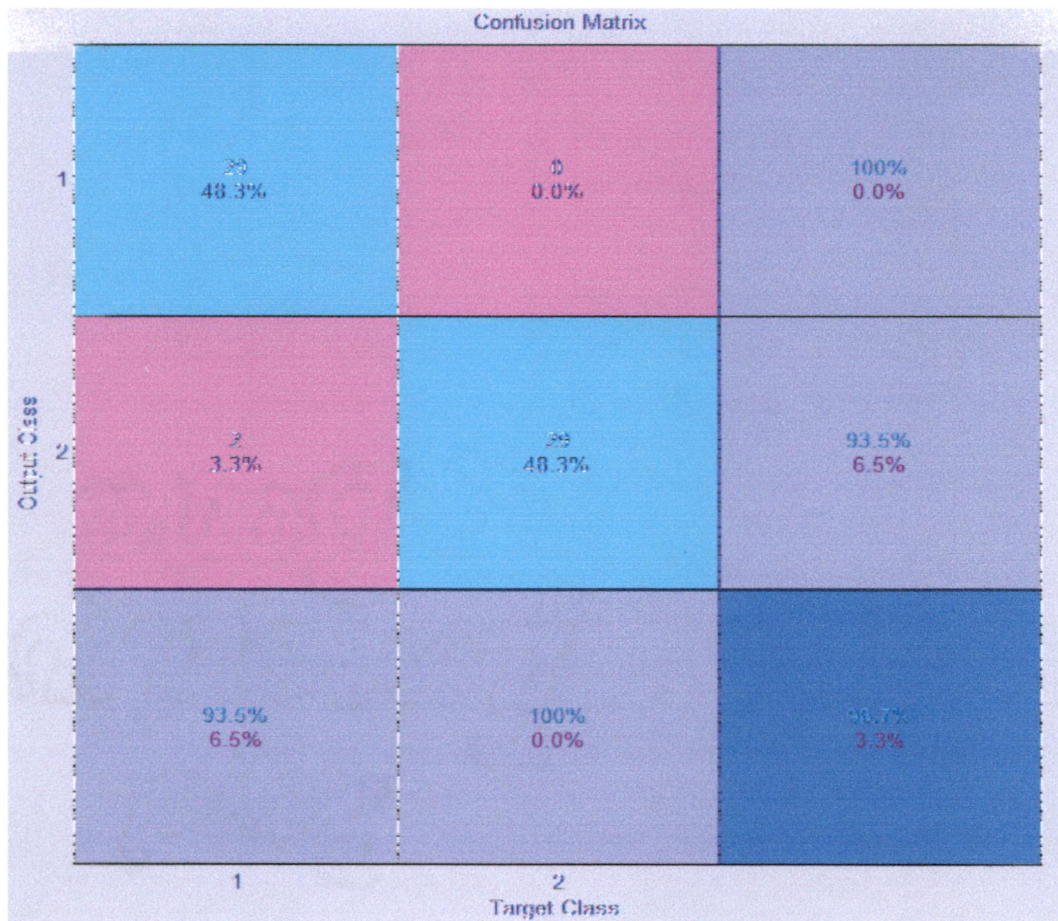


Figure 6.1: Confusion Matrix

Toolbars	Pharming Detection	Page Anomalies Analysis	URL Validation	User-Assistance
Antiphish	No	No	No	Complete
Spoofguard	No	Yes	No	Complete
PhishGuard	Yes	No	No	Complete
Blacklist	No	No	No	No
ITrustPage	No	No	Yes	Very less
PageSafe	Yes	Yes	Yes	Very less

Table 6.1: Comparison With Available Browser Toolbars

Chapter 6: Conclusion And Future Work

Conclusion

Phishing have brought a dramatic increase in the number and sophistication of attacks involved in stealing user's secret information. This thesis work presents PageSafe for preventing user from filling out phishing web forms. PageSafe relies on two key observations: (1) user's input can be used to disambiguate between legitimate and phishing sites and (2) Internet repositories of information can be used to assist the user with the decision making process. PageSafe does not preserve any secret information as information preserving approach completely dependent on user. PageSafe merges user assistance with automatic classification reducing the false positives by a significant value. PageSafe uses some new features and uses a machine learning approach (Artificial Neural Network) for automatic classification and achieves 97% accuracy. PageSafe protects user even if the system is compromised by detecting DNS poisoning and also from those phishing sites hosted on legitimate domains. PageSafe cannot protect user from key-loggers screen-grabbers and client side scripting.

Future Scope

More functionally can be added to PageSafe for protecting user against key-loggers and screen-grabbers and client side scripting attacks. PageSafe uses Google search to validate a URL which can be refined by using more external repositories such as Yahoo search etc. PageSafe uses artificial neural network approach and achieves 97% accuracy. This can be improved by adding more effective rules and using other machine learning approaches.

