Digital Evidence Seizure in Network Intrusions against Cyber-crime on Internet Systems

Shiuh-Jeng Wang[1,*] Yao-Han Chang[2] Hung-Jui Ke[3] Wen-Shenq Juang[1,4]

[1,2]Department of Information Management
Central Police University
Taoyuan 330, Taiwan

[3]Information Office at Changhua Police Bureau
Changhua County, Taiwan

[4]Department of Information Management
Shih Hsin University
Taipei, 116 Taiwan

Email:sjwang@mail.cpu.edu.tw

Received 16 November 2007; Revised 30 November 2007; Accepted 9 December 2007

Abstract. With the global trend of internet, many companies set up websites for international recognition and marketing. However, the hackers and potential attackers lurk on internet. More and more web attack methods have invented and threat these vulnerable websites. Recently, the "Code Injection" has become the major problem, such as SQL Injection, ASP Injection, PHP Injection and XSS (Cross Site Scripting) attack. The victims include the biggest Blog "Wrench" in Taiwan and the largest friend's community website "MySpace" in the world. In this paper, we will analyze the XSS attack and propose a scheme to collect evidence on network systems after XSS attack. We also propose our management strategy against XSS attack.

Keywords: Forensics and Evidence, internet attacks, website systems, XSS(Cross Site Scripting)

1 Introduction

The rapid development in computer and Internet technology has brought forth tremendous influence in our society. It is more and more necessary for many companies to expand their business by websites. The modern websites also have become more interesting while the consumers can interact with them. Along with the WEB2.0 evolution, the web designers focus on the interaction, participation and sharing. But we always have to leave our personal information on websites for customized service. The important personal information and weak security attract the hackers and the attackers. In recent years, the "Code Injection" has become the most serious problem. According to news report in 2007, there are almost 1,000 websites which are injected malicious programs [1]. According to Google and recent statistics, there are 984 websites have injected malicious programs in Taiwan, including many famous commercial websites. Google also claimed that your computers may download malicious programs without notice while browsing these websites. It means the malicious code like Trojan horse, backdoor program, spy-ware or other virus can be installed on their machines when the users browse the articles and pictures on the internet. These malicious programs does not only cause computer crash but also steal the account, password, and other personal information. Although the websites you browse are legitimate, you still have to be careful about identity theft. It is not possible to gather effective evidence of the high-tech crime by traditional methods. We need a new investigation strategy to collect the digital evidence and catch the criminals. In this paper, we provide a scheme to investigate XSS attacks and present the digital evidence in court.

The organization of the paper is presented as follows. In Sec. 2, we address the basic concept of XSS attacks. Our scheme is then proposed in Sec. 3. Our discussions are tabled in Sec. 4. Sec. 5 is our conclusions.

[*] Correspondence author

2 Backgrounds

"Code Injection" exploits the weakness of web application to cause damage. One of the most common attacks is SQL Injection. The vulnerability is present when user input is either incorrectly filtered for string literal escape characters embedded in SQL statements or user input is not strongly typed and thereby unexpectedly executed. All browsers will interpret the overwhelming majority of instructions in a similar manner, however, discrepancies in the handling of esoteric code may result in a radically different outcome between internet Explorer and Firefox [2]. The standard web page contains text and HTML(Hyper Text Markup Language) tags. In order to enhance the interaction and function on the website, we can insert the JavaScript, VB Script into the web pages. Therefore, when a web page contains JavaScript functions, for example, the popular WEB2.0 BLOG, forum, and the web mail service may be attacked. When a user browses the website, the browser will interpret the HTML tags, such as <BODY>, <A HREF>, but it also executes the script language with <SCRIPT>, <APPLET>, <FORM>, <EMBED/OBJECT> tags. Sometimes the malicious code is triggered by the script language.

There are different kinds of "Code Injection". The most serious attack is the Cross-Site Scripting (XSS). XSS attack is ranked first on the OWASP Top 10 Web application vulnerabilities in 2007. The Open Web Application Security Project (OWASP) is an open community and non-profit organization which is dedicated to finding and fighting the causes of insecure software. The OWASP Top Ten shows the most serious problems of web application security [3]. Table 1 shows XSS attack in 2007 has become the most serious vulnerability of site applications, it is really critical to have our attention.

Table 1. OWASP Top 10 Web application vulnerabilities for 2007

A1 - Cross Site Scriptinging (XSS)	XSS flaws occur whenever an application takes user supplied data and sends it to a web browser without first validating or encoding that content. XSS allows attackers to execute script in the victim's browser which can hijack user sessions, deface web sites, possibly introduce worms, etc.
A2 - Injection Flaws	Injection flaws, particularly SQL injection, are common in web applications. Injection occurs when user-supplied data is sent to an interpreter as part of a command or query. The attacker's hostile data tricks the interpreter into executing unintended commands or changing data.
A3 - Malicious File Execution	Code vulnerable to remote file inclusion (RFI) allows attackers to include hostile code and data, resulting in devastating attacks, such as total server compromise. Malicious file execution attacks affect PHP, XML and any framework which accepts filenames or files from users.
A4 - Insecure Direct Object Reference	A direct object reference occurs when a developer exposes a reference to an internal implementation object, such as a file, directory, database record, or key, as a URL or form parameter. Attackers can manipulate those references to access other objects without authorization.
A5 - Cross Site Request Forgery (CSRF)	A CSRF attack forces a logged-on victim's browser to send a pre-authenticated request to a vulnerable web application, which then forces the victim's browser to perform a hostile action to the benefit of the attacker. CSRF can be as powerful as the web application that it attacks.
A6 - Information Leakage and Improper Error Handling	Applications can unintentionally leak information about their configuration, internal workings, or violate privacy through a variety of application problems. Attackers use this weakness to steal sensitive data, or conduct more serious attacks.
A7 - Broken Authentication and Session Management	Account credentials and session tokens are often not properly protected. Attackers compromise passwords, keys, or authentication tokens to assume other users' identities.
A8 - Insecure Cryptographic Storage	Web applications rarely use cryptographic functions properly to protect data and credentials. Attackers use weakly protected data to conduct identity theft and other crimes, such as credit card fraud.
A9 - Insecure Communications	Applications frequently fail to encrypt network traffic when it is necessary to protect sensitive communications.
A10 - Failure to Restrict URL Access	Frequently, an application only protects sensitive functionality by preventing the display of links or URLs to unauthorized users. Attackers can use this weakness to access and perform unauthorized operations by accessing those URLs directly.

