Virtual Universal Resource Locator(VURL)

Yao-Ming Yeh, Jill Chen, Arr-Mien Chou
Department of Information and Computer Education
National Taiwan Normal University
ymyeh@ice.ntnu.edu.tw , jillchen@ice.ntnu.edu.tw , amchou@ice.ntnu.edu.tw

Abstract

On the World Wide Web zone, users frequently receive "File Not Found" messages[1]. We propose a new method, which we call it VURL (Virtual Universal Resource Locator), to utilize the well-known URL system to remedy this problem. An URL usually specifies the exact static path where the Web page is located; on the contrary, a VURL. specifies the virtual path of a Web page, the location of which may be changed from time to time.

To the public, VURL is a permanent identify to locate Web page, no matter how the URLs of the Web pages are changed. Using VURL, the Web hyperlink reference in the Web space can be easily maintained consistent way.

Introduction

On Internet, when users use Browser to look for data, Browser usually responds by the message "File Not Found."[1]. This is the so-called "dangling link." There may be two causes of these dangling links. First, the Web pages in fact do not exist, called "file-nonexistent dangling link". Second, the Web pages do exist but cannot be found due to incorrect URLs, called "file-existent dangling link".

The "dangling link" problem and booming total number of Web pages have put the Web administrators in a difficult situation: they are forced either to make sure that no Web page is moved, or to do some maintenance work whenever a Web page is moved. Often, the administrator will send "address change" messages to all his registered customers. He would either ask for help from a third-party, such as the NETMIND [8] or to leave a short notice at the old URL. All three methods require users who bookmark the old, obsolete URL to do corresponding maintenance too. Unfortunately, none of these methods can successfully inform either user or owner of the changes that happen to those Web pages, which contain the old, obsolete URLs as its embedded linkages.

In this paper, we propose the idea of VURL (Virtual Universal Resource Locator) to cope with the "dangling links" problem. An URL usually specifies the exact static path where the web page is located; on the contrary, an VURL specifies the virtual path of a Web page, the location of which may be changed from time to time. Remember that while Web users always have to face the possible change of a URL, a VURL is valid forever unless the Web page is deleted.

In our implementation, every Web page in the VURL system has a pair of VURL and URL and these data are recorded in a table. Once some Web page is moved to a new location, the URL in the table will be modified according to its new location, but VURL remains unchanged. In fact, the VURL of a Web page will never be changed once it is created. In this paper, we also provide Web administrator a manageable tool to maintain the consistency of the VURL.

Related Work

"Dangling links" problem is a problem that all users of the World Wide Web have to face and there are many approaches aimed to solve this problem. The first solution to the problem, more formally known as "referential integrity," is *html_analyzer* [2] that attempts to assist the maintenance of HyperText MarkUp Language (HTML) database. The next approach extends local hyperlinks of referential integrity to a distributed environment represented by *Mom Spider* [3] and EIT's link analyzer software [4]. These approaches are helpful in identifying and removing broken links.

There are some disadvantages of these approaches. One is that these hyperlinks are not modified immediately after they are changed, and they are run by programs only nightly, or weekly, because some level of local hyperlink consistency has to be maintained.

Two other approaches emphasize on the bi-directional nature of hyperlinks-based Web[1] [7] and on adding a complementary hyperlink service [5] [6]. These solutions have several shortcomings: one is that it is required to implement a complicated new protocol or to modify the HTTP protocol in their architectures. Moreover, none of the above solutions provide a manageable tool the for Web administrator who then needs to read E-mails which contain the modified or removed Web pages information sent by the associated Web servers and update the information by himself. The latter solution is related to a kind of dilemma: whether to inform the user that the Web pages are deleted or moved or to find a complementary hyperlink for the users. In addition, these two approaches extract all hyperlinks from all HTML files and need at least two databases: Link Database and MetaInfo Database.

To summarize, existent attempts to do link maintenance are still limited by the traditional thinking of the exact path where a Web page is located and which a URL is supposed to represent. In principle, in this paper our key idea is to substitute the dynamic VURL for the traditional URL.

