Server Side Support for SiteWatcher, an Antiphishing Software

City University of Hong Kong
Department of Computer Science

BSCCS/BSCS Final Year Project Report 2006 - 2007

06CS051

Server Side Support for SiteWatcher, an Antiphishing Software

(Volume 1 of 1)

Student Name : Man Kin Pong

Student No. :

Programme Code : BSCCS

Supervisor : Dr. Liu, Wenyin

1st Reader : Dr. Wang, Jianping

2nd Reader : Prof. Deng, Xiaotie

For Official Use Only
Server Side Support for SiteWatcher, an Antiphishing Software

Student Final Year Project Declaration

I have read the project guidelines and I understand the meaning of academic dishonesty, in particular plagiarism and collusion. I hereby declare that the work I submitted for my final year project, entitled:

**Server Side Support for SiteWatcher, an Antiphishing Software**

does not involve academic dishonesty. I give permission for my final year project work to be electronically scanned and if found to involve academic dishonesty, I am aware of the consequences as stated in the Project Guidelines.

Student Name: ___________________________ Signature: ___________________________

Student ID: ___________________________ Date: ___________________________
Abstract

With the growth of Internet use and e-commerce, the entire Internet community is familiar with phishing attack. While spam is an annoyance in nowadays, phishing attacks can cause the major financial disruptions for those victims.

Lots of anti-phishing systems have been developed to fight against phishing attacks. We, the AntiPhishing Group [8], have implemented an anti-phishing tool, SiteWatcher Client, too. However, those anti-phishing systems including ours may block certain phishing attacks, but not all. After studying, it is proposed to develop a smart server support for SiteWatcher Client in order to increase the accuracy in phishing detection. SiteWatcher Server is involved to support the client application.

Based on the study of the SiteWatcher Server, it ensures the value of the server support to SiteWatcher Client.
Acknowledgement

We would like to deeply thank the various people who, during the semesters in which this endeavor lasted, provided us with helpful guidance and support. Without their care and consideration, this project would likely not have completed maturely.

First of all, we would like to thank especially Dr. Liu, WY at the City University of Hong Kong, for his encouragement and patience throughout the duration of the project. His professional knowledge and advices stimulate us, making the project to be completed smoothly.

Second, we would like to thank all the assistants who have given quick supports and helpful suggestions in doing the final year project, Mr. Anthony Y.FU, Mr. Xuan Li, Mr. Chenke Y. and Miss Huiqiong W.

Third, we thank the support of our department, the Department of Computer Science of the City University of Hong Kong. With the technical supports, hardware and software provided by the department, we have an appropriate working environment.

Finally but not least, to all classmates, who are willing to exchange both technical and non-technical ideas with us and give support in any form.
# Table of Content

Student Final Year Project Declaration .......................................................... 2  
Abstract .............................................................................................................. 3  
Acknowledgement ................................................................................................. 4  
Table of Content .................................................................................................. 5  
Table of Figures, Tables and Code Snippets .......................................................... 8  

1. Introduction ........................................................................................................ 11  
   1.1 Origins of the word "Phishing" ........................................................................ 12  
   1.2 How phishing attacks? .................................................................................. 12  
   1.3 Problem of Phishing Attacks ........................................................................ 14  
   1.4 Is Hong Kong safe? ....................................................................................... 15  
   1.5 Anti-phishing System .................................................................................... 15  

2. Project Objectives ............................................................................................... 17  
   2.1 Aims of the project ........................................................................................ 17  
   2.2 Scope of the project ....................................................................................... 19  
   2.3 Importance of the project ............................................................................. 19  

3. Literature Review ............................................................................................... 20  
   3.1 Microsoft Phishing Filter .............................................................................. 21  
      3.1.1 How to Detect Phishing Attack? ............................................................... 21  
         3.1.1.1 Process 1: Browser DAT and URL Local Cache .............................. 22  
         3.1.1.2 Process 2: URL Reputation Web Server ........................................ 23  
         3.1.1.3 Process 3: Heuristic Analysis and URL Verification ....................... 24  
      3.1.2 How to Collect Phishing Data? ............................................................... 24  
   3.2 Mozilla AntiPhish ......................................................................................... 25  
      3.2.1 How to Detect Phishing Attack? ............................................................... 26  
         3.2.1.1 Before Detection ............................................................................. 27  
         3.2.1.2 Store Sensitive Information into the "Watch List" ......................... 27  
         3.2.1.3 Control the Sensitive Information Flow ........................................ 27  
      3.2.2 How to Collect Phishing Data? ............................................................... 28  
   3.3 Google Safe Browsing ................................................................................... 28  
      3.3.1 How to Detect Phishing Attack? ............................................................... 29  
         3.3.1.1 Phishing Detection on Local Blacklist ............................................ 29  
         3.3.1.2 Phishing Detection on Lookup Server .......................................... 29  
      3.3.2 How to Collection Phishing Data? ........................................................... 30  
   3.4 SiteWatcher Client ....................................................................................... 31  
      3.4.1 How to Detect Phishing Attacks? ............................................................. 32  
         3.4.1.1 Process 1: Preset Before Detection ............................................... 33  
         3.4.1.2 Process 2: Phishing Detection by Lists ......................................... 34  
         3.4.1.3 Process 3: Phishing Detection by Visual Similarity ....................... 35  
      3.4.2 How to Collect Phishing Data? ............................................................... 35  
   3.5 Small Problem Leads Disaster ...................................................................... 36  

4. Methodology and Resources .............................................................................. 38  
   4.1 Overview ....................................................................................................... 39  
      4.1.1 Organization of this Chapter .................................................................. 39  
      4.1.2 What is SiteWatcher Server? ............................................................... 40  
      4.1.3 What does SiteWatcher Server do? ..................................................... 41  
   4.2 System Specification ...................................................................................... 42  
   4.3 Others Requirements .................................................................................... 43  
      4.3.1 Hardware Requirement ......................................................................... 43  
      4.3.2 Software Requirement ......................................................................... 43  
      4.3.3 Programming Language ....................................................................... 44
4.4 SiteWatcher Server Architecture ................................................................. 44
  4.4.1 Multi-layer Application Architecture ...................................................... 44
    4.4.1.1 The Presentation Layer ........................................................................ 45
    4.4.1.2 The Business Logic Layer ..................................................................... 45
    4.4.1.3 The Data Layer ..................................................................................... 45
  4.4.2 Detailed Architecture .................................................................................. 45
    4.4.2.1 PhishingDefService ................................................................................. 47
    4.4.2.2 SiteWatcher Administration Webpage .................................................. 47
    4.4.2.3 SiteWatcher Scheduler .......................................................................... 48
    4.4.2.4 MySQL Server ....................................................................................... 48
    4.4.2.5 SiteWatcher Client ................................................................................ 49

4.5 Security ........................................................................................................... 50
  4.5.1 SiteWatcher Administration Webpage Security ........................................... 50
    4.5.1.1 Authentication ....................................................................................... 50
    4.5.1.2 Authorization ......................................................................................... 51
  4.5.2 PhishingDefService Security ....................................................................... 52
    4.5.2.1 Client Framework ................................................................................... 52
    4.5.2.2 Admin Framework .................................................................................. 53
    4.5.2.3 Both Client and Admin Frameworks ...................................................... 54
  4.5.3 SiteWatcher Scheduler Security ................................................................... 55

4.6 Database ......................................................................................................... 55
  4.6.1 Database Schema ....................................................................................... 55
  4.6.2 Database Description .................................................................................. 57

4.7 PhishingDefService Features and Functionalities ........................................... 58
  4.7.1 PhishingDefService Client Framework ....................................................... 58
  4.7.2 PhishingDefService Admin Framework ...................................................... 59

4.8 SiteWatcher Scheduler Features and Functionalities ...................................... 62
  4.8.1 Auto-Update Black List .............................................................................. 62
  4.8.2 Auto-Check Black List ............................................................................... 64
  4.8.3 Auto-Delete .................................................................................................. 66
  4.8.4 Reporting ..................................................................................................... 67
  4.8.5 Configuration Page ..................................................................................... 69

4.9 SiteWatcher Administration Webpage Features and Functionalities .............. 70
  4.9.1 Suspected List Page ..................................................................................... 71
    4.9.1.1 Actions in the Suspected List Page ......................................................... 72
    4.9.1.2 Auto Correction in the Suspected List Page ............................................ 72
    4.9.1.3 Threshold Training ................................................................................ 72
  4.9.2 Black List Page ............................................................................................ 75
    4.9.2.1 Actions in the Black List Page ................................................................ 76
  4.9.3 White List Page ........................................................................................... 78
    4.9.3.1 Actions in the White List Page ............................................................... 78
  4.9.4 White Domain Name List Page .................................................................... 78
    4.9.4.1 Actions in the White Domain Name List Page ....................................... 80
  4.9.5 Protected List Page ..................................................................................... 81
    4.9.5.1 Actions in the Protected List Page ......................................................... 81
  4.9.6 Customized Protected List Page .................................................................. 85
    4.9.6.1 Actions in the Customized Protected List Page ..................................... 86
  4.9.7 User Profile .................................................................................................. 88
  4.9.8 Configuration Page ..................................................................................... 89
  4.9.9 Help File ...................................................................................................... 89

5. Result and Discussion ......................................................................................... 91
  5.1 SiteWatcher Server .......................................................................................... 92
    5.1.1 Auto-Update Black List Performance Evaluation ....................................... 92
  5.2 SiteWatcher Client with SiteWatcher Server Support ..................................... 93
5.2.1 SiteWatcher Client Performance Evaluation .......................................................93

6. Conclusion and Future Work ..............................................................................95
   6.1 Critical Review ...............................................................................................96
   6.2 Achievement .................................................................................................97
   6.3 Future Work .................................................................................................98
     6.3.1 More Powerful Services to SiteWatcher Client ...........................................98
     6.3.2 More Advanced Frameworks to Developer ................................................99
   6.4 Conclusion ..................................................................................................102

7. Reference ........................................................................................................103

8. Appendix ........................................................................................................107
   8.1 Monthly Log ...............................................................................................108
Table of Figures, Tables and Code Snippets

Figure 1.1 Phishing email example .......................................................... 13
Figure 1.2 Phishing reports received from January 2006 to January 2007 14
Figure 1.3 Most target industry sectors in January 2007 .......................... 14

Figure 3.1 Phishing detections in Microsoft Phishing Filter .......................... 22
Figure 3.2 Yellow warning raised in Internet Explorer 7 ............................ 23
Figure 3.3 Red warning raised in Internet Explorer 7 ............................... 23
Figure 3.4 Yellow warning raised in MSN Search Toolbar ....................... 24
Figure 3.5 Red warning raised in MSN Search Toolbar ............................ 24
Figure 3.6 Collection of phishing data ..................................................... 25
Figure 3.7 Flowchart of how AntiPhish controls the sensitive information 26
Figure 3.8 The AntiPhish application menu integrated into the Firefox 2.0 27
Figure 3.9 Phishing alert message in the Firefox 2.0 ............................... 28
Figure 3.10 Alert message raised by Google Safe Browsing in the Firefox 2.0 30
Figure 3.11 Flow of SiteWatcher Client .................................................. 32
Figure 3.12 Protected List in SiteWatcher Client ..................................... 34
Figure 3.13 Alert message raised by SiteWatcher Client ......................... 34
Figure 3.14 Image Signature in size 10*10............................................. 35

Figure 4.1 Simplified system flow of SiteWatcher Server ......................... 41
Figure 4.2 Classic 3-tier architecture model .......................................... 44
Figure 4.3 SiteWatcher Server 3-tier architecture model ....................... 46
Figure 4.4 Login module ...................................................................... 50
Figure 4.5 Access Denied Page .............................................................. 51
Figure 4.6 Communicate with SiteWatcher Server in different roles ........ 52
Figure 4.7: SiteWatcher Server database schema .................................... 56
Figure 4.8: Client Framework class diagram ......................................... 58
Figure 4.9: Admin Framework class diagram ....................................... 60
Figure 4.10: Auto-Update Black List process ........................................ 62
Figure 4.11: Contents in preprocessing.xml .......................................... 63
Figure 4.12: Part of the contents in index.xml ....................................... 64
Figure 4.13: Auto-Check Black List process ........................................ 65
Figure 4.14: Black URL status code ..................................................... 66
Figure 4.15: Suspected List before Auto-Delete ..................................... 67
Figure 4.16: Suspected List after Auto-Delete ....................................... 67
Figure 4.17 Reporting in automated tasks .............................................. 68
Figure 4.18: Auto-Delete event log and email report ............................. 68
Figure 4.19: SiteWatcher configuration page ....................................... 69
Figure 4.20: configuration values in web.config .................................... 70
Figure 4.21: Login page of SiteWatcher Administration Webpage ........ 71
Figure 4.22: Suspected List Page ......................................................... 72
Figure 4.23: A sample of EMD and threshold ....................................... 74
Figure 4.24: Black List Page ............................................................... 76
Figure 4.25: Add Black URL Page ........................................................ 77
Figure 4.26: Connectivity report of a Black URL .................................. 77
Figure 4.27: White List Page ............................................................... 78
Figure 4.28: Add White List Page ........................................................ 78
Figure 4.29: White Domain List Page .................................................. 80
Figure 4.30: Add White Domain Name List Page ................................. 81
Figure 4.31: Protected List Page .......................................................... 82
My momma always said,

"Life was like a box of chocolates.
You never know what you're gonna get."