Cross Site Scriptinging first initials should be CSS. But CSS had been used by Cascading Style Sheets. It is used to a structured adding style (such as fonts, spacing and color) of computer language. For example, CSS is applied to HTML or XML applications. Therefore, Cross-site scripting changes the acronym to XSS. XSS attack is usually aimed at the web server applications, such as the provision of certain website allows users to fill in the form. An attack can exploit website application weaknesses to hack in the others computers by XSS. XSS attack can evade the security restrictions of browser. For example, you might only allow the script in the internal network, but XSS attack can be triggered in the third party. Moreover, XSS attack can be started without user's operation while surfing on websites. Exploiting the weakness, attackers can not only steal the data on the internet but also execute windows bomb attack or DDoS attacks. In addition, attackers can hack in web mail service to send virus mail and spam by XSS.

XSS attack even has classified as a virus. It can run on different platform and avoid firewall rules [4]. The traditional virus can access the file system and duplicate itself without network. The XSS attack has to rely on network and websites, and it has to be run on a client-server architecture.

3 Our scheme

Our scheme can be depicted in three parts as follow.

3.1 Invasion of XSS

After discussing the basic concept of XSS attacks, we will know about the attacks with three conditions. The three conditions include the attacker, the victim and a vulnerable website. We will analyze the four attack procedures and relationship among them. Our effort can help law enforcement to seize digital evidence under investigation.

The three conditions of XSS

It is necessary to meet the following three conditions for accomplishing XSS attack [5], as shown in Fig. 1.

Condition I: Vulnerable website applications.
There are many vulnerable website applications on the internet, such as the message board, forum, and guestbook. As long as the users can edit HTML contents on websites, it has the potential threat. There is a simple test. You can just input some texts between *<BODY></BODY>* label. For example：
<SCRIPT> alert ("Hello World")</SCRIPT>
The next step is to browse this page. If it pops up "Hello World" on your screen, it means this web application is not secure.
Condition II: One or more victims.
The innocent or gullible users would be the victims of the XSS attack.
Condition III: Malicious attackers.
The attackers have knowledge of XSS attack.

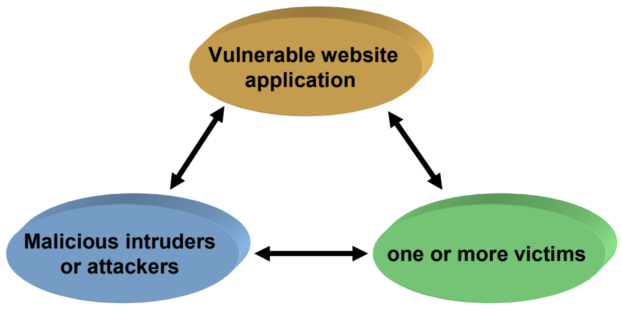

Vulnerable website application
Malicious intruders or attackers
one or more victims

Fig. 1. The three conditions of XSS

Four attack steps in XSS

The steps of XSS attack which we depict as follows:

Step 1: An attacker found a website application with XSS vulnerability.
Step 2: An attacker will input malicious code in message boards, guest books or hyperlinks on the websites.

Step 3: When the victims browse the web pages (Cookies will be established) with malicious code, the cookies and other private information will be stolen at the same time. This valuable information will be sent back to the attacker's hosts.
Step 4: After attacker got the victims' cookies or personal information, he could disguise the normal users on the internet.

The practice of XSS attack

There are four main steps to start the XSS attack, as shown in Fig. 2. Firstly, the malicious attackers use tools to search the vulnerable website applications. Secondly, the attackers will inject malicious code into the web pages. After taking advantage of the flaw of web applications, the attackers just need to sit back and wait for the victims. When the innocent users visit the websites with malicious code, their cookies and private information will be stolen. And to make matters worse, their computer systems will be planted the trojan horses. The cookies contain the information about authorized users. So, the attackers can analyze the cookies to have the authority of websites for illegal purpose.

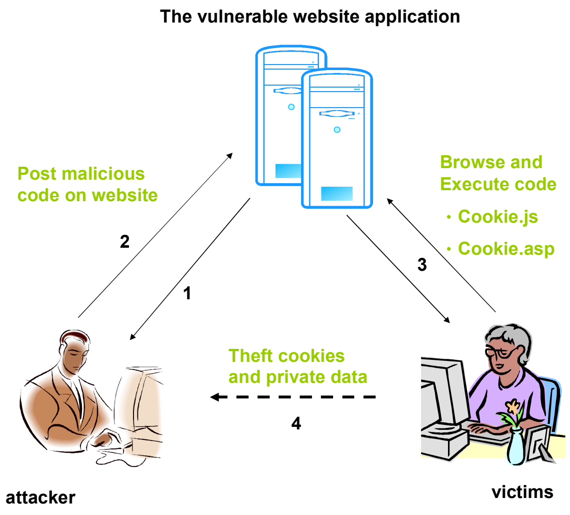

The vulnerable website application
Post malicious code on website
2
1
Browse and Execute code
· Cookie.js
· Cookie.asp
3
Theft cookies and private data
4
attacker
victims

Fig. 2. The invasion attack of XSS

3.2 Evidence Seizure in XSS

The problem of computer crime is the cyberspace. The internet fraud, system hacking, intimidation, on-line gambling and phishing are all big problems for law enforcement because of the difficulty of identify and investigation. The characteristics of XSS attack are invisible on web pages and the malicious code can be stored in another host. Therefore, it is more difficult for law enforcement to investigate and collect the digital evidence. So we propose the investigation procedure of XSS attack and evidence collection in crime scene. Then we will analyze and link the two-oriented clues to reconstruct the crime scene. Three phases are presented in our scheme shown as follows.