The Architecture of VURL

There are three parts which constitute the design of the VURL architecture. They are as follows:

1. VAMT(Virtual Address Mapping Table): provide a mapping table between the logical and physical name of Web page.

2. The execution processes of PVURL, that is CGI program for Address Mapping: provide search mechanism for VAMT.

3. VAMM (Virtual Address Mapping Manager): provide consistency maintenance for VAMT.

We call a Web server which adopts the VURL architecture a **VURL Server.** Every Web page in a VURL Server is also called a **VURL page** and each VURL page has two addresses. One is traditional URL, the real address, which we call the URL_P (Physical URL); the other is the permanent address, provided for Web users by the VURL system, which we call the URL_V (Permanent Virtual URL). Figure 1 depicts the simple architecture of the VURL system. The mechanism of the VURL architecture is almost the same as that of the existent Web server except that a CGI program, which is named PVURL, will be triggered first for every connection request from a Web user using a URL_V. The PVURL CGI will find out the exact recent Web page's location by checking the VAMT (Virtual Address Mapping Table) (detail see Figure 3). The URL_V is a logical name of a Web page, whereas the URL_P is a physical name of this Web page. This concept is similar to that of the Domain Name(the logical name) and IP address(the physical name) of a TCP/IP server. In the VURL system, each VURL server maintains its own local VAMT to provide the mapping between logical name and physical name. While for TCP/IP Domain Name table of many servers is kept and maintained at some particular servers called Domain Name Server.

The process of PVURL CGI program

A URL_V follows the URL_P format, but with special structure. In Figure 1, the URL_V uses the word "PVURL.exe" to trigger the PVURL CGI, and the whole URL_V specifies an unique record in VAMT. Figure 2 depicts the execution processes of PVURL CGI when it is triggered. The PVURL CGI first receives the parameters from Web server and gets the content of Path_Translated environment variable from the URL_V. PVURL CGI will check whether the URL_V does exist in VAMT (Figure 3) . If the URL_V exist, PVURL CGI will get the corresponding mapped URL_P; then, PVURL CGI will go on checking whether the path of the URL_P is valid. If the URL_P is valid, i.e., if the Web page exists in the valid Web root directory, PVURL CGI will send out the requested Web page. Otherwise, PVURL CGI will send a proper error message to the user.

In Figure 3, the VAMT contains two parts: URL_V and URL_P. The URL_V field is the permanent virtual URL which provides for the public Web users; the URL_P field is the exact URL which really presents the Web page location. In the VURL system, the mapping of these two fields is one to one, for example, if a user requests the URL_V of 140.138.32.68/PVURL.exe/cgidemo.htm, the PVURL CGI will find out the URL_P which is the correct location of the Web page – 140.138.32.68/cgidemo.htm. And PVURL CGI will send this Web page (cgidemo.htm) to the user.

VAMM (Virtual Address Mapping Manager)

As far as the Web administrator is concerned, our architecture provides a feasible manageable environment. In our design, all the management tasks can be accomplished by VAMM which will preserve the consistency of the VAMT.

There are several kinds of operators involved in VAMM which shown in Figure 4:

REFRESH: This function should be executed when a traditional Web Server is open to the public for providing VURL service, or when it offers the index file to a searching engine. This function will provide necessary works needed to modify all the Web pages in the Web Server into the VURL page. That is to say, it will change the local URL_P (the link that is connected to the other pages in its own Web Server) of every Web page into the URL_V; and it will change the URL_P of its Web page into the URL_V, which will be written into the VAMT for the mapping of the URL_V and URL_P. The following is an example of modifying the local URL_P of a Web page into the URL_V. Suppose that grumpy.ice.ntnu.edu.tw is a VURL Sever.

For example:

original HTML

<A HREF="http://grumpy.ice.ntnu.edu.tw/jillchen/news/cloth.htm"> HTML Notes </A>

modified HTML

<A HREF="http://grumpy.ice.ntnu.edu.tw/PVURL.exe/cloth.htm"> HTML Notes </A>

ADD: The VURL administrator has to execute this function when he wants to add some new Web pages to the VURL Server. This program will take two steps aiming at those newly-added Web pages. First, it modifies the local URL_P of the newly-added Web pages into the URL_V, and writes the local URL_P to the VAMT along with the URL_V. Second, it modifies the URL_P of the newly-added Web pages into the URL_V, and writes to the VAMT both the URL_P and URL_V.

This function is very similar to REFRESH. The difference is that REFRESH modifies all the Web pages in the Web Server into the VURL page, and ADD only does this to the newly-added Web pages.

MODIFY: The VURL Server administrator has to execute this function when the content of an existent VURL page is changed, because there might be newly-added local URL_P in the VURL page or the previous URL_V in the Web page might be changed. To facilitate the management for the administrator, this program deals with both the newly-added local URL_P in the VURL page and the modification of the previous URL_V in the Web page. No matter which of the above mentioned action the administrator takes, this program will check, modify, and maintain the links in the VURL page and the mapping of the local URL_P and URL_V in the VAMT.

