An Automated User Transparent Approach to log Web URLs for Forensic Analysis

Muhammad Kamran Ahmed, Mukhtar Hussain, Asad Raza
Department of Information Security, College of Signals,
National University of Sciences and Technology, Pakistan
Kamran_ahmed79@yahoo.com, mukhtar.dr@gmail.com, asadraaza@gmail.com

Abstract

This paper presents an automated approach to record web activity as the user connects to Internet. It includes monitoring and logging of web URLs visited by the user. The distinctive features of this approach are a) it starts automatically, b) it is transparent to users, c) it is robust against intentional or un-intentional process kill, and d) it is robust against intentional or un-intentional corruption or deletion of log file. The first feature is achieved as the program/application will run with svchost.exe service which is initiated automatically. Transparency is achieved by storing the log file to a default hidden location defined by system variables as well as at a third location (logging server) on the network. Process killing is prevented through dependencies of this application on essential service required to connect to network and thus World Wide Web. The last feature determines that a log activity is also stored in logging server (not accessible to users) even if a user deletes or corrupts it from his local system. The log file contains important information of client, username, date and time of activity and URLs visited. The approach can give vital and potential evidential information of corporate web policy violations, employee monitoring, and law enforcement agencies (digital forensics investigators). This paper also carries out a comparative analysis of the performance and security of proposed scheme against some existing Web forensic and anti-forensic tools.

1. Introduction

Today, the society relies heavily on use of computers and Internet to accomplish everyday tasks, which includes almost everything from communicating, shopping online, banking and investing. With the wide-spread use of computers and Internet, comes a problem called computer crime, which includes fraud, identity theft, phishing and network infiltration to name a few. With computer crime on the rise, it is becoming extremely crucial for corporate to detect policy violators and digital forensic experts to examine and determine the web criminals.

Web browser forensic is extremely important in forensic analysis and incident response. Web applications that run on various platforms, record web activity in different formats. Windows records a number of user activities into log files. These log files can be deleted by authorized users as well as by those with malicious intentions by various means. Once these log files have been deleted, performing a forensic audit and monitoring user activity becomes difficult.

Most popular web browsers in use today are Internet Explorer (50%) and Mozilla Firefox (36%) [1]. During Web forensic analysis, it is necessary to parse history files in human readable format. Many different free and commercial forensic tools are available for this purpose.

978-0-7695-3807-5/09 $25.00 © 2009 IEEE
DOI 10.1109/IMF.2009.12

Internet Explorer records user web activities across multiple directories on host machines like temporary Internet files, cookie files and web browsing history file. Each of these directories contains a file called Index.dat. This is the file which is most populated and always retains the Internet Explorer (IE) history even if user deletes web cookies, temporary Internet files and web browsing history file. The Index.dat file records web activity data in a proprietary binary format which is known to Microsoft [3]. Table1 lists additional areas of the file system where other index.dat files may be located for IE running on Windows XP

Table 1. Location of index.dat files Windows XP

Operating System	File Path(s)
Windows 2K/XP	\Documents and Settings\<username>\Local Settings\Temporary Internet Files\Content.IE5\ \Documents and Settings\<username>\Cookies\ \Document and Settings\<username>\Local Settings\History\History.IE5\ C:\WINDOWS\system32\config\systemprofile\Local Settings\History\History.IE5

Detailed work on internal structure of Index.dat has been carried out by Keith J.Jones[4]. However, reconstructing records manually is tedious and time consuming. To automate this process, Keith J.Jones developed open source tools Pasco [5], Galleta [6] and Web Historian [7].

Many anti-forensic techniques and tools are also freely available to effectively destroy data on storage devices. Anti-forensic techniques include Steganography, Encryption and Wipe. Anti-forensic tools that clear web related activities are Secure Clean, Tracks Eraser Pro and Erase [2].

In this paper, we present simple proactive approach for Small and Medium Enterprise (SME) to capture, monitor and create a hidden log file of clients HTTP requests at local system as well as at remote server, without creating much overhead on underlying operating system (Windows XP). Thus, enabling system administrator to carry out the audit based on the log maintained by proposed application, even if user manages to clear the default log file created by various Web browsers. This proactive approach will help to detect organization Web policy violation. It provides better performance as:

- The program executes automatically with svchost.exe service which runs and provides network services.
- It stores the log file to a default hidden location defined by system variables as well as at a third location (logging server) on the network.
- It is robust against process killing, as it has dependencies on essential service required to connect to network and thus World Wide Web.
- The log file is also stored in logging server (not accessible to users) even if a user deletes or corrupts it from his system.

This paper is organized into four sections. Section 2 discusses the different web forensic and anti-forensic tools and techniques. Section 3 discusses the proposed scheme, its comparative analysis with tools discussed in section 2, performance analysis of proposed work and security analysis. Paper is concluded in section 4.