- Forrest Gump (1994)

You may probably get a phishing!

- Carlos
1. Introduction
1.1 Origins of the word “Phishing”

The term “ phishing” (FISH-ing) comes from the fact that Internet scammers are using increasingly sophisticated lures as they “fish” for users’ sensitive information such as usernames, passwords and credit card details, by masquerading as trustworthy and well known enterprises such as Paypal and eBay. The “ph” comes from a common hacking term, phreaking, which is the first type of hacking before the widespread of the use of the Internet. In 1996, some America Online (AOL) accounts were hacked and they were called “phish”. These phish were treated as a form of electronic currency where scammers could trade phish for hack software. According to the Anti-Phishing Working Group (APWG), it is the first mention on the Internet of phishing which is on the alt.2600 hacker newsgroup in January 1996, but phishing may have been used even earlier in the printed edition of the hacker newsletter “2600” [1]:

“It used to be that you could make a fake account on AOL so long as you had a credit card generator. However, AOL became smart. Now they verify every card with a bank after it is typed in. Does anyone know of a way to get an account other than phishing?” [2].

1.2 How phishing attacks?

There are several types of phishing attacks. In this paper, we would focus on phishing attacks through emails and websites.

Thousands of phishing emails are sent out with a link to a phishing website, which solicits users’ private information such as credit card data or password. When the form is submitted by the user, it sends the data to phishers while leaving the user on the real company’s website so that they do not suspect anything.

As the phishing emails and websites look quite authentic, featuring corporate logos and format similar to ones used by legitimate enterprises, many victims do not know the phishing traps until they find that their money have gone. According to APWG’s estimation, 5% of all recipients of a particular phishing attack have provided personal data to the phishers [1]. It means that if a single phishing email is spammed to 1,000,000 Internet users, nearly 50,000 of those recipients have submitted their bank,
credit-card account or other personal information to the phishers. Besides, it is estimated that the phishing attack cost US$137 million to US$1.2 billion in one year [3].

In order to attract victims to submit their information to those phishing websites, the appearance of phishing emails and websites should be similar to the legitimate ones. Copying the email and webpage code from a major site such as AOL, Paypal is the most common ploy for phishing attacks. The original links in the legitimate email are replaced by links which redirect users to a replica page that appears to be part of the company’s website.

Figure 1.1 is a screen capture of a phishing email in which it tries to gain trust from users by pretending the real eBay email. The link leads to a replica webpage of eBay login page that was used to trick users into submitting their private information. When clicking the link, a pop-up window is created and it masks its identity so that the address appeared legitimate [4].

---

Figure 1.1 Phishing email example

---
1.3 Problem of Phishing Attacks

The phishing is growing rapidly. And the phishing attacks not only target on AOL accounts but also online banking, payment services and online e-commerce sites.

![Phishing Reports Received Jan. '06 - Jan. '07](image)

Figure 1.2 Phishing reports received from January 2006 to January 2007

![Most target industry sectors in January 2007](image)

Figure 1.3 Most target industry sectors in January 2007

From figure 1.2, the phishing reports received have been increased nearly 100% within one year. 29,930 unique phishing URLs are reported as phishing spoof websites in
January 2007. The phishing attacks tend to keep on growing in the coming years. Furthermore, according to the Figure 1.3, financial services are the most target industry sector at 88.9% of all phishing attacks in January 2007. With the rapid growth of phishing attacks, it is eroding consumer trust in the safety of online transactions. And it is negatively impacting the way companies do business with their existing customers. The phishing threats also damage consumer confidence in conducting business over the Internet. This confidence is critical to the continued growth of e-commerce.

1.4 Is Hong Kong safe?

No. Phishing attacks are very much global. The threats targeting the Asia-Pacific region is the same compared with the rest of the world. In October 2004, 12 people have been arrested in Hong Kong in connection with a banking phishing scam. 6 of the suspects, including a Russian man, have been charged with theft amid allegations that they hoodwinked HK$600,000 (US$77,000) in an Internet banking scam. 12 HK-based customers of HSBC bank fell for the phishing bait and handed out their personal details after receiving a fake email [5].

Hong Kong users continue to suffer from phishing attacks targeting local financial institutions and phishing websites were found to host in Hong Kong during 2006 [6]. In the future, a dangerous trend of targeted attacks is hitting organizations in Hong Kong. To avoid further financial lost, a highly reliable antiphishing system is a must to protect both legitimate enterprises and individuals.

1.5 Anti-phishing System

The APWG has suggested that phishing attacks are evolving. Conventional phishing techniques could become obsolete in the future. And social engineering techniques used by phishers are increasingly aware [7]. Therefore, it is urgent to fight against phishing attacks with a smart, evolutionary anti-phishing system.

Nowadays, many anti-phishing systems have been implemented in many forms such as features embedded in browsers, as part of website login procedures and as extensions or toolbars for browsers. These anti-phishing systems help users to identify legitimate websites, alert users to phishing websites, eliminating phishing email, etc. In
the later part of the paper (3. Literature Review), we will study and discuss how these anti-phishing systems protect us from phishing attacks.
2. Project Objectives
2.1 Aims of the project

This project, **Server Side Support for SiteWatcher, an Antiphishing Software**, aims in providing various supports to SiteWatcher Client in form of a server. SiteWatcher Client is an anti-phishing software which has been released to the public in the AntiPhishing Group @ Dept. of CS, City University of Hong Kong webpage [8]. SiteWatcher Client uses a visual similarity approach to detect phishing websites proposed by Dr. Liu and Prof. Deng [9]. We will discuss the Visual Similarity in the later part of the paper.

**SiteWatcher Server** is designed to provide services to keep the SiteWatcher Client most updated and accurate in phishing detections. As the SiteWatcher Client has been released for a certain time, the SiteWatcher Server is required to design a communication between itself and the SiteWatcher Client. The communication will be done based on a web service with self-defined security to enhance information security. With the communication developed, the SiteWatcher Client can receive updated phishing-analyzed data from the SiteWatcher Server while the SiteWatcher Server can collect data from the SiteWatcher Client for further Visual Similarity analysis.

After receiving data from the SiteWatcher Client, the SiteWatcher Server is designed to analyze the data in order to “train” the Visual Similarity in the SiteWatcher Client. During training, the SiteWatcher Client keeps on receiving training data to increase the accuracy in detecting phishing attacks. After the completion of training, the SiteWatcher Client is said to be “well trained” and it has the most accurate in phishing detections.

Besides, the SiteWatcher Server is designed to collect phishing information. Passively, the SiteWatcher Server receives phishing URLs reported by the SiteWatcher Client. It is a passive way relied on reporting from users using the SiteWatcher Client. In another way, the SiteWatcher Server can “search” for phishing URLs by itself. It is a more effective in collecting phishing information.

Furthermore, since the database of the SiteWatcher Server grows rapidly, the SiteWatcher Server has to clean up “useless” data by itself.
2.2 **Scope of the project**

- Study some well developed anti-phishing systems
- Study the importance of the SiteWatcher Server to the SiteWatcher Client
- Design the architecture of SiteWatcher Server
- Design a secured communication between SiteWatcher Server and the SiteWatcher Client
- Implement the SiteWatcher Server
- Conduct testing to compare the performance of the new version of SiteWatcher (Client and Server) with the old version (Client only)
- Conduct testing to compare the performance of the new version SiteWatcher with other antiphishing software

2.3 **Importance of the project**

If the SiteWatcher Server is a success, the performance, accuracy of the SiteWatcher Client will be greatly improved. Since SiteWatcher Server can evolve itself to against the rapid changes of phishing attacks, more enterprises and individuals can be protected with the latest SiteWatcher Client with the support of the SiteWatcher Server.
3. Literature Review
Much effort has been made to protect Internet users from phishing attacks. There are many anti-phishing systems nowadays. After studying, several anti-phishing systems are selected to compare the anti-phishing technologies and the collections of phishing data from the Internet.

We study those antiphishing systems and give features and descriptions in these topics.

3.1 Microsoft Phishing Filter

Microsoft Corporation has introduced the Microsoft® Phishing Filter to block websites and caution users about reported and suspected phishing websites [10]. Currently, the Phishing Filter is embedded in the Microsoft products only. Internet Explorer 7, MSN® Search Toolbar and MSN® Hotmail® are now protected by the Microsoft Phishing Filter. It uses several patent-pending technologies designed to warn or block users from potentially harmful websites:

- Several layers of phishing detections in the Phishing Filter
- Reporting system for users to report and for collecting phishing attacks

Jeremy Dallman, an IE program manager, expressed that “it was the only one that consistently caught more than 60% of phishing sites while having the lowest possible rates of incorrect ratings (otherwise known as false positives)” [11]. According to Microsoft, over 1,000,000 suspected and phishing websites are blocked each week, and over 10,000 phishing websites are added to blacklists every week in which it can provide quick response in blocking the known phishing websites.

3.1.1 How to Detect Phishing Attack?

According to a report written by Jefferson, Microsoft Phishing Filter detects phishing attacks in 3 processes [12]. Figure 3.1 shows the phishing detections in the Microsoft
Phishing Filter. Each process has different strategy to detect and filter specific type of phishing attacks.

![Diagram showing the process of phishing detections in Microsoft Phishing Filter]

### 3.1.1.1 Process 1: Browser DAT and URL Local Cache

The Microsoft Phishing Filter first checks the .DAT file and cache in the local machine to find a match of a previously rated phishing site when a user enters a website using Browser Technology embedded with Phishing Filter such as Internet Explorer 7 and Windows Live Toolbar. As it is the first checking, it simply checks the domain and path of the URL (i.e. `http://domain.com/path`) with the query string data removed. If the URL matches the rated phishing site in the .DAT file and cache, a yellow warning or a red warning will be raised depending on the phishing rating of the URL to alert the user about phishing attacks. A yellow warning means that the user has entered a suspected phishing website and it is recommended to avoid entering any personal information on the website. Figure 3.2 shows the yellow warning raised by the phishing filter in the Internet Explorer 7. A red warning means that the user has entered a confirmed phishing website. A threat level warning page is raised to navigate the user to a new page. This warning page offers users the option to close the webpage immediately or proceed at their own risk to the website. Figure 3.3 shows the red warning raised by the phishing filter in the Internet Explorer 7.

If the Browser DAT file and URL Local Cache do not contain the phishing ratings of the URL, the phishing detection will move on to the URL Reputation Web Server.
3.1.1.2 Process 2: URL Reputation Web Server

When the first process, Browser DAT file and URL Local Cache, cannot judge for the phishing attacks, the Phishing Filter in the local machine tries to match a rated phishing list of URLs stored on the URL Reputation Web Server hosted by MSN. If the URL matches the list, a yellow or red warning will be raised depending on rating of the website. Figure 3.4 shows the yellow warning and Figure 3.5 shows the red warning raised by the phishing filter in MSN Search Toolbar.

If the match is not identified, the phishing detection will move on the next detection process, the Heuristic Analysis and URL Verification.
3.1.1.3 Process 3: Heuristic Analysis and URL Verification

If URL Reputation Web Server still cannot judge for the URL, the webpage is subjected to a heuristic analysis by the Phishing Filter. Based on the heuristic analysis and the URL verification, the Phishing Filter will calculate the phishing rating for the URL. If the rating exceeds a limit, a yellow or red warning will be raised to alert user depending on the rating of the URL. Otherwise, the webpage continues as normal.

3.1.2 How to Collect Phishing Data?

The Microsoft Phishing Filter collect phishing data by 2 ways: A team of Graders from Microsoft confirms the phishing reports from end users and site owners and inserts
these data to the URL Reputation Web Server; Phishing databases from third parties also provide phishing data to the URL Reputation Web Server [13]. Figure 3.6 shows the collection of phishing data.