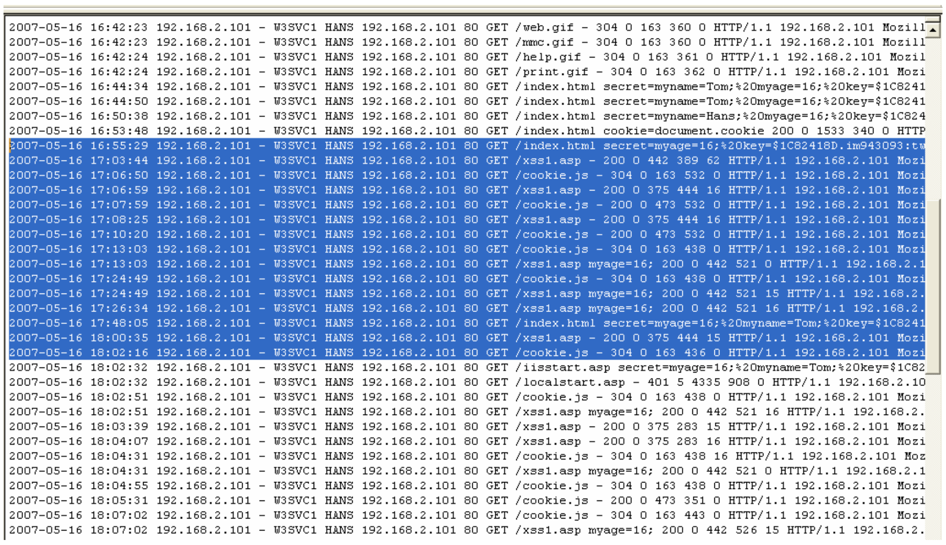
Phase I: the investigation of XSS

Firstly, we have to identify which page is injected the Trojan horse or virus. Then we analyze HTML labels to identify the malicious code and attacker's host. Secondly, we review the log files, such as the event log, IIS log, Apache log, FTP log, internet Explore Temporary internet Files and Cookies. These log files can help up to find out the source of attacker. Thirdly we follow up the malicious code to find the pattern which can be linked to the attacker. For example, we can use IP address to trace the location of attacker's host. Fourthly, we can seize the attacker hosts to find out the invasion time and the other victims. Lastly, we can link the connection of the attackers, victims and websites. Then we can present the report and evidence in court.

Phase II: seizing the digital evidence of XSS
We need to collect digital evidence from the victims, the victims' computer and the attacker's hosts. The evidence can prove the crime. The evidence we can find from:

1. Victim website logs: The websites usually have default setting to record the users' browsing records, such as IIS logs, Apache logs, FTP logs and the event logs. The log files contain connection time, IP address, user account, browsing pages and actions. Fig. 3 and Fig. 4 shows Web Server and FTP Server logs on the windows platform.