MOVE: The VURL Server administrator has to execute this function when the position of an existent VURL page is changed. The corresponding URL_P field in the VAMT needs to be corrected, because the URL_P of the VURL page is already changed.

DELETE: The VURL Server administrator has to execute this function to delete all the information about the corresponding URL_V in this page and keep correct the information of the VAMT when an existent VURL page is deleted.

Eternal URL

Our VURL system introduces a new and important concept called "Eternal URL". That is, Web pages in the cyberspace can have a permanent identity for every user when they use browser to browse the Web pages. Even when the system administrator moves the Web pages from one directory to another or from one server to another, our VURL system can still keep its permanent identity valid by maintaining its corresponding entry in VAMT. Therefore, after a Web page is created, a VURL can be assigned to it, and one entry according to this VURL (URLv) and the current path of the Web page (URLp) in VAMT is also created when this Web page is published to the public. When a client browses the Web page, he can bookmark this VURL as the permanent identity of this Web page. As long as the Web page is not deleted, the client can always use the bookmark to locate the Web page no matter how it is moved from one location to another.

The concept of VURL can solve the problem of "file-existent dangling links". Save the owner of Web pages and the system administrator a lot of works when they update the content of Web pages and reconfigure the disk space of the Web server. They don't have to bother the trouble to notify the old users of the movement of the Web pages.

As for the problem of “file-nonexistent dangling links”. Since when the Web page is deleted, the corresponding entry in VAMT is also deleted in our VURL system. Therefore, the client will receive “File not Found” message, which means “file was deleted” in our VURL system. For this problem, some researchers proposed that a complementary hyperlink is suggested in their system. However, the search of complementary link for a dangling link is a heuristic work. In most of the cases, the suggested complementary hyperlink is not appreciated by the user. We believe that when a Web page is deleted, it is the Web server’s responsibility to tell the user that it is really deleted, and when a Web page is moved to other place, the maintenance works should be transparent to the user. Our VURL system can satisfy the above criteria.

Conclusions

A new concept “VURL” is proposed to solve the dangling links problem in World Wide Web. Using VURL, the idea of Eternal URL for a Web page can be provided. Our VURL system implements the VURL concept by a CGI program and the maintenance of VAMT, which is simple and straightforward. Our solution can release many maintenance works for the publisher of Web pages and the administrator of Web server.

Reference

[1] James E. Pitkow, R. Kipp Jones, “Supporting the Web: A Distributed Hyperlink Database System”, Fifth International WWW Conference, April 1996, <URL:http://www5conf.inria.fr/fich_html/papers/P10/Overview.html>

[2] Pitkow, J., “The Html Analyzer Documentation”, 1993, <URL:http://www.cc.gatech.edu/ftp/people/pitkow/html_analyzer/README.html>

[3] Fielding, R., “What is MOMspider? “, 1994, <URL:http://www.gu.edu.au/gwis/cwis/momspider/momhelp.html>

[4] McGuire, J., “The EIT Link Verifier Robot”, 1994, <URL:http://info.webcrawler.com/mak/projects/robots/active/html/eit.html>.

[5] Leslie Carr, David De Roure, Wendy Hall, and Gary Hill, “The Distributed Link Service: A Tool for Publishers, Authors and Readers”, Third International WWW Conference, April 1994, <URL:http://www.w3.org/pub/Conferences/WWW4/Papers/178>

[6] Hong-Sharng Chiu, “Bidirectional Integrated Web Page System”, Master Thesis, Department of Information and Computer Education, National Taiwan Normal University, July 1997

[7] Bidirectional links. <URL:http://www.win.tue.nl/2L670/dynamic/bidirectional.html>