![URL Reputation Service Diagram](https://urs.microsoft.com)

**Figure 3.6 Collection of phishing data**

### 3.2 Mozilla AntiPhish

Mozilla Firefox 2.0 has introduced the AntiPhish, a Firefox extension for anti-phishing support. Unlike the several layers of antiphishing strategies used by Microsoft Phishing Filter, AntiPhish simply keeps track of users’ sensitive information and prevents this information being passed to a website that cannot be “trusted” [14].

Although AntiPhish only supports Firefox 2.0, the performance in fighting against phishing attacks is outstanding. A recent study [15] commissioned by Mozilla in 2006 claimed that the AntiPhish in Firefox 2.0 had a higher accuracy in detecting phishing websites than Microsoft’s Internet Explorer 7. The study found that Firefox in its most...
secure configuration blocked 81.5% of all phishing websites. Internet Explorer 7 blocked just 66.35% of phishing websites. And also, Firefox 2.0 failed to flag a phishing websites in 117 instances when Internet Explorer 7 caught it, while the Internet Explorer 7 let 243 URLs slop through that Firefox stopped.

3.2.1 How to Detect Phishing Attack?

Instead of checking the characteristics of phishing attacks, AntiPhish detects use another approach to detect phishing. It uses JavaScript to check the HTML of the webpage where information is going to be submitted to determine if the sensitive information “belongs” to the domain of the website. Figure 3.7 shows how the sensitive information flow is controlled by AntiPhish [14].

![Flowchart of how AntiPhish controls the sensitive information](image)

Figure 3.7 Flowchart of how AntiPhish controls the sensitive information
3.2.1.1 Before Detection

When the AntiPhish is first installed in the Firefox 2.0, a browser prompts a request for a master password when a user enters input into a form for the first time. The master password is used to encrypt the sensitive information before it is saved for the auto form-filler. The encryption and decryption are done by the symmetric DES algorithm.

3.2.1.2 Store Sensitive Information into the “Watch List”

After the user fills in the sensitive information and submits the form, the AntiPhish scans the webpage and to capture this information. This scanning is done by manually in the current version of AntiPhish. The AntiPhish then stores the sensitive information and a mapping of where this information “belongs” to. The information and the domain name will be encrypted and stored into a “watch” list. Figure 3.8 shows the AntiPhish application menu integrated into Firefox 2.0.

![Figure 3.8 The AntiPhish application menu integrated into the Firefox 2.0](image)

3.2.1.3 Control the Sensitive Information Flow

As HTML form elements such as text field of type text and password and the HTML textarea are the most likely to be used to phish information, AntiPhish starts checking the potential phishing attacks when a website contains a form and the form elements mentioned above.

When the user enters any information into any of these form elements, AntiPhish checks the “Watch List”. For each value in the list that is identical to the one just
entered by the user, the corresponding domain is determined. If the current website is not among these domains, AntiPhish generates an alert and redirects to an information page about phishing attacks. Figure 3.9 shows the phishing alert message box in Firefox 2.0.

![Phishing alert message box in Firefox 2.0](image)

**Figure 3.9 Phishing alert message in the Firefox 2.0**

### 3.2.2 How to Collect Phishing Data?

Since the AntiPhish is designed to avoid user to submit sensitive information to “phishing” websites rather than blocking the “phishing” website directly, it is no need to store phishing information in a server for users to download.

### 3.3 Google Safe Browsing

“Responsible disclosure allows companies like Google to keep users safe by fixing vulnerabilities and resolving security concerns before they are brought to the attention of the bad guys. [16]”. As one of the most famous search engine in the world, Google play an important role in anti-phishing. Google Safe Browsing is designed to identify phishing websites in the Google Toolbar. Originally Google Safe Browsing was a
Firefox extension, but it has been integrated into the Google Toolbar in the later version and it is expected to be built into Firefox 2.0.

3.3.1 How to Detect Phishing Attack?

Among the antiphishing system studied in this paper, Google Safe Browsing use the simplest way to detect phishing attack – the blacklist-based detection. There are 2 types of blacklist for the Google Safe Browsing: Blacklist in local machine and Blacklist in the Lookup Server. Both blacklists will be updated periodically to make sure that the blacklisting is effective and most updated. The advantages of using blacklist are that it is simple and easy to deploy.

The Google Safe Browsing provides an “enhanced protection” option for users to select if they would like to be protected by local blacklist or remote blacklist in the Lookup Server.

3.3.1.1 Phishing Detection on Local Blacklist

If enhanced protection is disabled, the user keeps a local blacklist of phishing URLs in the local machine. When the user visit websites through Firefox, every URL the browser requests will be checked if it is in the local blacklist. The list is updated timely from the Lookup Server in which it sends diffs from the user’s current blacklist or a new full blacklist.

3.3.1.2 Phishing Detection on Lookup Server

If enhanced protection is enabled, Google Safe Browsing looks up URLs in a remote blacklist hosted on a Lookup Server. Enhanced protection is a recommended option to protect from being phished as it provides better coverage because the remote blacklist is updated every minute, which is more updated than the local blacklist. When the user visits websites through Firefox, every browser request of the URL will be checked in the remote blacklist.

When a blacklisted page is loaded in the user’s machine, the page is disabled and a warning will be shown to the user. The user can choose to navigate away or to
continue to visit the page. Figure 3.10 shows the alert message raised by the Google Safe Browsing in the Firefox.

Figure 3.10 Alert message raised by Google Safe Browsing in the Firefox 2.0

3.3.2 How to Collection Phishing Data?

It is not known how URLs are added to the remote blacklist in the Lookup Server. According to the Google Safe Browsing for Firefox Official website, the toolbar combines “advanced algorithms with reports about misleading pages from a number of sources. [17]” Lorrie [18] suspected that Google Safe Browsing uses blacklists as well as heuristics which is the same as the phishing analysis in the Microsoft Phishing Filter.

Although we do not know how Google Safe Browsing gets the phishing websites, it can be predicted that end user reporting is one way in growing the blacklist.

For end user reporting, it allows users to report both false positive and false negatives. False positive means “it is a not a phishing website but it reports as phishing”. False negative means “it is a phishing website but it does not report as phishing”.

3.4 SiteWatcher Client

While enterprises are spending huge resources to fight against phishers, medium and small corporations introduce light-weighted antiphishing technology other than relying on enterprise server support.

Wenyin, Xiaotie, Guanglin, and Anthony [9], an anti-phishing working group in Computer Science Department in City University of Hong Kong, carried out studies in detecting phishing attacks. Anthony expressed that most of the early phishing webpages are very similar to the real webpages in form of appearances and HTML codes. These copy-cat phishing websites can be easily detected by HTML based visual similarity assessment proposed by Wenyin [19]. However, advanced phishing websites have different coding. And these websites even use Flash, Movie, ActiveX, Java Applet and other visual components to imitate real websites. These advanced phishing websites cannot be detected by HTML based visual similarity assessment.

Therefore, SiteWatcher Client is designed to tackle with these advanced phishing websites. Apart from detecting phishing by lists of websites, Wenyin and Xiaotie proposed an anti-phishing strategy know as “Visual Similarity” which uses visual characteristics to identify potential phishing websites and measure suspicious page’s similarity to “real” websites registered in the SiteWatcher Client.

The SiteWatcher Client protects registered “protected” websites from phishing attacks rather than scanning all websites to determine a phishing attack. Since financial services are the main target for phishers, they are most likely to be the “protected” target by SiteWatcher Client. If a user enters a website in which the appearance is “similar” to a “protected” website, alert will be raised to inform the user that the website is pretending a “protected” website.
Besides, the SiteWatcher Client is designed to be stand alone software. It means that apart from detecting phishing attacks in specific browsers such as Internet Explorer and Firefox, it can detect phishing in other software through network ports level like HTTP, POP3 and SMTP. Therefore the SiteWatcher Client can detect all phishing attacks from different software which connect to network ports and thus the SiteWatcher provides a full protection in all applications connected to the Internet.

3.4.1 How to Detect Phishing Attacks?

For SiteWatcher Client, the phishing detection is divided into 3 processes: Preset Before Detection, Phishing Detection by Lists and Phishing Detection by Visual Similarity. Figure 3.11 shows the simple flow of the SiteWatcher Client.

![Figure 3.11 Flow of SiteWatcher Client](image-url)
3.4.1.1 Process 1: Preset Before Detection

As the SiteWatcher Client is stand alone software without server support, several lists are set when installing the SiteWatcher Client into the client’s computer. The lists include:

- **Protected List** – stores URLs that would be registered and protected by the SiteWatcher Client. These URLs are called Protected URLs. The keywords of the URLs would also be stored also. (i.e. [www.paypal.com](http://www.paypal.com) contains keywords “paypal”, “pay pal”). These keywords are called Sensitive Words. Apart from these, every protected website has a threshold value which indicates the limit of similarity.

- **Black List** – stores “known” phishing URLs related to the Protected URL. These URLs are called Black URLs.

- **White List** – stores “known” non-phishing URLs. These URLs are called White URLs.

- **White Domain Name List** – store “known” non-phishing domain names. These domains are called White Domains.

With these lists, the SiteWatcher Client can perform phishing detections. Note that users are only allowed to add, edit and delete the Protected List only. Figure 3.12 shows the Protected URL with its Sensitive Word and threshold value.

* The latest version of SiteWatcher Client contains a new list called Customized Protected List and the definitions of some lists are updated. These will be discussed in the Chapter 4.9 SiteWatcher Administration Webpage Features and Functionalities.
3.4.1.2 Process 2: Phishing Detection by Lists

When a Sensitive Word of a Protected URL is found in a website that the user are visiting, the URL of the suspected URL is checked if it is in the Protected, White, White Domain Name list. If it is found in these lists, then it is treated as “trusted” website and the phishing detection ends.

If the suspected URL is not found in the above lists, the URL is checked if it is in the Black List which is a larger list of black URLs (confirmed phishing URLs). Alerts message will be raised when the URL is found in the Black List. Otherwise it will move to Visual Similarity Detection. Figure 3.13 shows the alert message raised by SiteWatcher Client.
3.4.1.3 Process 3: Phishing Detection by Visual Similarity

Visual Similarity Detection is done based on Earth Mover’s Distance (EMD) [20]. EMD is an effective approach in detecting phishing websites by measuring webpage visual similarity.

First, the “pretended” target protected website and the suspected phishing website first converted into normalized images as image signature shown in Figure 3.14.

Then EMD is applied to calculate the visual similarity difference by comparing the image signatures of the protected and the suspected website. If the EMD of the suspected webpage exceeds the threshold (visual similarity limit) of a protected webpage, it means that the appearance of the suspected is similar to the protected website, and the suspected website is classified as a phishing website. An alert message is raised to warn users. Otherwise, it is classified as a “trusted” website.

<table>
<thead>
<tr>
<th>URL &amp; Original Size</th>
<th>Original</th>
<th>100*100</th>
<th>10*10</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://www.bbb.org">www.bbb.org</a></td>
<td><img src="image.png" alt="Image Signature" /></td>
<td><img src="image.png" alt="Image Signature" /></td>
<td><img src="image.png" alt="Image Signature" /></td>
</tr>
<tr>
<td>800*796</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 3.14 Image Signature in size 10*10

3.4.2 How to Collect Phishing Data?

Since the SiteWatcher Client relied on Visual Similarity to detect phishing attacks. There is no server side support (old version). And thus all the phishing data is preset when installing it into the client’s computer.

However, the SiteWatcher Client allows users to add their own Protected URLs into the Protected List. It makes the Protected List to grow if the users wish. However, the users need to determine and add Sensitive Words to the Protected URL. And also they need to set the threshold limit by themselves.
3.5 Small Problem Leads Disaster

Both enterprises, medium and small corporations have developed different approaches to fit with their users. It seems that anti-phishing technologies mentioned above are secure enough to protect users from phishing attacks. However, as phishing attacks evolve, more and more problems are found in existing anti-phishing technologies:

1. Enterprises tend to embed anti-phishing tools into a specific browser or tool. For example, Microsoft Phishing Filter can only detect phishing in Internet Explorer 7, MSN Search Toolbar, but it is unable to detect phishing in other browser or tool such as Netscape.

2. Mozilla AntiPhish detects phishing only in HTML code level. It cannot detect phishing when the websites are coded in other language such as flash, Java Applet. Besides, it does not block the phishing website until the form is submitted. Phishers can use JavaScript to intercept the keys that the user pressed and send the information character by character to a phishing server [21].

3. Google Safe Browsing faced personal information leakage problem. Internet security firm Finjan confirmed that Google’s blacklist contained confidential usernames and passwords of individuals [22].