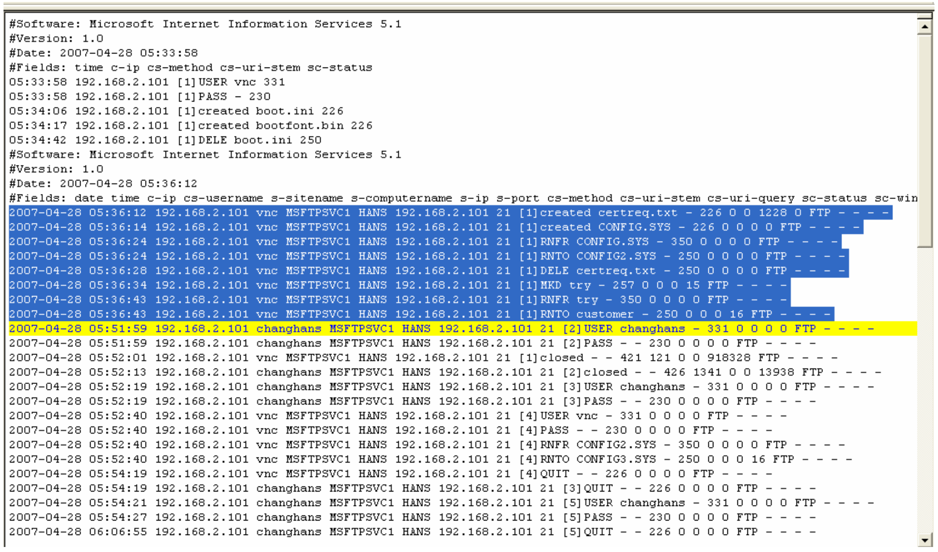
- Web Server log file：*C:\WINDOWS\system32\Logfiles\W3SVC1*



```
2007-05-16 16:42:23 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /web.gif - 304 0 163 360 0 HTTP/1.1 192.168.2.101 Mozill
2007-05-16 16:42:23 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /mmc.gif - 304 0 163 360 0 HTTP/1.1 192.168.2.101 Mozill
2007-05-16 16:42:24 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /help.gif - 304 0 163 361 0 HTTP/1.1 192.168.2.101 Mozil
2007-05-16 16:42:24 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /print.gif - 304 0 163 362 0 HTTP/1.1 192.168.2.101 Mozi
2007-05-16 16:44:34 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /index.html secret=myname=Tom;%20myage=16;%20key=$1C8241
2007-05-16 16:44:50 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /index.html secret=myname=Tom;%20myage=16;%20key=$1C8241
2007-05-16 16:50:38 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /index.html secret=myname=Hans;%20myage=16;%20key=$1C824
2007-05-16 16:53:48 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /index.html cookie=document.cookie 200 0 1533 340 0 HTTP
2007-05-16 16:55:29 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /index.html secret=myage=16;%20key=$1C82418D.im943093:tw
2007-05-16 17:03:44 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /xss1.asp - 200 0 442 389 62 HTTP/1.1 192.168.2.101 Mozi
2007-05-16 17:06:50 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /cookie.js - 304 0 163 532 0 HTTP/1.1 192.168.2.101 Mozi
2007-05-16 17:06:59 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /xss1.asp - 200 0 375 444 16 HTTP/1.1 192.168.2.101 Mozi
2007-05-16 17:07:59 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /cookie.js - 200 0 473 532 0 HTTP/1.1 192.168.2.101 Mozi
2007-05-16 17:08:25 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /xss1.asp - 200 0 375 444 16 HTTP/1.1 192.168.2.101 Mozi
2007-05-16 17:10:20 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /cookie.js - 200 0 473 532 0 HTTP/1.1 192.168.2.101 Mozi
2007-05-16 17:13:03 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /cookie.js - 304 0 163 438 0 HTTP/1.1 192.168.2.101 Mozi
2007-05-16 17:13:03 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /xss1.asp myage=16; 200 0 442 521 0 HTTP/1.1 192.168.2.1
2007-05-16 17:24:49 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /cookie.js - 304 0 163 438 0 HTTP/1.1 192.168.2.101 Mozi
2007-05-16 17:24:49 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /xss1.asp myage=16; 200 0 442 521 15 HTTP/1.1 192.168.2.
2007-05-16 17:26:34 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /xss1.asp myage=16; 200 0 442 521 16 HTTP/1.1 192.168.2.
2007-05-16 17:48:05 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /index.html secret=myage=16;%20myname=Tom;%20key=$1C8241
2007-05-16 18:00:35 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /xss1.asp - 200 0 375 444 15 HTTP/1.1 192.168.2.101 Mozi
2007-05-16 18:02:16 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /cookie.js - 304 0 163 436 0 HTTP/1.1 192.168.2.101 Mozi
2007-05-16 18:02:32 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /iisstart.asp secret=myage=16;%20myname=Tom;%20key=$1C82
2007-05-16 18:02:32 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /localstart.asp - 401 5 4335 908 0 HTTP/1.1 192.168.2.10
2007-05-16 18:02:51 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /cookie.js - 304 0 163 438 0 HTTP/1.1 192.168.2.101 Mozi
2007-05-16 18:02:51 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /xss1.asp myage=16; 200 0 442 521 16 HTTP/1.1 192.168.2.
2007-05-16 18:03:39 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /xss1.asp - 200 0 375 283 15 HTTP/1.1 192.168.2.101 Mozi
2007-05-16 18:04:07 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /xss1.asp - 200 0 375 283 16 HTTP/1.1 192.168.2.101 Mozi
2007-05-16 18:04:31 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /cookie.js - 304 0 163 438 16 HTTP/1.1 192.168.2.101 Moz
2007-05-16 18:04:31 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /xss1.asp myage=16; 200 0 442 521 0 HTTP/1.1 192.168.2.1
2007-05-16 18:04:55 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /cookie.js - 304 0 163 438 0 HTTP/1.1 192.168.2.101 Mozi
2007-05-16 18:05:31 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /cookie.js - 200 0 473 351 0 HTTP/1.1 192.168.2.101 Mozi
2007-05-16 18:07:02 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /cookie.js - 304 0 163 443 0 HTTP/1.1 192.168.2.101 Mozi
2007-05-16 18:07:02 192.168.2.101 - W3SVC1 HANS 192.168.2.101 80 GET /xss1.asp myage=16; 200 0 442 526 15 HTTP/1.1 192.168.2.
```