2. Related Work

Computer hard disk stores a wealth of information about user activities in log files, created by the underlying operating system and applications. Contents of log files can provide wealth of information to solve cases involving computers crimes. However, primary goal of an attacker is to remove its traces either by tampering or deleting log files to avoid detection. Once deleted or overwritten, retrieving such records will be challenging and time consuming. Sometimes, it is only possible to retrieve partial information.

Web activity investigation is one important aspect in computer forensic investigation. This leads to solving cases including company Web policy violation, online fraud, and identity theft etc. This section discusses some of the traditional Web Forensic and Anti-forensic tools existed.

2.1. Forensic Tools

Computer Forensic Tools (CFTs) allow investigators to recover deleted files, reconstruct activities, and gain intelligence about a computer's user. CFTs can be broadly classified into two categories: a) Persistent Tools: that analyzes data which is retained when computer is shutdown, and b) Volatile data analysis tools: which handles transitory data. Web forensic tools discussed in this paper are persistent data tools which analyze the hard drive to find log files created by Web browsers and parse the output in a format that can be presented in court of law. Different Web browsers create user activity files in their default directories, e.g., for IE, the default directory is "C:\Documents and Settings\ <username>\Local Settings\Temporary Internet Files\Contet.IE5\", for Mozilla Firefox and Netscape Navigator default directory path is "C:\ Documents and Settings\ <user name>\ Application Data\ Profiles\ <profile name>\ <random text>\ history. dat".

Traditional Web Forensic tools used to reconstruct the Web activity records of IE, Mozilla Firefox, and Netsacpe Nevigator are "Pasco" (an IE index.dat activity file parser), "Galleta" (an IE cookie viewer) and "Web Historian" (the directories analyzer) to find Web activity files created by IE, Mozilla Firefox, Safari and Opera.

2.2. Anti-Forensic Tools

Anti-forensic is a collection of tools and techniques aiming to avoid detection that some event has taken place, disrupting the collection of evidence hence causing doubts on a forensic report.

Windows default tools do not actually remove the trace. Like using simple delete command does not actually delete the file but merely changes one entry in disk's File Allocation Table (FAT) indicating that space is available. Similarly, format command does not remove sensitive files, but only overwrites the FAT.

Anti-forensic tools employ "disk wiping" (the process of overwriting sensitive data on hard disk) technique to securely remove the traces. These programs use Department of Defense standard (DOD 5220.22-M) to overwrite the entire media and files that were previously deleted but their traces were left on the hard disk.

Anti-forensic researchers have developed and distributed tools freely on Internet. Common Web anti-forensic tools like Tracks Eraser Pro [8] are designed to clear all unwanted Internet activity files from computer. It provides support for Microsoft

Windows 98/200/PX/Vista. Secure Clean [9] is a freely available disk wiping tool to securely clean and overwrite the slack space, swap files, free space, RAM, slack space and deleted e-mails. It can be configured to overwrite the file with alternating bit patterns as defined by DOD 5220.22-M.

3. Proposed Work

Unlike traditional forensic approach, a proactive forensic approach actively collects the data of evidential value in real time. Thus, investigators do not have to collect data from the history files but the data is actively collected and stored at a protected storage area, where it cannot be tampered. In this paper, we propose a simple and lightweight proactive forensic solution to capture and log the user out going HTTP request. The application is started automatically at system startup. The detailed working of the proposed solution is described as a flow chart shown in the figure 1.