4. SiteWatcher Client overcomes the problems in point 1 and 2, but the threshold value of each Protected URL requires manual setting in order to perform accurate EMD calculations. It is not possible to have all manual setting in all protected URLs for normal users. Besides, the lists in the SiteWatcher Client are lack of updating without the present of server support. Without updated lists, SiteWatcher relied heavily on Visual Similarity to detect phishing attacks. As the Visual Similarity require time to calculate the EMD, the detections become slow and it uses up client’s CPU.

Since SiteWatcher Client is successful in detecting in different types of phishing websites, the performance can be further improved by introducing a smart, automated Server to support the SiteWatcher Client. Therefore, SiteWatcher Server is a must for the SiteWatcher Client.
The aim of the paper is to determine how SiteWatcher Server pays an important role in supporting the phishing detection in SiteWatcher Client. And how SiteWatcher Server tackle with the problems mentioned above.
4. Methodology and Resources
4.1 Overview

SiteWatcher Client is a successful in fighting against phishing attacks. With the support of the newly designed SiteWatcher Server, SiteWatcher Client can be further improved. This chapter gives a general idea of SiteWatcher Server.

4.1.1 Organization of this Chapter

As it is the largest chapter in this paper, let us have an outline of the sub-chapters before we start:


Chapter 4.3, “Other Requirements”, covers other less important requirements during the development of the SiteWatcher Server.

Chapter 4.4, “SiteWatcher Server Architecture”, explains the concepts of the SiteWatcher Server architecture in details and how the architecture benefits the development.

Chapter 4.5, “Security”, discusses the strategies and techniques used to enhance the security of the SiteWatcher Server. And show how to implement the defense towards hackers.

Chapter 4.6, “Database”, outlines the database structure of the SiteWatcher Server and briefly explains the use of each database table.

Chapter 4.7, “PhishingDefService Features and Functionalities”, explains the features and functionalities of the PhishingDefServer web service in details. You will learn how the PhishingDefServer works with other parts of the SiteWatcher Server.
Chapter 4.8, “SiteWatcher Scheduler Features and Functionalities”, explains the features and functionalities of the SiteWatcher Schedule in details. You will learn how the SiteWatcher Scheduler enhances the whole SiteWatcher project by the power of the scheduler.

Chapter 4.9, “SiteWatcher Administration Webpage Features and Functionalities”, explains the features and functionalities of the administration webpage. Besides, it shows the interaction between administrators and the webpage.

### 4.1.2 What is SiteWatcher Server?

As the SiteWatcher Client relies on EMD calculation to measure the webpage visual similarity and detect phishing websites, the EMD calculation process would consume lots of resources when many suspected websites are found at the same time. It is recommended that EMD should be avoided for frequently used webpages. Besides, as the number of phishing websites is increasing in a high speed, it is impossible for SiteWatcher Client user to determine, insert and update the phishing information into the client database manually. A server is an urgent need for automation of updating phishing database. Further more, threshold training would be performed in the server automatically. Therefore, SiteWatcher Server project starts in October 2006.

SiteWatcher Server is a server working in the office of AntiPhishing Group at Department of Computer Science in the City University of Hong Kong [8]. The aims of the SiteWatcher Server are providing background services to solve the problems mentioned above. The SiteWatcher Server shows off many key features and also provides the “best practice” framework that developers can use as a base to build an advanced SiteWatcher Server and communicate with the SiteWatcher Client.
4.1.3 What does SiteWatcher Server do?

Figure 4.1 shows a simplified system flow of the SiteWatcher Server. In general, processes can be divided into:

1. Collect phishing data from SiteWatcher Client through HTTP protocol.
2. Analyze phishing data and perform threshold training to adjust the threshold value of each Protected URL in the SiteWatcher Administration Webpage.
3. Get the latest phishing Black URLs from authoritative third party phishing database.
4. Check the connectivity of Black URLs in the Black List. Remove all disconnected Black URLs.
5. Cleanup and remove all “useless” data to prevent the rapid growth of the SiteWatcher Server database.
Server Side Support for SiteWatcher, an Antiphishing Software

6. Send back the latest confirmed data including latest Black List, Protected List with Sensitive Words and threshold values, White List, White Domain Name List. The data is sent through HTTP protocol.

Note that processes are not running in linear. Processes 3 to 5 are controlled by the SiteWatcher Scheduler which is running in the background of the SiteWatcher Server. It means that while SiteWatcher Client is receiving the latest phishing data from SiteWatcher Server, the SiteWatcher Scheduler can check the connectivity of Black URLs in parallel.

4.2 System Specification

As a released version operating in the production server, the proposed SiteWatcher Server is required to:

1. Provide an effective storage for receiving phishing data from SiteWatcher Client.
2. Provide a secure communication channel between SiteWatcher Client and Server.
3. Provide a secure administration user interface to control all the phishing data and perform threshold training.
4. Provide the latest confirmed phishing information and threshold training results to the SiteWatcher Client.
5. “Search” for phishing information automatically.
6. “Check” for the connectivity of phishing Black URLs stored in the server automatically.
7. “Clean-up” useless data in the database automatically.
8. Provide a secure configuration page to configure the settings of point 5 – 7.
10. Implement SiteWatcher Server in a multi-tier application architecture.
11. Provide flexible, easy-to-maintain frameworks for development to reduce
development effort.

4.3 Others Requirements

This section would discuss other minor requirements in the SiteWatcher Server project.

4.3.1 Hardware Requirement

The hardware is provided by the AntiPhishing Group, it includes:

- Intel® Xeon™ CPU 3.4 GHz
- 3.00 GB of RAM
- 140GB of available hard-disk

Note that the hardwares mentioned above are not the minimum requirement of the
SiteWatcher Server. They are only the resources provided for the project.

4.3.2 Software Requirement

Software is provided by both the AntiPhishing Group and the Department of Computer
Science in City University of Hong Kong. The software includes:

- Microsoft® Internet Information Services (IIS) Manager Version 6.0
- Microsoft® Visual Studio .NET 2003 with .NET Framework 1.1
- MySQL Client Version 5.1.11

Under Windows environment, IIS provides a comprehensive set of tools for managing
a web server and its components in a highly reliable, manageable and scalable way for
Windows Server 2003 [23]. Besides, IIS provides an advanced security in protecting
the SiteWatcher Server by monitoring and limiting the attack surface area through
aggressive security defaults.
4.3.3 Programming Language

With Microsoft Visual Studio .NET 2003, the SiteWatcher Server is implemented using ASP.NET 1.1 and C# .NET under Microsoft development environment. And also, ADO.NET provides consistent access to data sources like MySQL database server using Open Database Connectivity (ODBC). By using ODBC interface, it allows .NET to communicate with MySQL database.

4.4 SiteWatcher Server Architecture

Building an understanding of architectural concepts is an essential aspect of managing the whole system. This section would focus on the architecture of the SiteWatcher Server. The features and functionalities of each sub-system would be discussed in Chapter 4.7 to 4.9.

4.4.1 Muti-layer Application Architecture

The SiteWatcher Server is combined of ASP.NET application and web service that uses a logical, 3-tier architecture [24]. Layered application design is recommended because it increases application performance, scalability, flexibility, code reuse, etc. Figure 4.2 shows the classic 3-tier architecture model.

![Figure 4.2 Classic 3-tier architecture model](image-url)
According to Damon [25], Applications can be broken down into 3 major areas of functionality under the 3-tier architecture model:

4.4.1.1 The Presentation Layer

The presentation layer is used in implementing the user interface. And it gets the data in the database by communicating with the business logic layer. It is separated from the business logic and database access code.

4.4.1.2 The Business Logic Layer

The business logic layer is treated as libraries of codes and it separates codes specific to the application. In this way, other line of applications can be built using the business logic layer if needed. It maximizes code re-use.

4.4.1.3 The Data Layer

The data layer encapsulates database-specific code such as SQL statements so that the database is separated from the business logic layer.

4.4.2 Detailed Architecture

Based on the 3-tier architecture model, the SiteWatcher Server is divided into 3 subsystems: SiteWatcher Administration Webpage, PhishingDefService and SiteWatcher Scheduler. Figure 4.3 shows the SiteWatcher Server 3-tier architecture model.
Figure 4.3 SiteWatcher Server 3-tier architecture model
4.4.2.1 PhishingDefService

The PhishingDefService is a web service deployed in the AntiPhishing Group production server. It is the business logic layer of the SiteWatcher Server and is the core part of the project because it provides frameworks to other SiteWatcher projects and modules. Currently, there are 2 frameworks released: PhishingDefService Client Framework and PhishingDefService Admin Framework.

PhishingDefService Client Framework is the framework that public users (SiteWatcher Client) can use it. The framework has its own Presentation Layer (Client) to provide libraries and functions to users. Since it is released to public, the functionality is limited to a certain lever due to the security. Security will be discussed in detail in Chapter 4.5 Security.

PhishingDefService Admin Framework is the framework that only authorized users (SiteWatcher Administration Webpage, SiteWatcher Scheduler) can use it. The framework has its own Presentation Layer (Admin). The functionality of PhishingDefService Admin Framework is much powerful than the Client Framework, but it requires authentication to access.

4.4.2.2 SiteWatcher Administration Webpage

The SiteWatcher Administration Webpage is an ASP.NET application deployed in the AntiPhishing Group production server used by the SiteWatcher Server administrators. Administrators can control the Lists and perform threshold training for Protected URLs through the webpage. The detailed functionalities will be discussed in Chapter 4.9 SiteWatcher Administration Webpage: Functionality.

The webpage only contains the presentation layer codes. And it uses the libraries provided by the PhishingDefService Admin Framework, which is treated as the business logic layer of the webpage, to access the data layer.

One of the advantages of implementing the administration webpage in this way is that the separation between the user interface, business logic and the data access layers is clear and each layer is independent. It means that changing the presentation layer (the webpage) would not affect the other 2 layers.
Besides, due to the separation of layers of codes, development is easier as developers can focus on different parts of the system during implementation. Developers who implementing the interface of the Administration Webpage do not need to concern the business logic, and uses the libraries provided by business logic layer implemented by other teams of developers.

Here is the implementation of the SiteWatcher Administration Webpage. It shows a clear separation of codes between the webpage and the PhishingDefService.

```csharp
Using SiteWatcher.Protected; // add the reference of PhishingDefService web service
private void GetNewProtected()
{
    // initialize the object provided in the web service
    ProtectedDB objProtected = new Protected();

    // call the function provided by the web service
    DataSet objDataSet = objProtected.dsGetProtected();

    // Display the content of the DataSet…
}
```

**Code Snippet 1: GetNewProtected() method**

### 4.4.2.3 SiteWatcher Scheduler

SiteWatcher Scheduler is a timer embedded in the PhishingDefService. The scheduler performs scheduled tasks such as checking the connectivity of Black URLs. The detailed functionalities of the scheduler will be discussed in Chapter 4.8 SiteWatcher Scheduler Features and Functionalities.

The scheduler is an independent module. It gets the libraries and functions by directly accessing the business logic layer of the PhishingDefService.

### 4.4.2.4 MySQL Server

MySQL server is the data access layer of the SiteWatcher Server. It perform select, insert, update, delete in form of SQL statements. Since this layer is independent of the presentation and the business logic layer, it allows changing the database (i.e. replacing MySQL Server with SQL Server) without affecting other layers.
4.4.2.5 SiteWatcher Client

SiteWatcher Client is a client application to detect phishing attacks (see Chapter 3.4 *SiteWatcher Client*). As it is installed in the client’s computer, it gets the libraries and functions from SiteWatcher Server by accessing the Presentation Layer (Client) of the Client Framework of the PhishingDefService Web Service through HTTP protocol.

Because the SiteWatcher Client is independent to the SiteWatcher Server, changing the SiteWatcher Client does not affect the SiteWatcher Server. There is a little adjustment of codes when changing the Presentation Layer (Client) and the Business Layer (Client) of the PhishingDefService. But the resources (time, work-force) spent in modifying the changes are minimized because of the layered design.

Due to the clear decomposition of functionality, SiteWatcher Client developers can uses the functions and libraries provided by PhishingDefService Client Framework. Based on the framework, SiteWatcher Client developers can implement the communication between SiteWatcher Client and Server. And thus they are encapsulated from the SiteWatcher Server.