Fig. 3. Web Server log file on windows system

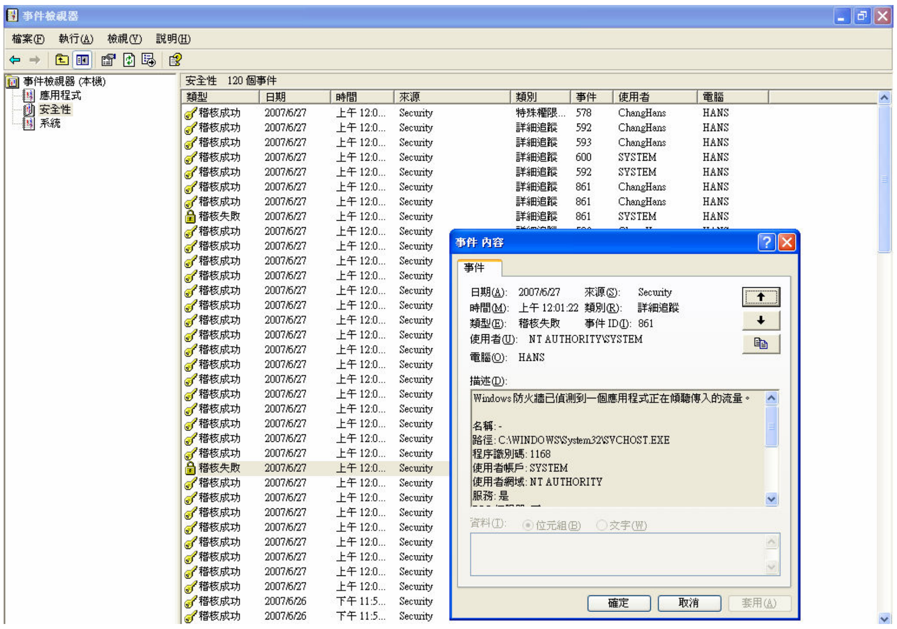
- FTP Server log file：*C:\WINDOWS\system32\Logfiles\MSFTPSVC1*



```
#Software: Microsoft Internet Information Services 5.1
#Version: 1.0
#Date: 2007-04-28 05:33:58
#Fields: time c-ip cs-method cs-uri-stem sc-status
05:33:58 192.168.2.101 [1]USER vnc 331
05:33:58 192.168.2.101 [1]PASS - 230
05:34:06 192.168.2.101 [1]created boot.ini 226
05:34:17 192.168.2.101 [1]created bootfont.bin 226
05:34:42 192.168.2.101 [1]DELE boot.ini 250
#Software: Microsoft Internet Information Services 5.1
#Version: 1.0
#Date: 2007-04-28 05:36:12
#Fields: date time c-ip cs-username s-sitename s-computername s-ip s-port cs-method cs-uri-stem cs-uri-query sc-status sc-win
2007-04-28 05:36:12 192.168.2.101 vnc MSFTPSVC1 HANS 192.168.2.101 21 [1]created certreq.txt - 226 0 0 1228 0 FTP - - - -
2007-04-28 05:36:14 192.168.2.101 vnc MSFTPSVC1 HANS 192.168.2.101 21 [1]created CONFIG.SYS - 226 0 0 0 0 FTP - - - -
2007-04-28 05:36:24 192.168.2.101 vnc MSFTPSVC1 HANS 192.168.2.101 21 [1]RNFR CONFIG.SYS - 350 0 0 0 0 FTP - - - -
2007-04-28 05:36:24 192.168.2.101 vnc MSFTPSVC1 HANS 192.168.2.101 21 [1]RNTO CONFIG2.SYS - 250 0 0 0 0 FTP - - - -
2007-04-28 05:36:28 192.168.2.101 vnc MSFTPSVC1 HANS 192.168.2.101 21 [1]DELE certreq.txt - 250 0 0 0 0 FTP - - - -
2007-04-28 05:36:34 192.168.2.101 vnc MSFTPSVC1 HANS 192.168.2.101 21 [1]MKD try - 257 0 0 0 15 FTP - - - -
2007-04-28 05:36:43 192.168.2.101 vnc MSFTPSVC1 HANS 192.168.2.101 21 [1]RNFR try - 350 0 0 0 0 FTP - - - -
2007-04-28 05:36:43 192.168.2.101 vnc MSFTPSVC1 HANS 192.168.2.101 21 [1]RNTO customer - 250 0 0 0 16 FTP - - - -
2007-04-28 05:51:59 192.168.2.101 changhans MSFTPSVC1 HANS 192.168.2.101 21 [2]USER changhans - 331 0 0 0 0 FTP - - - -
2007-04-28 05:51:59 192.168.2.101 changhans MSFTPSVC1 HANS 192.168.2.101 21 [2]PASS - - 230 0 0 0 0 FTP - - - -
2007-04-28 05:52:01 192.168.2.101 vnc MSFTPSVC1 HANS 192.168.2.101 21 [1]closed - - 421 121 0 0 918328 FTP - - - -
2007-04-28 05:52:13 192.168.2.101 changhans MSFTPSVC1 HANS 192.168.2.101 21 [2]closed - - 426 1341 0 0 13938 FTP - - - -
2007-04-28 05:52:19 192.168.2.101 changhans MSFTPSVC1 HANS 192.168.2.101 21 [3]USER changhans - 331 0 0 0 0 FTP - - - -
2007-04-28 05:52:19 192.168.2.101 changhans MSFTPSVC1 HANS 192.168.2.101 21 [3]PASS - - 230 0 0 0 0 FTP - - - -
2007-04-28 05:52:40 192.168.2.101 vnc MSFTPSVC1 HANS 192.168.2.101 21 [4]USER vnc - 331 0 0 0 0 FTP - - - -
2007-04-28 05:52:40 192.168.2.101 vnc MSFTPSVC1 HANS 192.168.2.101 21 [4]PASS - - 230 0 0 0 0 FTP - - - -
2007-04-28 05:52:40 192.168.2.101 vnc MSFTPSVC1 HANS 192.168.2.101 21 [4]RNFR CONFIG2.SYS - 350 0 0 0 0 FTP - - - -
2007-04-28 05:52:40 192.168.2.101 vnc MSFTPSVC1 HANS 192.168.2.101 21 [4]RNTO CONFIG3.SYS - 250 0 0 0 16 FTP - - - -
2007-04-28 05:54:19 192.168.2.101 vnc MSFTPSVC1 HANS 192.168.2.101 21 [4]QUIT - - 226 0 0 0 0 FTP - - - -
2007-04-28 05:54:19 192.168.2.101 changhans MSFTPSVC1 HANS 192.168.2.101 21 [3]QUIT - - 226 0 0 0 0 FTP - - - -
2007-04-28 05:54:21 192.168.2.101 changhans MSFTPSVC1 HANS 192.168.2.101 21 [5]USER changhans - 331 0 0 0 0 FTP - - - -
2007-04-28 05:54:27 192.168.2.101 changhans MSFTPSVC1 HANS 192.168.2.101 21 [5]PASS - - 230 0 0 0 0 FTP - - - -
2007-04-28 06:06:55 192.168.2.101 changhans MSFTPSVC1 HANS 192.168.2.101 21 [5]QUIT - - 226 0 0 0 0 FTP - - - -
```

Fig. 4. FTP Server log file on windows system

2. Victim host event logs: The event and audit logs are also the important information source. There are application logs, security logs and system logs which are shown in Fig. 5. In security logs, we can identify whether a malicious attacker has tried to invade the host from the account audit.