References

- [1] G. Aaron, R. Rasmussen, "Global Phishing Survey: Trends and Domain Name Use 2H2009", Anti-Phishing Working Group, 2009. Internet: <http://www.antiphishing.org/>. [Last accessed: May 12, 2010].
- [2] R. Dhamija, J. D. Tygar, and M. Hearst, "Why Phishing Works", *In Proceedings of the Conference on Human Factors In Computing Systems (CHI)* , ACM Press, Montreal, Canada, 2006, pp. 581-590.
- [3] A. Litan, "Phishing Attack Victims Likely Targets for Identity Theft", Gartner Research (2004). Internet: http://www.gartner.com/resources/120800/120804/phishing_attack.pdf. [Last accessed: May 12, 2010]
- [4] G. Ollmann, "The Phishing Guide" , Next Generation Security Software Ltd. Internet: <http://www.ngssoftware.com/papers/nisr-wp-phishing.pdf>. [Last accessed: May 30, 2010]
- [5] G. Ollmann, "The Pharming Guide", Next Generation Security Software Ltd. Internet: <http://www.ngssoftware.com/papers/nisr-wp-pharming.pdf>. [Last accessed: May 15, 2010]
- [6] http://searchsecurity.techtarget.com/sDefinition/0,,sid14_gci1134829,00.html.
- [7] "In Session Phishing Attacks", Trusteer Research Paper, December 29, 2008. Internet: www.trusteer.com/files/In-session-phishing-advisory-2.pdf. [Last accessed: June 11, 2010]
- [8] E. Kirda and C. Kruegel, "Protecting Users against Phishing Attacks with AntiPhish", *29th Annual International Computer Software and Applications Conference*, ACM Press, Washington, USA, 2005, Vol. 01, pp. 517-524.
- [9] N. Chou, R. Ledesma, Y. Teraguchi, D. Boneh, and J. C. Mitchell, "Client-side defense against web-based identity theft", *11th Annual Network and Distributed System Security Symposium*, ACM Press, Ontario, Canada, 2004, Vol. 380.
- [10] B. Ross, C. Jackson, N. Miyake, D. Boneh, J. C. Mitchell, "Stronger Password Authentication Using Browser Extensions ", *Proceedings of the 14th conference on USENIX Security Symposium* ,ACM Press, Baltimore, MD, 2005, Vol. 14, pp. 2-2
- [11] R. Dhamija and J.D. Tygar, "The Battle Against Phishing: Dynamic Security Skins", *Proceedings of the 2005 symposium of usable privacy and security*, ACM Press, Pittsburg, Pennsylvania, 2005, Vol. 93, pp. 77-88.

- [12] L. Sean M. Allister, E. Kirda, C. Kruegel , “On the Effectiveness of Techniques to Detect Phishing Sites ”, *Proceedings of the 4th international conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, ACM Press, Switzerland, 2007, Vol. 4579, pp. 20-39.
- [13] K. Oberoi, A. K. Sarje, “An Anti-Phishing Application for the End User”, *IIT Kanpur Hackers’ Workshop 2009*, IIT Kanpur, UP, INDIA, March 2009.
- [14] J.W. Ratcliff and D.Metzener, “Pattern Matching: The Gestalt Approach”, *Dr. Dobb’s Journal*, 1988, pp. 46-51.
- [15] Y. Pan, X. Ding, “Anomaly Based Web Phishing Page Detection”, *Proceedings of the 22nd Annual Computer Security Applications Conference*, ACM press, Washington, USA, 2006, pp. 381392
- [16] B. Ross, C. Jackson, N. Miyake, D. Boneh, J. C. Mitchell, ”Stronger Password Authentication Using Browser Extensions”, *Proceedings of the 14th conference on USENIX Security Symposium ,2005*, Vol. 14.
- [17] R. Dhamija and J.D. Tygar, “The Battle Against Phishing: Dynamic Security Skins ”, *Proceedings of the 2005 symposium on Usable privacy and security*, ACM Press, Pitsburg, Pennsylvania, 2005, Vol.93, PP 77-88.
- [18] J. Kang and D. Lee, ” Advanced White List Approach for Preventing Access to phishing Sites ” International Conference on Convergence Information Technology, Korea, 2007.
- [19] T. Ronda, S. Saroiu, A. wolman, “Itrustpage: a user-assisted anti-phishing tool”, *ACM SIGOPS Operating System Reviews*, ACM Press, 2008, Vol. 42.
- [20] M.F. Moller, “A Scaled Conjugate Gradient Algorithm For Fast Supervised Learning”, *Neural Network Letters*, ScienceDirect, 1993, vol.6, PP 525-533.