Below is the implementation of the SiteWatcher Client. It shows a developer-friendly, highly encapsulated PhishingDefService Client Framework used in the application.

```java
Using LocalDB.CityUSiteWatcher; // call the web reference of PhishingDefService
Public class LocalDB
{
    private PhishingDefService objWebService;
    public LocalDB()
    {
        // initialize the PhishingDefService web service
        this.objWebService = new PhishingDefService();

        // other stuff…
    }

    private void ReportPhishing()
    {
        // call the function provided by the web service
        this.objWebService.InsertSuspected(…);
    }
}
```

**Code Snippet 2: Calling PhishingDefService in SiteWatcher Client**

Besides, The SiteWatcher Client can only access the PhishingDefService Client Framework through the Presentation Layer (Client) of the PhishingDefService. The
functions used in the SiteWatcher Client are limited by the Presentation Layer (Client). Therefore it enhances the security of the SiteWatcher Server. Security will be discussed in detail in Chapter 4.5 Security.

4.5 Security

Security is an important topic in SiteWatcher Server as “information can be corrupted when it is available on an insecure network [26].” When data stored in the SiteWatcher Server is modified in an unexpected ways, it leads a loss of both integrity and availability. Unauthorized changes made by hackers can affect the phishing data or even crash the whole SiteWatcher Server. As a result, all SiteWatcher Client users suffer from the weak security server. Therefore, a reliable security is an essential to protect both SiteWatcher Server and Client.

4.5.1 SiteWatcher Administration Webpage Security

The security designed in the SiteWatcher Administration Webpage makes use of both authentication and authorization. Authentication verifies a user’s identity and credentials while authorization verifies authenticated user’s permission for accessing a resource.

4.5.1.1 Authentication

The SiteWatcher Administration Webpage supports forms based authentication. Forms based authentication stores the usernames and passwords in the database. Users are required to login before accessing the administration content page. Figure 4.4 shows the login module. When a user fails to fill in correct username and password, warning message is raised and the user cannot access the webpage.

![Login module](image)

Figure 4.4 Login module
4.5.1.2 Authorization

The authorization for the SiteWatcher Administration Webpage uses role based security to determine if a user has the access right to a particular resource or page. Users are grouped into various such as senior administrators, administrators and the role mappings are stored in the database. Role mapping prevents users to access the functionalities in which they do not have permission.

For example in the Page_Load event in the admin Protected.aspx, a call is made to verify the role of the user:

```csharp
// Verify that the current user has access right to access this page
// Session["rid"] stores the Role ID of the current user
if (Convert.ToInt32(Session["rid"]) == objRole.intAdmin)
{
    // do some tasks here
}
else
{
    Response.Redirect("~/AccessDenied.aspx");
}
```

**Code Snippet 3: Authorization in SiteWatcher Server**

If a user who does not have permission and tries to access a specific page without login, the user will redirect to an Access Denied Page and the user is asked to login before accessing the page. Figure 4.5 shows the Access Denied Page.

![Access Denied](access_denied.png)

**Figure 4.5 Access Denied Page**
4.5.2 PhishingDefService Security

PhishingDefService web service provides both Admin and Client Frameworks for users. Figure 4.6 shows the communication between SiteWatcher Server and users with specific roles. As different roles of users access the SiteWatcher Server in different way, the Admin and Client Frameworks have their own security strategy.

![Figure 4.6 Communicate with SiteWatcher Server in different roles](image)

4.5.2.1 Client Framework

Initially, the Client Framework is released to public users. Everyone can use this framework through HTTP protocol. However, it was found that the SiteWatcher Server may experience Denial-of-service attack (DoS attack) in the future. DoS is a computer crime which violates the Internet proper user policy [27]. DoS floods the network and prevent valid SiteWatcher Client user to access the server. Another issue is that we do not want to expose phishing data to unknown users in order to prevent data abuse.

After several meetings among the AntiPhishing Group, it is confirmed to limit the access right of the Client Framework. Only SiteWatcher Client users with valid license code can access the framework. It can be done in the code level by checking the license code at the beginning of a function.

The implementation (the Client Framework interface Service1.asmx) below verifies users before downloading the Protected List. The download task will not be proceeded
if the user cannot pass the license identification. And thus it reduces the burden of DoS attacks and data abuse.

```csharp
[WebMethod]
public DataSet dsGetProtected(string license)
{
    // check the license code of a user
    if (SWLicenseManager.InvalidateLicense(license) &&
        LicenseDB.GetLicenseExpired(license) != "EXPIRED" && … )
    {
        ProtectedDB objProtected = new ProtectedDB();
        // download the latest Protected List
        DataSet objDataSet = objProtected.dsGetProtected();
        // return result if it is a valid request
        return objDataSet;
    }
    // return nothing if it is an invalid request
    return null;
}
```

**Code Snippet 4: User identification in the Client Framework**

Besides, the 3-tier architecture enhances the security of the Client Framework. According to the Figure 4.3, SiteWatcher Client can only access the Presentation Layer (Client) of the Client Framework. And also, there is no delete function provided in the Client Framework such that it avoids mis-use of the functionalities. The business logic layer and the data access layer are encapsulated under the 3-tier architecture. See *Chapter 4.4 SiteWatcher Server Architecture* for more details about 3-tier architecture.

### 4.5.2.2 Admin Framework

As the Admin Framework provides a wider range of functionalities such as deletion, updating and performing threshold training, the security is restricted that only SiteWatcher Server administrators or developers can access the framework.

To enhance the security, we rely on Microsoft Internet Information Services (IIS). For those administrators use the Admin Framework by directly accessing the SiteWatcher Server machine, window authentication is required. Administrators should be registered in the server and they have to fill in correct username and password before accessing the Admin Framework. Administrators can use, read and modify both Client and Admin Frameworks in this way.
If administrators access the framework through HTTP protocol (i.e. add the web reference of the framework), authentication is also required. Administrators should be registered in the Server machine. However, administrators can only use the Client and Admin Frameworks in this way.

4.5.2.3 Both Client and Admin Frameworks

As a web server, SiteWatcher Server is possibly experiencing SQL injection attack. SQL injection attacks allow hackers to “inject” SQL statements in the SQL commands to get private or server information [28].

To tackle the SQL injection attack, both Client and Admin Frameworks use strongly typed parameters with SQL statement. The following Business Layer (Admin) code shows the strongly typed parameter with SQL statement to prevent SQL injection.

```csharp
public DataSet dsGetSingleProtected(string strProtectedURL)
{
    // Connection string
    string strConnection = ConfigurationSettings.AppSettings["PhishingDB"];
    OdbcConnection objConnection = new OdbcConnection(strConnection);
    StringBuilder sb = new StringBuilder();

    // SQL statement
    sb.Append("select * from ProtectedDB where url = ? ");
    OdbcDataAdapter objAdapter = new OdbcDataAdapter(sb.ToString(), objConnection);

    // setting the type of the parameter, to prevent SQL injection
    objAdapter.SelectCommand.Parameters.Add("@url", OdbcType.NVarChar, 255, "url");
    objAdapter.SelectCommand.Parameters["@url"].Value = strProtectedURL;

    DataSet objDataSet = new DataSet();
    objAdapter.Fill(objDataSet);
    return objDataSet;
}
```

**Code Snippet 5: Strongly typed parameter with SQL statement**

In this case, the @url parameter is treated as a literal value and not as execute code. The type and length of @url parameter is also checked. If the data does not satisfy the type and length defined by the parameter, exceptions are thrown. No information is exposed under the SQL injection attack.
4.5.3 SiteWatcher Scheduler Security

Since there is no presentation layer for users to access (See Figure 4.3), there is no way for users to access through Internet. Administrators need to access the SiteWatcher Server machine directly to read and modify it. Window authentication is required and the administrators should be registered in the server in order to get the permission to access the server.

4.6 Database

All data for the SiteWatcher Server is stored in a MySQL Server database. MySQL database is chosen because of its consistent fast performance, high reliability and ease of use [29]. This chapter provides an overview of the database used in the SiteWatcher Server.

4.6.1 Database Schema

The SiteWatcher database schema is a very simple schema that contains a simple repository for Lists of URLs and tables to store Users and User Roles. The physical schema is shown in Figure 4.7.
Different table of Lists to store URLs with different aspect, they includes:
1. Suspected List
2. Protected List with it Sensitive Words
3. Black List with its Connection Status
4. White List
5. White Domain List
6. Customized Protected Lists with its Sensitive Words and reported info

RoleDB is a lookup table with the following values:
1. Senior Administrator
2. Administrator

LicenseDB store the SiteWatcher Client users

Figure 4.7: SiteWatcher Server database schema
4.6.2 Database Description

As we do not focus on the database design, this chapter will briefly describe the SiteWatcher database table. Table 4.1 shows the tables with description.

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ProtectedDB</td>
<td>Stores official protected URLs and threshold value.</td>
</tr>
<tr>
<td>SenWordDB</td>
<td>Stores sensitive words and the relation to protected URLs.</td>
</tr>
<tr>
<td>BlackDB</td>
<td>Stores confirmed phishing URLs, the Black URLs.</td>
</tr>
<tr>
<td>BlackStatusDB</td>
<td>Stores the records of connectivity of Black URLs.</td>
</tr>
<tr>
<td>SuspectedDB</td>
<td>Stores the suspected phishing URLs and EMD value reported by the SiteWatcher Client.</td>
</tr>
<tr>
<td>WhiteDB</td>
<td>Stores confirmed safe URLs.</td>
</tr>
<tr>
<td>WhiteDomainNameDB</td>
<td>Stores confirmed safe domain name.</td>
</tr>
<tr>
<td>CustomizedProtectedDB</td>
<td>Stores “user defined” protected URLs reported by the SiteWatcher Client.</td>
</tr>
<tr>
<td>CustomizedSenWordDB</td>
<td>Stores “user defined” sensitive words and the relation to customized protected URLs reported by SiteWatcher Client.</td>
</tr>
<tr>
<td>CustomizedRecordDB</td>
<td>Stores the user records in reporting customized protected URLs and sensitive words.</td>
</tr>
<tr>
<td>UserDB</td>
<td>Stores users who are allowed to access the SiteWatcher Server.</td>
</tr>
<tr>
<td>RoleDB</td>
<td>Stores the role of users in UserDB.</td>
</tr>
<tr>
<td>StatusDB</td>
<td>Stores the status.</td>
</tr>
<tr>
<td>LicenseDB</td>
<td>Store user’s SiteWatcher Client license code.</td>
</tr>
<tr>
<td>Version</td>
<td>Stores the version of the SiteWatcher Server database.</td>
</tr>
</tbody>
</table>

Table 4.1: SiteWatcher Server database table
4.7 PhishingDefService Features and Functionalities

After studying design, architecture, security of the SiteWatcher Server, this chapter will discuss about the functionality of the PhishingDefService web service.

As it is discussed in Chapter 4.2.2 Detailed Architecture, PhishingDefService is treated as the business logic layer of the SiteWatcher Server. It aims in providing PhishingDefService Client and Admin Frameworks for development, so there is absent of user interface in the web service.

4.7.1 PhishingDefService Client Framework

The Client Framework provides functions for SiteWatcher Client to interact with SiteWatcher Server. Figure 4.8 shows the class diagram of the Client Framework. Because of the clear separation of layers of code, classes in the Business Logic Layer (Client) are not concerned.

![Class diagram](image)

Figure 4.8: Client Framework class diagram
Server Side Support for SiteWatcher, an Antiphishing Software

The ClientService is the only class in the Presentation Layer (Client). As the functionalities used by the SiteWatcher Client are limited, ClientService is enough to be stand alone class for SiteWatcher Client developers to implement with.

SiteWatcher Client can call the ClientService class to:

- Download the latest confirmed lists including Protected, Sensitive Word, Black, White, and White Domain Name Lists from SiteWatcher Server.
- Report suspected URLs with EMD value to the SiteWatcher Server.
- Report Customized Protected URLs with Customized Sensitive Word to the SiteWatcher Server.
- Check the version of the SiteWatcher Server database.
- Check, acquire and give up the license of the SiteWatcher Client.

4.7.2 PhishingDefService Admin Framework

The Admin Framework is the base framework for authorized SiteWatcher Server administrators or developers. Based on this framework, developers can implement other SiteWatcher sub-system such as SiteWatcher Schedule and SiteWatcher Administration Webpage. Figure 4.9 shows the class diagram of the Admin Framework. Due to the 3-tier architecture, we only focus on classes in the Presentation Layer (Admin) of PhishingDefService.
Unlike the Client Framework, many classes are designed in the Presentation Layer (Admin) of the PhishingDefServer due to the advanced functionalities provided by the
Admin Framework. As only authorized users can use this framework, functions such as updating and deletion of data are provided for the development of SiteWatcher Server. Table 4.2 briefly describes the classes in the Presentation Layer (Admin) of the framework.