安全性 120 個事件

類型	日期	時間	來源	類別	事件	使用者	電腦
稽核成功	2007/6/27	上午 12:0...	Security	特殊權限...	578	ChangHans	HANS
稽核成功	2007/6/27	上午 12:0...	Security	詳細追蹤	592	ChangHans	HANS
稽核成功	2007/6/27	上午 12:0...	Security	詳細追蹤	593	ChangHans	HANS
稽核成功	2007/6/27	上午 12:0...	Security	詳細追蹤	600	SYSTEM	HANS
稽核成功	2007/6/27	上午 12:0...	Security	詳細追蹤	592	SYSTEM	HANS
稽核成功	2007/6/27	上午 12:0...	Security	詳細追蹤	861	ChangHans	HANS
稽核成功	2007/6/27	上午 12:0...	Security	詳細追蹤	861	ChangHans	HANS
稽核失敗	2007/6/27	上午 12:0...	Security	詳細追蹤	861	SYSTEM	HANS
稽核成功	2007/6/27	上午 12:0...	Security				
稽核成功	2007/6/27	上午 12:0...	Security				
稽核成功	2007/6/27	上午 12:0...	Security				
稽核成功	2007/6/27	上午 12:0...	Security				
稽核成功	2007/6/27	上午 12:0...	Security				
稽核成功	2007/6/27	上午 12:0...	Security				
稽核成功	2007/6/27	上午 12:0...	Security				
稽核成功	2007/6/27	上午 12:0...	Security				
稽核成功	2007/6/27	上午 12:0...	Security				
稽核成功	2007/6/27	上午 12:0...	Security				
稽核成功	2007/6/27	上午 12:0...	Security				
稽核成功	2007/6/27	上午 12:0...	Security				
稽核成功	2007/6/27	上午 12:0...	Security				
稽核成功	2007/6/27	上午 12:0...	Security				
稽核成功	2007/6/27	上午 12:0...	Security				
稽核成功	2007/6/27	上午 12:0...	Security				
稽核失敗	2007/6/27	上午 12:0...	Security				
稽核成功	2007/6/27	上午 12:0...	Security				
稽核成功	2007/6/27	上午 12:0...	Security				
稽核成功	2007/6/27	上午 12:0...	Security				
稽核成功	2007/6/27	上午 12:0...	Security				
稽核成功	2007/6/27	上午 12:0...	Security				
稽核成功	2007/6/27	上午 12:0...	Security				
稽核成功	2007/6/27	上午 12:0...	Security				
稽核成功	2007/6/27	上午 12:0...	Security				
稽核成功	2007/6/26	下午 11:5...	Security				
稽核成功	2007/6/26	下午 11:5...	Security				

日期(A): 2007/6/27 來源(S): Security
時間(M): 上午 12:01:22 類別(R): 詳細追蹤
類型(E): 稽核失敗 事件 ID(I): 861
使用者(U): NT AUTHORITY\SYSTEM
電腦(O): HANS
描述(D):

Windows 防火牆已偵測到一個應用程式正在傾聽傳入的流量。

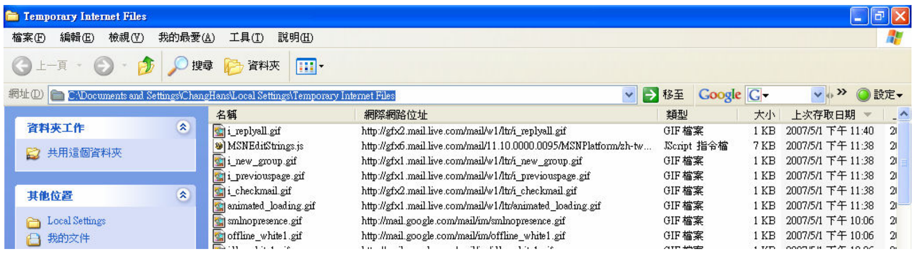
名稱: -
路徑: C:\WINDOWS\System32\SVCHOST.EXE
程序識別碼: 1168
使用者帳戶: SYSTEM
使用者網域: NT AUTHORITY
服務: 是

Fig. 5. Event logs on window system

3. Victims' browsing logs: We can know whether the victims had visited the malicious website from checking the victims' browser logs. The most common used browsers are Internet Explorer and Firefox, Their browsing logs are shown as Fig. 6 and Fig. 7.

- Internet Explore browsing logs:

C:\Documents and Settings\ChangHans\ Local Settings\Temporary Internet Files



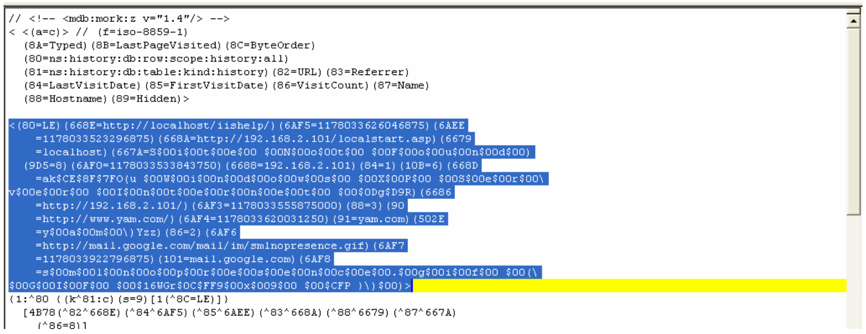
C:\Documents and Settings\ChangHans\Local Settings\Temporary Internet Files