[8] NETMIND.
<URL :http://www.netmind.com/>

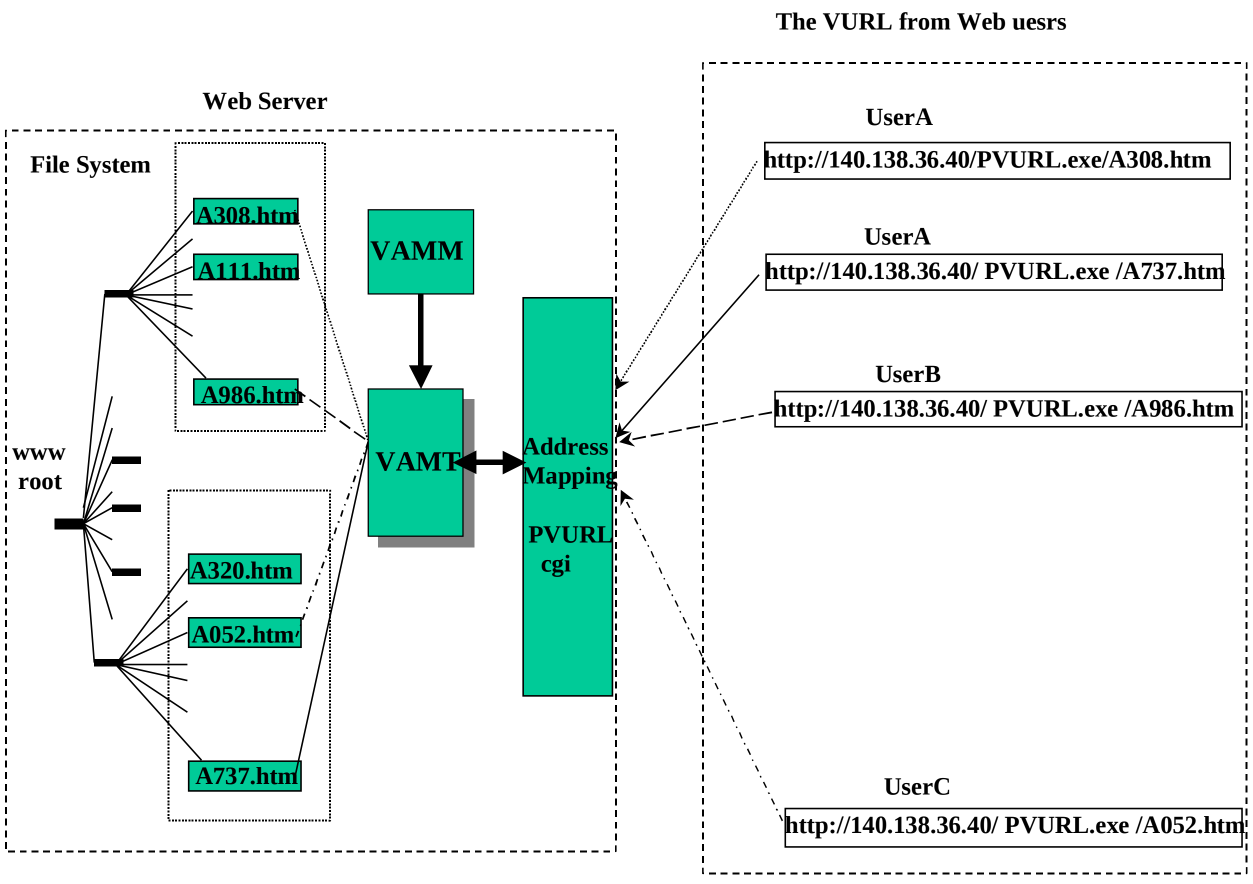