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ProtectedDB</td>
<td>Allows select, insert, update and delete records in the Protected List.</td>
</tr>
<tr>
<td>SenWordDB</td>
<td>Allows select, insert, update and delete records in the Sensitive Word List.</td>
</tr>
<tr>
<td>SuspectedDB</td>
<td>Allows select, update and delete records in the Suspected List.</td>
</tr>
<tr>
<td>BlackDB</td>
<td>Allows select, insert, update and delete records in the Black List.</td>
</tr>
<tr>
<td>BlackStatusDB</td>
<td>Allows select connectivity records of the Black List.</td>
</tr>
<tr>
<td>WhiteDB</td>
<td>Allows select, insert, update and delete records in the White List.</td>
</tr>
<tr>
<td>WhiteDomainNameDB</td>
<td>Allows select, insert, update and delete records in the White Domain Name List.</td>
</tr>
<tr>
<td>CustomizedProtectedDB</td>
<td>Allows select, update and delete records in the Customized Protected List.</td>
</tr>
<tr>
<td>CustomizedSenWordDB</td>
<td>Allows select, update and delete records in the Customized Sensitive Word List.</td>
</tr>
<tr>
<td>CustomizedProtectedRecordDB</td>
<td>Allows select and delete records of user report.</td>
</tr>
<tr>
<td>ThresholdTraining</td>
<td>Provides threshold training to Protected URLs.</td>
</tr>
<tr>
<td>UserDB</td>
<td>Allows select, insert and update users.</td>
</tr>
<tr>
<td>RoleDB</td>
<td>Get the roles of the SiteWatcher Server</td>
</tr>
</tbody>
</table>

Table 4.2: Admin Framework class description

With the Admin Framework, it promotes code re-use and SiteWatcher Server developers can maintain and enhance the code easier. Development is also easier due to the clear separation of layers.
4.8 SiteWatcher Scheduler Features and Functionalities

SiteWatcher Scheduler is a sub-system implemented based on the PhishingDefService Admin Framework. It is a timer in which it can perform certain tasks in a time interval. This chapter will discuss the automated tasks and the scheduler configuration page.

4.8.1 Auto-Update Black List

Reporting by the SiteWatcher Client is one way to grow the Black List. However, it is a passive way to collect Black URLs and it requires administrators to confirm the URLs. The ideal way is that the SiteWatcher Server can look for confirmed phishing URLs and insert these URLs into the database automatically.

Auto-Update Black List (Auto-Update in short form) is designed for the issue mentioned above. It is one of the automated tasks scheduled by the SiteWatcher Schedule. Auto-Update timely checks for the phishing URLs in PhishTank which is “collaborative clearing house for data and information about phishing on the Internet [30].” Figure 4.10 shows how Auto-Update gets the phishing URLs from PhishTank.

![Figure 4.10: Auto-Update Black List process](image-url)
Auto-Update does not always download all the phishing URLs from PhishTank. To reduce the traffic between the servers:

1. Auto-Update starts and sends a HTTP request to the PhishTank Server for requesting preprocessing.xml which is a XML file containing the number of latest phishing URLs stored in the PhishTank Server. Figure 4.11 shows the contents of the preprocessing.xml. The `<total_entries>` node stores the amount of the latest phishing URLs in the PhishTank Server.

   ```xml
   <?xml version="1.0" encoding="utf-8"?>
   <output>
     <meta>
       <generated_at>2007-04-13T17:30:33+00:00</generated_at>
       <total_entries>22973</total_entries>
     </meta>
   </output>
   ```

   Figure 4.11: Contents in preprocessing.xml

2. The preprocessing.xml is downloaded and stored in the harddisk of the SiteWatcher Server. Auto-Update compares the number of URLs in the newly download preprocessing.xml with the old XML file which is downloaded during the last Auto-Update process. If they are the same, Auto-Update stops and waits for next time interval.

3. If different numbers of URLs are found in the 2 XML files, Auto-Update sends another request to the PhishTank Server for requesting another XML file named as index.xml. This XML file contains a list of latest phishing URLs confirmed by PhishTank. Figure 4.12 shows part of the contents in the index.xml. The `<url>` node stores the confirmed phishing URL.
4. The index.xml is downloaded and stored in the harddisk of the SiteWatcher Server.

5. Auto-Update compares the lists of phishing URLs with the Black List to remove duplicate records. These new unique phishing URLs are inserted into the Black List in the SiteWatcher Server. The Black List is said to be the latest version.

6. Auto-Update stops and waits for next time interval.

In this way, Auto-Update can actively “search” and download phishing URLs without any human activity involved.

4.8.2 Auto-Check Black List

According to the report conducted by APWG in 2007, the average time online for a phishing website is 4 days and the longest time online for the phishing website is 30 days [1]. As phishing websites “die” in a short time, many Black URLs stored in the Black List of the SiteWatcher Server are said to be “disconnected”. Thanks to the reporting by SiteWatcher Client users and Auto-Update (See Chapter 4.8.1 Auto-Update Black List), the Black List grows rapidly and it is difficult for SiteWatcher Server administrators to check all Black URLs. As a result, these “disconnected” Black URLs consume lots of space in the database. And thus SiteWatcher Client users suffer from
spending lots of time to wait the client application to download a huge list of useless “disconnected” Black URLs.

To replace human judgment in determining the great list of Black URLs, Auto-Check Black List (Auto-Check in short form) is designed as a result. Auto-Check is a scheduled task raised by the SiteWatcher Scheduler. It aims in checking the connectivity of Black URLs and defining the Black URLs as “disconnected” if necessary. Figure 4.13 shows the Auto-Check process.

![Figure 4.13 Auto-Check Black List process](image)

To conduct the connectivity analysis of Black URLs:

1. Auto-Check starts and sends a request through HTTP for getting the HTTP status code of a specific Black URL from the phishing web server.

2. HTTP status code is received by the SiteWatcher Server. Figure 4.14 shows the status codes received from Black URLs.
3. Auto-Check records all “disconnect” record in the database. If a Black URL is found that it has been disconnected continuously for certain times (say 10 times), the Black URL will be updated as “disconnected”.

After each checking, the Auto-Check stops and waits for next time interval. Those “disconnected” Black URLs will not be downloaded by SiteWatcher Client and they will be automatically permanent deleted by another scheduled task, Auto-Delete, which will be discussed in Chapter 4.8.3 Auto-Delete.

With the help of Auto-Check, the Black List is always kept most updated with alive Black URLs without human involvement. It also eases the growth of Black List and the download traffic between SiteWatcher Server and Client.

4.8.3 Auto-Delete

Since we do not encourage SiteWatcher administrators to permanent delete any record to avoid deleting important records by mistakes, they are recommended to change the status of the records to “deleted” instead. The benefits of changing the status of records are that it keeps the records in the database and others such as SiteWatcher Client will not use and download them anymore. But it leads the growth of the database since those records are not deleted actually.

Therefore, Auto-Delete is designed as a scheduled task of the SiteWatcher Scheduler. All records with status “deleted” will be permanent deleted by Auto-Delete. Figure 4.15 and 4.16 shows the Auto-Delete process displayed in the SiteWatcher Administration Webpage. The suspected URLs with status “deleted” is permanent deleted when Auto-Delete is raised.
Apart from deleting the “deleted” records, Auto-Delete is also applied to the “disconnected” Black URLs which have been discussed in Chapter 4.8.2 Auto-Check Black List. Auto-Delete will delete all “disconnected” Black URLs when Auto-Check defines them as “disconnected”.

4.8.4 Reporting

Reporting is important because it helps administrators and developers to keep trace the system. Once errors or abnormal situations occur, they can be noticed at once and fix the problem. It minimizes the impact to both the system and users. Besides, reporting helps developers to discover bugs when running the system.

Reporting is applied to the SiteWatcher Scheduler and it is shown in Figure 4.17. When scheduled tasks including Auto-Update, Auto-Check and Auto-Delete complete their own processes, a log is recorded in the SiteWatcher Server machine and an email is sent to the administrators.
Figure 4.18 is the sample of Auto-Delete report. An event log and the email are produced. The event log can be viewed in the SiteWatcher Server. SiteWatcher Server administrators can read the reports through email.

Figure 4.18: Auto-Delete event log and email report
4.8.5 Configuration Page

The Configuration Page allows SiteWatcher Server administrators to configure the SiteWatcher Scheduler as needed. Authentication and authorization are required to access the page. See Chapter 4.5 Security for more details on security. Figure 4.19 shows the Configuration Page.

![Configuration Page Screenshot]

Figure 4.19: SiteWatcher configuration page

The Configuration Page allows administrators to:

- Set the connection string of the database server
- Set the time interval of Auto-Delete, Auto-Update and Auto-Check
- Set the definition of “disconnected” Black URLs
- Set the SMTP server
- Set report email from and to addresses
The values of the setting are stored in the web.config file in the PhishingDefService project. It is an XML file containing all the configuration of the PhishingDefService. In the Configuration Page, administrators are limited to configure the SiteWatcher Scheduler. Figure 4.20 shows the value of setting of the SiteWatcher Scheduler in the web.config XML file.

![Figure 4.20: configuration values in web.config](image)

When the setting is saved in the Configuration Page, the SiteWatcher Schedule will be restarted automatically with the new setting. With the Configuration Page, administrators can configure the SiteWatcher Schedule without access the SiteWatcher Server machine directly.

### 4.9 SiteWatcher Administration Webpage Features and Functionalities

SiteWatcher Administration Webpage allows administrators to keep monitoring and control all the lists stored in the SiteWatcher Server in their own computers without directly accessing the server machine. Authentication and authorization is required before access the content page of the webpage (See Chapter 4.5.1 SiteWatcher Administration Webpage Security). Figure 4.21 shows the login page of the SiteWatcher Administration Webpage.
4.9.1 Suspected List Page

The Suspected List Page displays all suspected URL records from the Suspected List in the server. When a potential suspected webpage is found in the client’s browser tool, the SiteWatcher Client starts phishing detection. If the client application cannot find the suspected URL in its Black List and it determines the suspected URL as a phishing attack by Visual Similarity analysis, the suspected URLs and its records including the EMD value will be sent to the Suspected List of the SiteWatcher Server.

When suspected URL records are received and stored in the SiteWatcher Server database, administrators can view these records through Suspected List Page. Initially,
only new suspected URL records are shown in the page, but administrators can view the analyzed records by the filter provided in the page. Figure 4.22 is the Suspected List Page with new suspected URL records.

![Suspected List Page](image)

**Figure 4.22: Suspected List Page**

### 4.9.1.1 Actions in the Suspected List Page

In the page, administrators can determine URLs by the actions shown in Table 4.3:

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Define as a phishing attack</td>
<td>The URL is a phishing attack. These URLs will be transferred and stored in the Black List. Threshold training is performed to the related Protected URL during the action.</td>
</tr>
<tr>
<td>Define as a non-phishing attack</td>
<td>The URL is safe. The whole URL is stored in the White List. Threshold training is performed to the related Protected URL during</td>
</tr>
</tbody>
</table>
### Table 4.3: Actions in Suspected List Page

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Define as non-phishing attacks</td>
<td>All URLs from this domain name are safe. The domain name of the URL (i.e. <a href="http://www.domain.com">www.domain.com</a>) is stored in the White Domain Name List. Threshold training is performed to the related Protected URL during the action.</td>
</tr>
<tr>
<td>and safe domain name</td>
<td></td>
</tr>
<tr>
<td>Delete</td>
<td>The URL may be a duplicate or administrators define it as a “useless” record. The status of record is changed to “deleted”. The URL record will be permanent deleted by Auto-Delete or administrators.</td>
</tr>
<tr>
<td>Permanent delete</td>
<td>Administrators can choose to permanent delete records manually. But it is not recommended.</td>
</tr>
</tbody>
</table>

#### 4.9.1.2 Auto Correction in the Suspected List Page

Besides, when the administrators do the action mentioned above in the Table 4.3, auto correction is performed to reduce human mistakes. The auto correction prevents suspected URLs to:

- Store in the Black List if it has already been stored into Protected, White and White Domain Name List.
- Store in the White, White Domain Name List if it has already been stored in the Black List.

#### 4.9.1.3 Threshold Training

When actions mentioned in Table 4.3 are made by the administrators, threshold training runs and the threshold value of related Protected URLs will be adjusted automatically.