名稱	網際網路位址	類型	大小	上次存取日期
i_replyall.gif	http://gfx2.mail.live.com/mail/w1/ltr/i_replyall.gif	GIF 檔案	1 KB	2007/5/1 下午 11:40
MSNEditStrings.js	http://gfx6.mail.live.com/mail/11.10.0000.0095/MSNPlatform/zh-tw...	JScript 指令檔	7 KB	2007/5/1 下午 11:38
i_new_group.gif	http://gfx1.mail.live.com/mail/w1/ltr/i_new_group.gif	GIF 檔案	1 KB	2007/5/1 下午 11:38
i_previouspage.gif	http://gfx1.mail.live.com/mail/w1/ltr/i_previouspage.gif	GIF 檔案	1 KB	2007/5/1 下午 11:38
i_checkmail.gif	http://gfx2.mail.live.com/mail/w1/ltr/i_checkmail.gif	GIF 檔案	1 KB	2007/5/1 下午 11:38
animated_loading.gif	http://gfx1.mail.live.com/mail/w1/ltr/animated_loading.gif	GIF 檔案	1 KB	2007/5/1 下午 11:38
smlnopresence.gif	http://mail.google.com/mail/im/smlnopresence.gif	GIF 檔案	1 KB	2007/5/1 下午 10:06
offline_white1.gif	http://mail.google.com/mail/im/offline_white1.gif	GIF 檔案	1 KB	2007/5/1 下午 10:06

Fig. 6. Internet Explore browsing logs

- Firefox browsing logs:

C:\Documents and Settings\ChangHans\ Application Data\Mozilla\Firefox\Profiles\0ll27z3r.default\



```
// <!-- <mdb:mork:z v="1.4"/> -->
< <(a=c)> // (f=iso-8859-1)
  (8A=Typed)(8B=LastPageVisited)(8C=ByteOrder)
  (80=ns:history:db:row:scope:history:all)
  (81=ns:history:db:table:kind:history)(82=URL)(83=Referrer)
  (84=LastVisitDate)(85=FirstVisitDate)(86=VisitCount)(87=Name)
  (88=Hostname)(89=Hidden)>

<(80=LE)(668E=http://localhost/iishelp/)(6AF5=1178033626046875)(6AEE
    =1178033523296875)(668A=http://192.168.2.101/localstart.asp)(6679
    =localhost)(667A=S$00i$00t$00e$00 $00N$00o$00t$00 $00F$00o$00u$00n$00d$00)
  (9D5=8)(6AF0=1178033533843750)(6688=192.168.2.101)(84=1)(10B=6)(668D
    =ak$CE$8F$7FO(u $00W$00i$00n$00d$00o$00w$00s$00 $00X$00P$00 $00S$00e$00r$00\
v$00e$00r$00 $00I$00n$00t$00e$00r$00n$00e$00t$00 $00$0Dg$D9R)(6686
    =http://192.168.2.101/)(6AF3=1178033555875000)(88=3)(90
    =http://www.yam.com/)(6AF4=1178033620031250)(91=yam.com)(502E
    =y$00a$00m$00\)Yzz)(86=2)(6AF6
    =http://mail.google.com/mail/im/smlnopresence.gif)(6AF7
    =1178033922796875)(101=mail.google.com)(6AF8
    =s$00m$00l$00n$00o$00p$00r$00e$00s$00e$00n$00c$00e$00.$00g$00i$00f$00 $00(\
$00G$00I$00F$00 $00$16WGr$0C$FF9$00x$009$00 $00$CFP )\)$00)>
{1:^80 {(k^81:c)(s=9)[1(^8C=LE)]}
  [4B78(^82^668E)(^84^6AF5)(^85^6AEE)(^83^668A)(^88^6679)(^87^667A)
    (^86=8)]
```

Fig. 7. Fire-Fox browsing logs

3.3 The Defense Strategy of XSS attack

To counter the XSS attack, we propose four key-strategies S1-S4.

S1. Filter the data form before they are sent to server

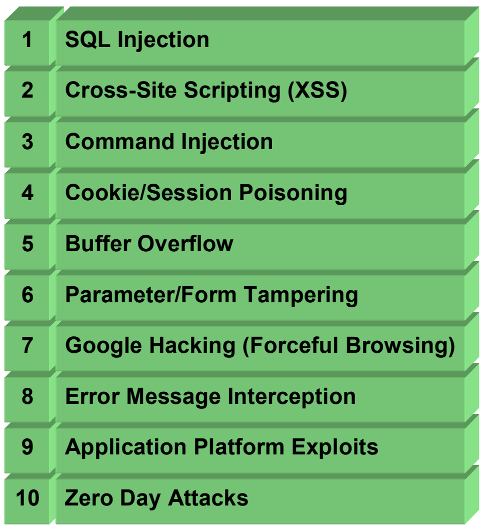
The main problem of website application is processing user input without filtering or checking. The attackers can input any character into form field, or show it on web pages. It causes that other users may browse the attacker's website without awareness. Therefore, an appropriate filter strategy is essential to prevent XSS attack.

For example, we only allow users input numbers in the telephone field, and input letters or in the name field. We can set the rules for users input as our first step. The second step is to deal with the specific characters and symbols. The specific symbols "<", ";" and so on, which can be part of program. Besides, the symbol ' < ' and '% 3c ' has the same meaning in HTML interpretation. Therefore, an attacker could use ' %3cSCRIPT% 3c ' to replace ' <SCRIPT> ' to avoid filter for attack. The third step is to check the user input recursively. The attacker may input ' <SCR <SCRIPT> IPT> ' to avoid the filter of '<SCRIPT>'. If we only filter the input once, the string ' <SCR <SCRIPT> IPT> ' would be ' <SCRIPT> ' after filtering. So, here are our two rules for filter the user input and output:

- Confirm the form which is filled with allowed characters.
- The user input should be filtered. If the users need to show specific characters or symbols on the web pages. The specific characters and symbols should be replaced with predefined HTML code.

S2. Application Firewall

Our strategy 2 is the application firewall. Many serious network security attacks target the vulnerabilities of the web applications. These attacks can avoid the traditional firewalls and intrusion-detection and prevention systems. The new generation of application firewall must provide a web-based applications protection mechanism. This mechanism can protect system from buffer overflow attack, SQL injection, and XSS intrusion, etc. These attacks can not be detected by general firewall or Intrusion Detection System or Intrusion Prevention System. Now, there is already a commercial product "Traffic Shield application firewall". This firewall system aims at web security under the corporate structure. It can resist Google hacking, XSS attack and other code injection by checking user session, input and response [6], as shown in Fig. 8.