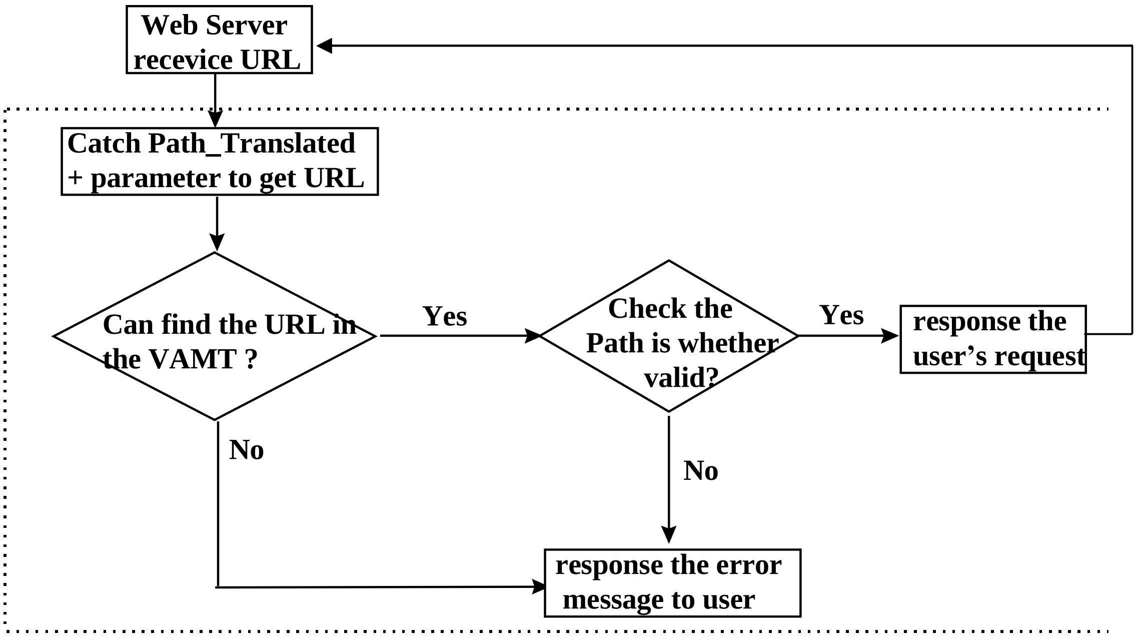
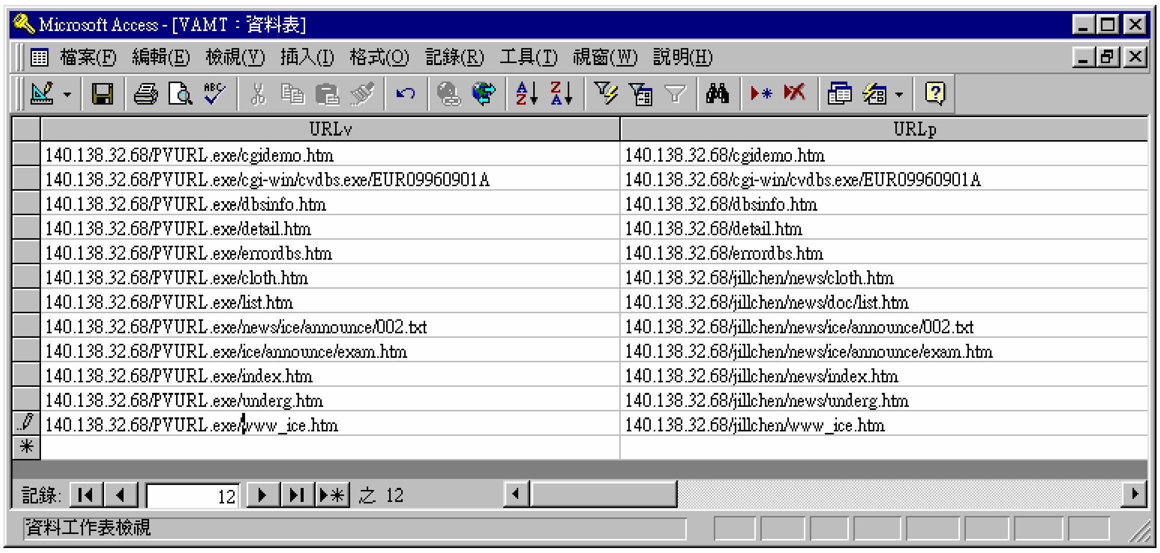