Threshold training makes use of the EMD value calculated by Visual Similarity in the SiteWatcher Client. EMD is the difference in appearance between 2 websites where

\[
0.0 \leq \text{EMD} \leq 1.0
\]

When the EMD tends to 0, it means that the difference between 2 websites is small. And thus they are similar.
Threshold value is the similarity of 2 websites where

\[ 0.0 \leq \text{Threshold} \leq 100.0 \text{ and} \]

\[ \text{Threshold} = (1.0 - \text{EMD}) \times 100 \]

When the threshold value tends to 100, it means that 2 websites are 100% the same. In the SiteWatcher Server, we would focus on threshold value rather than EMD. Figure 4.23 is a sample of EMD and threshold value.

![Figure 4.23: A sample of EMD and threshold](image)

Threshold training is applied in order to get the most accurate threshold value of the protected URL so that the possibility of false alarm (alert message) raised by the SiteWatcher Client is minimized. The false alarm includes false positive and false negative. Below is the algorithm proposed by Anthony [20] to get the minimum false alarm:

Let \( P \) be a specific Protected URL, with the current threshold value \( T_p \),
Let $S_i$ be the $i^{th}$ suspected URL related to $P$,

Let $T_i$ be the threshold value calculated by $S_i$ and $P$ so that

\[
(P, S_i) = \{T_1, T_2, \ldots, T_i\} \quad \text{and} \quad T_{p\ \text{new}} = \min \ (S_i, P)
\]

According to Anthony, $T_{p\ \text{new}}$ is said to be stable when 20 or more threshold training has been performed. In SiteWatcher Server, the threshold training of a specific protected URL is completed when it has performed 25 times of training. After the completion of training, the status of the protected URL is changed from “Training” to “Release”.

### 4.9.2 Black List Page

The Black List Page displays the confirmed Black URLs in the Black List of the SiteWatcher Server. Administrators are allowed to monitor and change the status of the Black URL if necessary. Figure 4.24 shows the Black List Page.
4.9.2.1 Actions in the Black List Page

In the page, administrators can determine URLs by the actions shown in Table 4.4:

<table>
<thead>
<tr>
<th>Actions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Release</td>
<td>The URL is defined as a phishing attack and it is released for SiteWatcher Client to download.</td>
</tr>
<tr>
<td>Delete</td>
<td>The URL is disconnected or administrators define it as a “useless” record. The status of record is changed to “deleted”. The record will be permanent deleted by Auto-Delete or administrators.</td>
</tr>
<tr>
<td>Permanent delete</td>
<td>Administrators can choose to permanent delete records manually. But it is not recommended.</td>
</tr>
<tr>
<td>Add</td>
<td>Black URL can be added by administrators manually. Figure 4.25 demonstrates the Add Black List Page. The added Black URL will belong to UnAssigned group where it is a phishing URL but it does not match any Protected URL.</td>
</tr>
</tbody>
</table>
View Connectivity Report

Administrators can view the connectivity report of each Black URL. Figure 4.26 shows the connectivity report.

Table 4.4: Actions in the Black List Page

Figure 4.25: Add Black URL Page

Figure 4.26: Connectivity report of a Black URL
4.9.3 White List Page

White List Page displays all confirmed White URLs in the White List stored in the SiteWatcher Server. Administrators are allowed to monitor and change the status of the White URL if necessary. Figure 4.27 shows the White List Page.

![White List Page](image)

Figure 4.27: White List Page

4.9.3.1 Actions in the White List Page

In the page, administrators can determine URLs by the actions shown in Table 4.5:

<table>
<thead>
<tr>
<th>Actions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Release</td>
<td>The URL is defined as safe and it is released for SiteWatcher Client to download.</td>
</tr>
<tr>
<td>Delete</td>
<td>The URL may be disconnected or administrators define it as a “useless” record. The status of record is changed to “deleted”. The</td>
</tr>
</tbody>
</table>
Server Side Support for SiteWatcher, an Antiphishing Software

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Permanent delete</td>
<td>Administrators can choose to permanent delete records manually. But it is not recommended.</td>
</tr>
<tr>
<td>Add</td>
<td>White URL can be added by administrators manually. Figure 4.25 demonstrates the Add White List Page.</td>
</tr>
</tbody>
</table>

Table 4.5: Actions in the White List Page

Figure 4.28: Add White List Page

4.9.4 White Domain Name List Page

White Domain Name List Page displays all the confirmed White Domain (i.e. www.domain.com) in the White Domain Name List stored in the SiteWatcher Server. Administrators are allowed to monitor and change the status of the White Domain Name if necessary. Figure 4.29 shows the White List Page.
4.9.4.1 Actions in the White Domain Name List Page

In the page, administrators can determine the domain name by the actions shown in Table 4.6:

<table>
<thead>
<tr>
<th>Actions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Release</td>
<td>The domain name is defined as safe and it is released for SiteWatcher Client to download.</td>
</tr>
<tr>
<td>Delete</td>
<td>The domain name may be disconnected or administrators define it as a &quot;useless&quot; record. The status of record is changed to &quot;deleted&quot;. The record will be permanent deleted by Auto-Delete or administrators.</td>
</tr>
<tr>
<td>Permanent delete</td>
<td>Administrators can choose to permanent delete records manually. But it is not recommended.</td>
</tr>
</tbody>
</table>
White Domain Name can be added by administrators manually. Figure 4.30 demonstrates the Add White Domain Name List Page.

Table 4.6: Actions in the White Domain Name List Page

![Add White Domain Name List Page](image)

**Figure 4.30: Add White Domain Name List Page**

### 4.9.5 Protected List Page

The new version of SiteWatcher Client defines that Protected List stores the official defined Protected URLs. Protected List Page displays all the confirmed Protected URLs in the Protected List stored in the SiteWatcher Server. Administrators are allowed to monitor and change the status of the White URL if necessary. Figure 4.31 shows the Protected List Page.
4.9.5.1 Actions in the Protected List Page

In the page, administrators can determine URLs by the actions shown in Table 4.7:

<table>
<thead>
<tr>
<th>Actions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Release</td>
<td>The URL is defined as safe and it is released for SiteWatcher Client to download.</td>
</tr>
<tr>
<td>Training</td>
<td>Reset the threshold of the Protected URL and start the threshold training again.</td>
</tr>
<tr>
<td>Delete</td>
<td>The URL may be disconnected or administrators define it as a “useless” record. The status of record is changed to “deleted”. The record will be permanent deleted by Auto-Delete or administrators.</td>
</tr>
<tr>
<td>Permanent delete</td>
<td>Administrators can choose to permanent delete records manually. But it is not recommended.</td>
</tr>
</tbody>
</table>
Edit

The threshold value of a Protected URL can be set by administrators. Besides, the domain name of the Protected URL can be added into the White Domain Name List. Figure 4.32 shows the Edit Protected URL Page.

View/Add Sensitive Words

Administrators can view and add Sensitive Words to a specific Protected URL. Figure 4.33 and 4.34 show the View Sensitive Word Page and Add Sensitive Word Page respectively.

View/Add Black URLs

Administrators can view and add Black URLs related to a specific Protected URL. Figure 4.35 and 4.36 show the View Protected-Black List Page and Add Protected-Black URL Page respectively.

Table 4.7: Actions in the Protected List Page

![Figure 4.32: Edit Protected URL Page](image)
Figure 4.33: View Sensitive Word Page

Figure 4.34: Add Sensitive Word Page
4.9.6 Customized Protected List Page

The new version of SiteWatcher Client defines that Customized Protected List stores user’s defined Protected URLs. Customized Protected List Page displays all the Customized Protected URLs in the Customized Protected List stored in the SiteWatcher Server. Administrators are allowed to monitor and change the status of
the White URL if necessary. Besides, administrators can define the Customized Protected URL as official Protected URL for all SiteWatcher Client to download. Figure 4.37 shows the Customized Protected List Page.

![Customized Protected List Page](image)

**Figure 4.37: Customized Protected List Page**

### 4.9.6.1 Actions in the Customized Protected List Page

In the page, administrators can determine URLs by the actions shown in Table 4.8:

<table>
<thead>
<tr>
<th>Actions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Release</td>
<td>The URL is defined as safe. Administrators define it as an official Protected URL. The URL is copied to Protected List for all SiteWatcher Client users to download. Note that the Customized Sensitive Words will not be released to public in this way.</td>
</tr>
<tr>
<td>Release with Sensitive Words</td>
<td>The URL is defined as safe. Administrators define it as an official Protected URL. Both URL and related sensitive words are copied to</td>
</tr>
</tbody>
</table>
Protected List for all SiteWatcher Client users to download.

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delete</td>
<td>The URL may be disconnected or administrators define it as a “useless” record. The status of record is changed to “deleted”. The record will be permanent deleted by Auto-Delete or administrators.</td>
</tr>
<tr>
<td>Permanent delete</td>
<td>Administrators can choose to permanent delete records manually. But it is not recommended.</td>
</tr>
<tr>
<td>View Customized Sensitive Words</td>
<td>Administrators can view, release and delete Customized Sensitive Words of a specific Customized Protected URL. Figure 3.38 shows the View Customized Sensitive Word Page.</td>
</tr>
<tr>
<td>View User Report</td>
<td>Administrators can view the user report of a specific Customized Protected URL. Figure 3.39 shows the View User Report Page.</td>
</tr>
</tbody>
</table>

Table 4.8: Actions in the Customized Protected List Page

Figure 4.38: View Customized Sensitive Word Page
4.9.7 User Profile

The User Profile Page displays the user information of administrators such as username and role. Administrators can change their password in this page. It is recommended to change the password frequently to prevent being hacked. Besides, administrators with enough permission can create administrator account in this page. Figure 4.40 shows the User Profile Page.
4.9.8 Configuration Page

The SiteWatcher Administration Webpage redirects administrators to the SiteWatcher Scheduler Configuration Page (See Chapter 4.8.5 Configuration Page). It allows administrators to configure the SiteWatcher Scheduler when necessary.

4.9.9 Help File

A help file is provided for all administrators to get familiarized with the use of SiteWatcher Administration Page. Figure 4.41 shows the Help File.
Server Side Support for SiteWatcher, an Antiphishing Software

Help

Overview

The Help File provides you with the overview of the SiteWatcher Server. Please read it before using the SiteWatcher Server. Note that only SiteWatcher Administrators can access the SiteWatcher Server.

As the server aids support for the SiteWatcher Client, the SiteWatcher Server serves as collecting and updating the phishing data for the SiteWatcher Client. Make sure that the connection of the SiteWatcher Server and the database are alive, otherwise the SiteWatcher Client cannot connect to the server to perform updating.

- Suspected List
- Black List
- White List
- White Domain Name List
- Protected List
- Custom Protected List
- User Profile
- Configuration Page

Figure 4.41: Help File
5. Result and Discussion
5.1 SiteWatcher Server

SiteWatcher Server is a large phishing data center. Besides, the server has to get the latest phishing URLs from PhishTank. It is important that Auto-Update can get as much phishing URLs as it can and it is done in an expected time. Therefore, Auto-Update Black List performance is tested to estimate the performance.

5.1.1 Auto-Update Black List Performance Evaluation

Auto-Update performance test has been carried out in the testing machine. Before starting the test, the Black List is empty in the testing machine and the Auto-Update is configured to run 1 time per day. Table 6.1 shows the testing result.

<table>
<thead>
<tr>
<th>Date (2007)</th>
<th>URLs download from PhishTank</th>
<th>URLs inserted into Black List</th>
<th>Insertion rate</th>
<th>Process Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>8 April</td>
<td>12078</td>
<td>12078</td>
<td>100%</td>
<td>19 min 3 sec</td>
</tr>
<tr>
<td>9 April</td>
<td>12131</td>
<td>784</td>
<td>6.4%</td>
<td>17 min 37 sec</td>
</tr>
<tr>
<td>10 April</td>
<td>12204</td>
<td>2754</td>
<td>22.6%</td>
<td>16 min 23 sec</td>
</tr>
<tr>
<td>11 April</td>
<td>15094</td>
<td>3064</td>
<td>14.5%</td>
<td>20 min 21 sec</td>
</tr>
<tr>
<td>12 April</td>
<td>21083</td>
<td>6423</td>
<td>30.5%</td>
<td>33 min 54 sec</td>
</tr>
<tr>
<td>13 April</td>
<td>22416</td>
<td>1576</td>
<td>7.0%</td>
<td>33 min 28 sec</td>
</tr>
<tr>
<td>14 April</td>
<td>23536</td>
<td>2014</td>
<td>8.6%</td>
<td>35 min 17 sec</td>
</tr>
</tbody>
</table>

Table 5.1 Auto-Update test result

After the performance test, some interesting findings are found:

- The insertion rate decreases significantly after the first download
- The XML files size downloaded is kept on increasing
- The process time is proportional to the number of URLs downloaded

Auto-Update runs successfully, new phishing URLs are inserted into the SiteWatcher Server database. But we may be careful on the process time as it depends on the number of URLs downloaded. If PhishTank inserts over 100,000 URLs in the XML file, it requires much more time for Auto-Update to process.
5.2 SiteWatcher Client with SiteWatcher Server Support

SiteWatcher Server is designed to provide phishing information and threshold value for performing Visual Similarity in SiteWatcher Client. Therefore it is necessary to know how SiteWatcher Client is improved with the support of SiteWatcher Server.