1	SQL Injection
2	Cross-Site Scripting (XSS)
3	Command Injection
4	Cookie/Session Poisoning
5	Buffer Overflow
6	Parameter/Form Tampering
7	Google Hacking (Forceful Browsing)
8	Error Message Interception
9	Application Platform Exploits
10	Zero Day Attacks

Fig. 8. Traffic Shield application firewall system defense list

S3. Enhance website management security

Different privilege should use different IP address range. Strictly speaking, the system managers and general users should not share the same IP address range. Sometimes the system managers use remote control to manage website, it could be a security problem. If we really need to remote control, we should restrict the IP source. As long as the website is highly secured, the attackers have to make every effort to hack. Besides, highly secure website will have detail logs which can help us to find out the attackers.

S4. Adopt secure sockets layer (SSL) with HTTPS protocol

A Secure Sockets Layer (SSL) is a cryptographic protocol that provides secure communication on the internet. SSL was originally developed by Netscape and supports RC4, MD5 and RSA algorithm. It is widely used in the e-commerce, on-line payment and other confidential data transmission. Using SSL, the data will be encrypted before transmission. Therefore, the attackers can not be improve your e-business transaction security by installing SSL server certificate in the browser, because the transaction data between users and servers are all encrypted. Besides, SSL will check data integrity and establish session identification of each other. Avoiding hackers or third party to intercept the information and revise it. HTTPS is to strengthen the HTTP protocol, it can enhance website security.

4 Discussions

It is easy to destroy, alter digital evidence, but on the other hand, digital evidence is difficult to be collected. If we would like to acquire sound digital evidence in computer crime scene, a standard and forensic-based procedure is required [7]. In this section, we summarize the evidence acquisition and evidence examination as Table 2. The Table 2 shows the trace of XSS attacks in Sec. 3.1 with the three conditions. In Table 2, we can link the evidence among server logs, event logs, and browsing logs. In Table 2, we can link the evidence among server logs, event logs, and browsing logs. Our research proposed the evidence collection and analysis of the XSS attack as case study. The investigators can follow the trace shown in the Table 2 to hunt down the attackers. As a matter of fact, it is still a challenge to identify XSS attack on a single host. We have to follow the trace of victims, websites and attackers. Following our strategy, we can use the time series analysis to uncover the attackers. If we want to know when the victim browsed the website, we can compare user's browsing logs with website's logs. We also can investigate the attacker's host to find out the stolen personal information.

Table 2. Evidence collection from three conditions of XSS attacks

	Where evidence exists	**Benefits**
I. Vulnerable website applications	Server logs： a. Web Server log file C:\WINDOWS\system32\Logfiles \W3SVC1 b. FTP Server log file C:\WINDOWS\system32\Logfiles \MSFTPSVC1 Event logs： a. Application logs C:\WINDOWS\system32\config\AppEvent.Evt b. Security logs C:\WINDOWS\System32\config\SecEvent.Evt c. System logs C:\WINDOWS\system32\config\SysEvent.Evt Firewall logs, IDS/IPS logs, Antivirus logs	(1). Identify attackers and victims (2). Identify website vulnerability
II. One or more victims	Browsing logs： a. Internet Explore browsing logs C:\Documents and Settings\ChangHans\Local Settings\Temporary Internet Files b. Fire-Fox browsing logs C:\Documents and Settings\ChangHans\ApplicationData\Mozilla\Firefox\Profiles\0ll27z3r.default\	(1). Identify vulnerable websites.
III. Malicious intruders or attackers	Browsing logs： a. Internet Explore browsing logs C:\Documents and Settings\ChangHans\Local Settings\Temporary Internet Files b. Fire-Fox browsing logs C:\Documents and Settings\ChangHans\ Application Data\Mozilla\Firefox\Profiles\0ll27z3r.default\	(1). Identify victims and vulnerable websites. (2). Identify malicious behavior

5 Conclusions

In this paper, we have depicted how the XSS attack steals personal information by using HTTP and cross-platform. It seems just a minor problem, but it causes the largest community website MySpace (which is famous for interactive network of friends) to shutdown service for several hours. For now, many commercial website designers and innocent users are not aware of this serious problem. The law enforcement also has no idea to hunt down the attackers.

In this regard, we have proposed the skills to collect the digital evidence in XSS attack case. The most important is the computer logs . The experienced investigators can retrieve the logs from computer systems. Using time series analysis with Web logs, FTP logs, browsing temporary and event logs, we can reconstruct the crime scenes and collect the digital evidence. Besides, we proposed our strategy about filtering user input, enhancing security level with application firewall, using HTTPS protocol. Our strategy is useful for web administrators to defend XSS attack.

Acknowledgement:

This work was supported in part by the National Science Council of the Republic of China under the Grant NSC 95-2221-E-015-002-MY2, and by the iCAST project sponsored, National Science Council under the Grants NSC 95-3114-P-001-001-Y02, NSC 95-3114-P-001-002-Y02 and NSC 96-3114-P-001-002-Y.

References

[1] URL：http://udn.com/NEWS/INFOTECH/INF3/ 3876264.shtml

[2] D. Morgan, "Web Injection Attacks," *Network Security*, pp.8-10, 2006.

[3] URL：http://www.owasp.org/index.php/Top_10_2007

[4] Ashcroft, D. J. Daniels, and S. V. Hart, Forensic Examination of Digital Evidence: A Guide for Law Enforcement, U.S. Department of Justice, Office of Justice Programs, National Institute of Justice, April 2004.

[5] R. Braganza, "Cross-site Scripting- An Alternative View," *Network Security*, pp.17-20, September 2006.

[6] URL：http://www.f5.com/products/TrafficShield/

[7] W. Alcorn, "Cross-site Scripting Viruses and Worms- a New Attack Vector," *Network Security*, pp.7-8.J, July 2006.