The VURL from Web uesrs
Web Server
File System
A308.htm
A111.htm
A986.htm
www
root
A320.htm
A052.htm
A737.htm
VAMM
VAMT
Address Mapping
PVURL cgi
UserA
http://140.138.36.40/PVURL.exe/A308.htm
UserA
http://140.138.36.40/ PVURL.exe /A737.htm
UserB
http://140.138.36.40/ PVURL.exe /A986.htm
UserC
http://140.138.36.40/ PVURL.exe /A052.htm

Figure 1 General VURL architecture



Web Server recevice URL
Catch Path_Translated + parameter to get URL
Can find the URL in the VAMT ?
Yes
Check the Path is whether valid?
Yes
response the user's request
No
No
response the error message to user

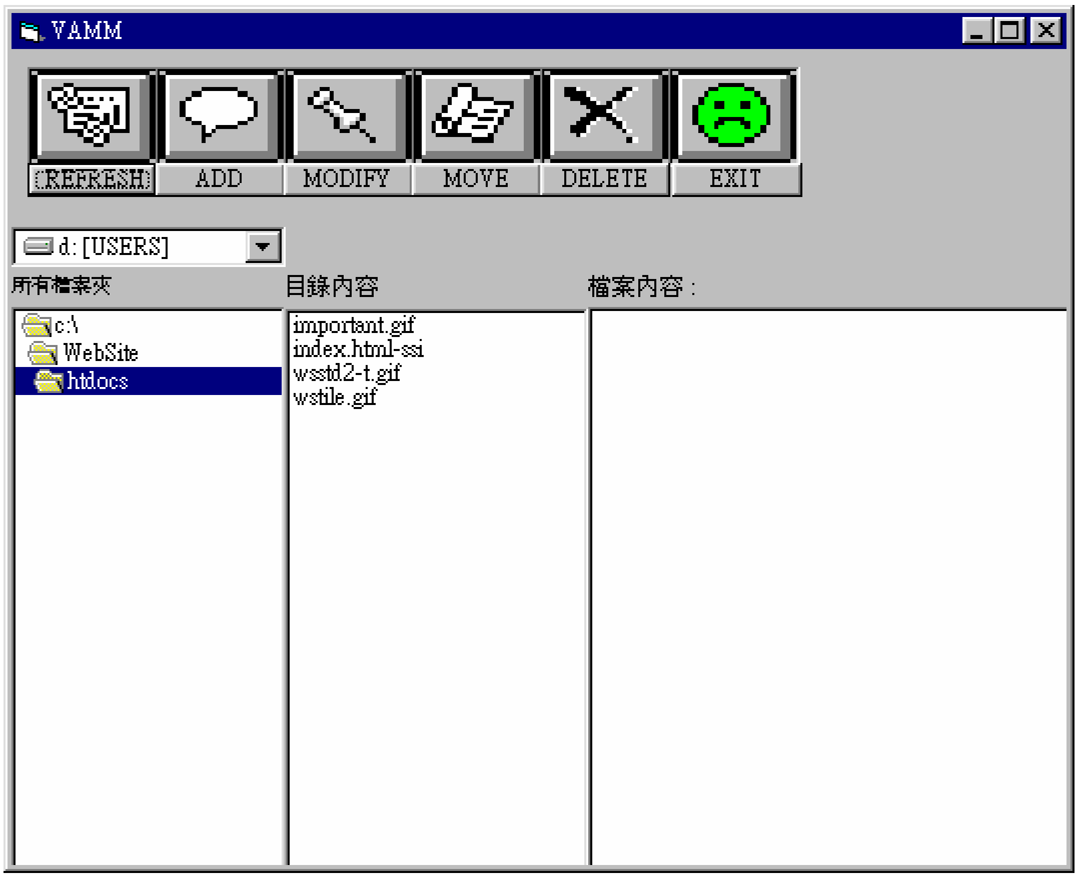
Figure 2 The process of PVURL CGI program



Microsoft Access - [VAMT：資料表]
檔案(F) 編輯(E) 檢視(V) 插入(I) 格式(O) 記錄(R) 工具(T) 視窗(W) 說明(H)
記錄: 12 之 12
資料工作表檢視

URLv	URLp
140.138.32.68/PVURL.exe/cgidemo.htm	140.138.32.68/cgidemo.htm
140.138.32.68/PVURL.exe/cgi-win/cvdbs.exe/EUR09960901A	140.138.32.68/cgi-win/cvdbs.exe/EUR09960901A
140.138.32.68/PVURL.exe/dbsinfo.htm	140.138.32.68/dbsinfo.htm
140.138.32.68/PVURL.exe/detail.htm	140.138.32.68/detail.htm
140.138.32.68/PVURL.exe/errordbs.htm	140.138.32.68/errordbs.htm
140.138.32.68/PVURL.exe/cloth.htm	140.138.32.68/jillchen/news/cloth.htm
140.138.32.68/PVURL.exe/list.htm	140.138.32.68/jillchen/news/doc/list.htm
140.138.32.68/PVURL.exe/news/ice/announce/002.txt	140.138.32.68/jillchen/news/ice/announce/002.txt
140.138.32.68/PVURL.exe/ice/announce/exam.htm	140.138.32.68/jillchen/news/ice/announce/exam.htm
140.138.32.68/PVURL.exe/index.htm	140.138.32.68/jillchen/news/index.htm
140.138.32.68/PVURL.exe/underg.htm	140.138.32.68/jillchen/news/underg.htm
140.138.32.68/PVURL.exe/www_ice.htm	140.138.32.68/jillchen/www_ice.htm

Figure 3 The Content of the VAMT



important.gif
index.html-ssi
wsstd2-t.gif
wstile.gif

Figure 4 The Interface of VAMM