5.2.1 SiteWatcher Client Performance Evaluation

SiteWatcher Client performance test has been carried out in the testing machine. Firefox 1.5 is used as the testing environment as it does not contain any anti-phishing tool. We choose phishing URLs in Black List of the SiteWatcher Server and Google’s Blacklist [31] as the testing sample resource. In each list, we select 30 phishing URLs randomly to test. Test is performed to compare the phishing detection rate of the SiteWatcher Client before and after updating from the SiteWatcher Server. Table 5.2 and 5.3 show the testing results respectively.

<table>
<thead>
<tr>
<th>Source of phishing URLs</th>
<th>Phishing detection rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>SiteWatcher Server</td>
<td>26.7%</td>
</tr>
<tr>
<td>Google’s Blacklist</td>
<td>13.3%</td>
</tr>
</tbody>
</table>

Table 5.2 Phishing detection rate before updating from server

<table>
<thead>
<tr>
<th>Source of phishing URLs</th>
<th>Phishing detection rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>SiteWatcher Server</td>
<td>100.0%</td>
</tr>
<tr>
<td>Google’s Blacklist</td>
<td>30.0%</td>
</tr>
</tbody>
</table>

Table 5.3 Phishing detection rate after updating from server

The testing results conclude that the phishing detection of the SiteWatcher Client is improved after updating from the SiteWatcher Server. Besides, the response time for SiteWatcher Client is shorten because more phishing URLs are found in its Black List. And it avoids the Visual Similarity analysis which is a time consumed process. However, the testing result may not be accurate. The phishing detection rate may be affected by:

- The protected URLs do not have enough sensitive words to detect the phishing websites. For example, a phishing website is pretending Paypal official website, which is stored as a protected URL with sensitive word (paypal, pay pal) in the
database of SiteWatcher Client. However, the phishing website does not contain any sensitive word in the web (it may use “poypal” and “poy pal” instead). Therefore SiteWatcher Client treats it as a safe website.

- Non-trained threshold value
- Bugs exist in the Visual Similarity performed by the SiteWatcher Client
- Sample size of the testing data

Although there are interferences mentioned about, the results at least show that SiteWatcher Client detects more phishing websites and performs faster. To get a more accurate result, it is recommended to perform further tests in testing the both SiteWatcher Server and Client.
6. Conclusion and Future Work
6.1 Critical Review

At last, I did it! The SiteWatcher Server is running in the production server smoothly and the final year project is end up with this paper. However, this end does not mean that the SiteWatcher Server project ends. SiteWatcher Server should evolve into a more stable and advanced background support of the SiteWatcher Client to fight against the complicated phishing attacks.

The greatest failure of this project is that we are not doing tests actively. Unit tests are done only by the developer himself. It is a bad practice as we always assume that our code does not contain bugs. Under this situation, I faced problems when I did the integration with other’s code. Since no one knew whose code produces bugs, I had to debug every line in the integration code. It was an extremely hard job as I had to trace my member’s code too! Although the bugs are fixed most of the time, it is not worth spending time to debug in this way.

During the final year project, I am proud of development of the PhishingDefServer Frameworks. I used to develop projects with 3-tier architecture, but it is the first time I extend the idea of the architecture to make the frameworks. Although the frameworks provide simple functionalities, it can be applied to all SiteWatcher project. I believe that the frameworks would help the FYP students next year in further development of the SiteWatcher Server.

My knowledge is much broadened and I gain experience from other people. First of all is my great supervisor Dr. Liu. Others including my group members (Rudolf, Bryan, Robert, Sherman, Mansion and Albert) and many .NET developers around the world. I have learnt much from them by sharing ideas. These knowledge and experience are much valuable than learning from books.
6.2 Achievement

Thanks to my placement work in a software house, I noticed the development of the SiteWatcher Client from my manager. And I tried the SiteWatcher Client beta version before my final year project. I found that the application was really great. But it is lack of server support. In the early April 2006, I met with Dr. Liu and showed my interest in developing a server support for the SiteWatcher Client. Although it is a rough idea, Dr. Liu kindly accepted my project plan. It is the first success of my final year project.

The first version of SiteWatcher Server was released together with the upgraded SiteWatcher Client in middle of January 2007. It was recorded that there were around 10 accounts (not testing accounts!) were registered the SiteWatcher Client and they were using the services provided by the SiteWatcher Server. It is the second success.

In the first version of SiteWatcher Server, the SiteWatcher Administration Webpage and PhishDefService were independent projects. The PhishDefService only provided services to the SiteWatcher Client. However, I found that these 2 projects could share some of the codes actually. Therefore the idea of PhishDefService Frameworks was proposed. I benefited from the frameworks in the later SiteWatcher Server development. It is the third success.

In February 2007, SiteWatcher Scheduler was introduced to reduce workload of administrators. With the frameworks, the scheduler was drafted in a short time. Originally, the scheduler only served as cleaning up “useless” records in the server. Thanks to Anthony, the idea of “searching” phishing data was generated. And finally, the scheduler performed Auto-Update, Auto-Check and Auto-Delete. The Auto-Update greatly promotes the growth of the Black List. And the Auto-Check and Auto-Delete remove useless records from database. SiteWatcher Clients benefit from these features. It is the forth success.

Today, more than 130 users have registered the SiteWatcher Client. I am glad with the success of increased number of users due to the enhanced SiteWatcher Client and Server. The success is not mine, but all the AntiPhishing Group who have spent much efforts in the project.
6.3 Future Work

The great response from the public and the new ideas generated through the meeting among AntiPhishing Group really pushes me a lot. The SiteWatcher Server itself is not only a product to serve SiteWatcher Client but also a developer-friendly framework for development. However, there are numerous improvements available for SiteWatcher Server.

6.3.1 More Powerful Services to SiteWatcher Client

Both Auto-Update and Auto-Check of the SiteWatcher Scheduler can be improved. As Auto-Update and Auto-Check are applied to Black Lists currently, the concepts of “searching” and “checking” Black URLs can be extended to URLs in all lists. Figure 6.1 and Figure 6.2 shows the design of Auto-Update and Auto-Check respectively.

![Figure 6.1 Extended Auto-Update design](image)
For Auto-Update, it is not only limited to get phishing URLs from PhishTank. Instead, Auto-Update can also get phishing URLs from other organizations that involve anti-phishing activities such as AWGP and Google.

For Auto-Check, not only Black URLs but also other URLs in the lists are required to check the connectivity in order to keep all URLs “alive”. However, we need to consider the checking performance. As more lists are checked, more time and resources are consumed in the Auto-Check. Therefore, we have to focus on the resource management.

Besides, since PhishingDefService Admin Framework has provided the base functions necessary to Auto-Update and Auto-Check, it is expected that the development would not take much time to implement the new design.

### 6.3.2 More Advanced Frameworks to Developer

Both PhishingDefService Client and Admin Frameworks have showed their flexibility and scalability. And they reduce the complexity of the SiteWatcher Server and Client. The pre-code solutions from the Client and Admin frameworks cover a large range of programming needs to SiteWatcher developers. And developers can combine the
functions in the frameworks with their own code to produce other advanced SiteWatcher applications.

Therefore it is necessary to maintain and keep the frameworks to be most updated. In the future, more functions will be included in the frameworks to provide more solutions to SiteWatcher developers. Besides, the SiteWatcher Scheduler, as a module, should be separated from the PhishingDefService so that PhishingDefService provides pure frameworks. Figure 6.3 shows the new design of PhishingDefService.
Figure 6.3 New design of PhishingDefService
6.4 Conclusion

Anti-phishing is an endless war against phishers. The AntiPhishing Group, SiteWatcher Client and Server are committed to protect Internet users from being phished before phishing attacks begin to impact computer networks. Development and improvement of the SiteWatcher Client by inducing SiteWatcher Server are proposed in this study. The result of the study is exciting; SiteWatcher Server is successful in supporting SiteWatcher Client to fight against phishing attacks.

In the future, the AntiPhishing Group will continue to work to provide innovative technology solutions that help combat phishing for benefit of Internet users.
7. Reference


[7]. Kawamoto, D. (2005). Faced with a rise in so-called pharming and crimeware attacks, the Anti-Phishing Working Group will expand its charter to include thee emerging threats. ZDNet India. August 4, 2005.

[8]. AntiPhishing Group @ Dept. of CS, City University of Hong Kong, http://antiphishing.cs.cityu.edu.hk/


   http://www.seclab.tuwien.ac.at/papers/compsac05_antiphish.pdf


8. Appendix
8.1 Monthly Log

**October 2006**
1) Project Plan
2) Debug warning and error in the SiteWatcher Client
3) Control the memory used
4) Scanning top level links of a website
5) Design of SiteWatcher Server database structure
6) Testing of the SiteWatcher Client

**November 2006**
1) Design of SiteWatcher Server
2) Implementing SiteWatcher Server, Suspected List is drafted
3) Implementing SiteWatcher Client (URLSniffer) and testing

**December 2006**
1) Design SiteWatcher Server
2) Design of PhishingDefServices

**January 2007**
1) Implementing SiteWatcher Server, new functions including:
   -a) Login System
   -b) Black List
   -c) White List
   -d) Protected List + Sensitive Word List
   -e) White Domain List
   -f) Customized Protected List + Customized Sensitive Word List
   -g) Threshold Training System
   -h) Improved UI
2) Implementing SiteWatcher Client (LocalDB)
3) Implementing PhishingDefServices, new functions including:
   -a) Report suspected URLs from SiteWatcher Client to Server
   -b) Download Protected List with Sensitive words to the SiteWatcher Client
   -c) Download White and Black List to the SiteWatcher Client
   -d) Download White Domain Name List to the SiteWatcher Client
   -e) Download Customized Protected URL with customized sensitive words
5) The SiteWatcher Server and PhishingDefService have been rolled to the production server
6) SiteWatcher Server and PhishingDefService maintenance
7) Design a new architecture for the communication between SiteWatcher Server, PhishingDefServices and SiteWatcher Client

**February 2007**
1) Implementing SiteWatcher Server, including:
   -a) Permanent Delete in Suspected, Protected, Sensitive Word, Black, White, White Domain Name, Customized Protected and Customized Sensitive Word List
   -b) Added CSS to the interface
   -c) Improved Javascript
   -d) Optimizing the code such as batch update and delete
   -e) A page for updating the user's password

2) The new version of the SiteWatcher Server and PhishingDefService based on the new architecture discussed
   -a) Most of the code in the business layer is moved to the PhishingDefService web services. It is better for maintenance as the developers only focus on the web services when editing the business layer. However, the tradeoff is the performance is slower than the previous architecture

3) PhishingDefService
   -a) Merge the code with others
   -b) Become the main framework for the SiteWatcher Server Side

4) SiteWatcher Client Help File
   -a) The help file has been completed. All the contents are updated to meet the current SiteWatcher version. The help file has been rolled out to the production server

5) SiteWatcher Monitor
   -a) It is a planned module or system. It is used to do scheduled tasks.
   -b) Permanent delete the records which is in "Delete" status or useless data after a certain time.
   -c) Auto update the Black List from other anti-phishing web sites
   -d) Reporting to administrators
March 2007

1) A scheduling timer is added into the PhishingDefService to perform:
   - a) Auto-Delete "Deleted" or useless records in the SiteWatcher Server database
   - b) Auto-Update Black List by downloading new Black URLs from http://data.phishtank.com/data/online-valid/index.xml. It is assumed that the URLs provided in the website above are unique and real Black URLs. The Auto-Update task is a time-consuming task (about 10 minutes for 10000 records)
   - c) Auto-Check Black List to check the existence of the URLs in the Black List. As the Black URLs’ life time is usually short, it is required to have a scheduled checking task to check the status code of the Black URLs

2) A Log will be dropped down in the Server and an email will be sent to the administrator’s email to report the status of each scheduled task

3) A Configuration Page is added to set the web service and the scheduling timer. Since we offer an user interface to edit the config file, user authentication is required in order to access this page and edit the setting. Once the config file is updated, the web service will be stopped and restarted.