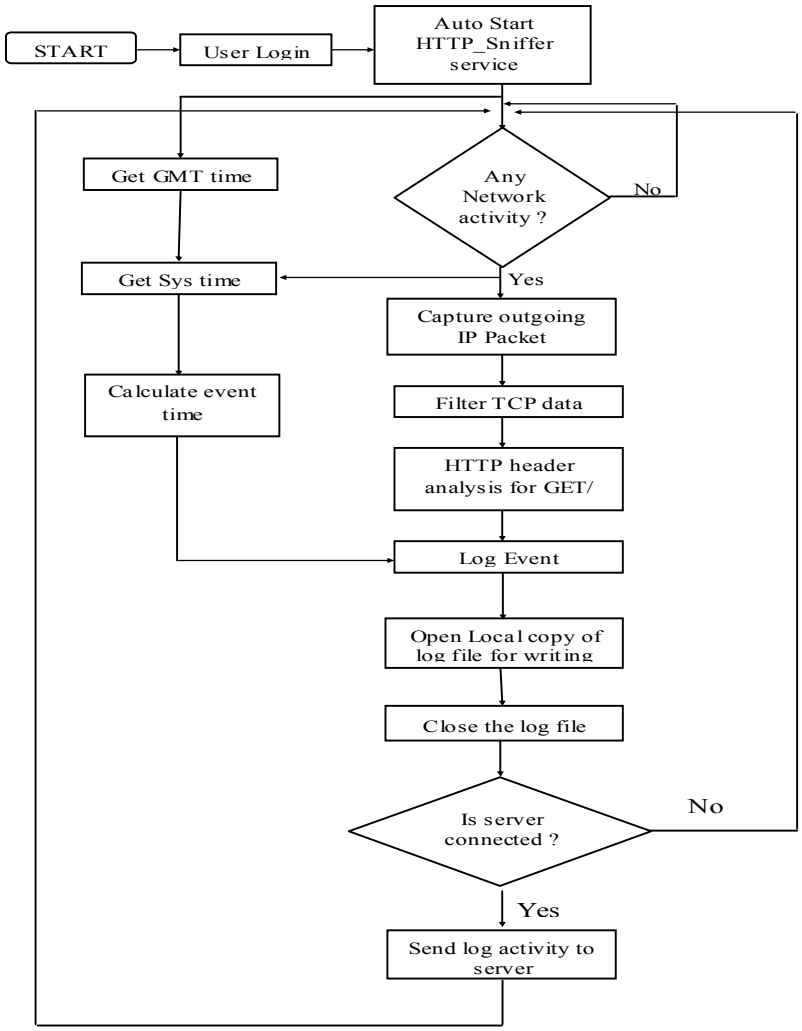

Figure 1. Proposed HTTP-Sniffer: Flow Diagram

As shown in figure 1 application start at system boot. It monitors client network activity by capturing any outbound IP packet. It gets GMT time from the remote server, to validate the current system date and time. Distinguish between TCP and UDP

packets. For TCP packets with data field >0, records the user requested URL in local log file and also sends the activity to local logging sever. Detailed description of flow diagram is given in implementation section.

Our scheme makes use of the fact that Web client uses HTTP requests to communicate with Web servers which follows client server model.

Client sends a HTTP request for getting access to some resource which can be a file or a Web page residing on remote server which is listening at port 80. Typical network packet carrying request is shown in figure 2.

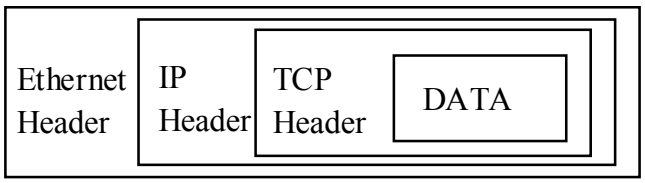

Figure 2. Network protocol layers in packets

HTTP header is sent as a payload in TCP packet and it contains information which can be used for digital evidence. The technique presented in this paper analyzes information carried in "HTTP Request header" only. A request message from a client to a server includes:

- A method to be applied to the resource, e.g. get method to retrieve information from the requested URL.
- The identifier of the resource.
- The protocol version in use.

Typical request header is shown in table2.

Table 2. A client request

GET /HTTP / 1.1 \ r \ n
ACCEPT: * / * \ r \ n
Referer: http: // www.google.com \r\n
Accept-Language: en-us \ r \ n
UA-CPU: x86 \ r \ n
User-Agent: Mozilla/4.0 (Compatible; MSIS 6.0; windows NT 5.2; SV1; .NET CLR 1.1.4322) \ r\ n

The Request-Line begins with a method token. The method token indicates the method to be performed on the resource identified in the URL. Allowed types of the methods for version HTTP/1.1 are PUT, GET, POST, DELETE, TRACE and CONNECT [10]. GET is the main method used for document retrieval. After the client uses GET method in the request, the server responds with a status line, headers, and data requested by the client. So, in this paper, we shall focus on gathering HTTP request headers with GET method sent by the client.

3.1. Implementation

Keeping in view the structure of network packet and information within HTTP Request header, a simple HTTP Sniffer program is written which runs on host machine as a service to capture any outgoing HTTP request and create a log of the activity at third location without being noticed by the user. It caters for expert users who can detect this activity but will not be able to stop this service due to its dependencies on other services necessary to connect to web. The fastest way to implement and capture the network traffic without using Network Driver Interface Specification (NDIS) is to use a Winsock raw socket. Since raw socket cannot work in promiscuous mode, so it will capture only that traffic which is destined for that host.

Strategy employed in this paper for capturing and logging HTTP header information is very simple. Sniffer proposed in this paper starts as a local service at system boot time, having dependencies on other services including TCP/IP NetBIOS Helper service and Routing and Remote Access. Once started, it continuously monitors and captures outgoing IP packets, based on their source IP address (HTTP uses port 80, but other ports can be used). It distinguishes between outgoing TCP and UDP packets, parses only TCP packets to extract HTTP header data, analyzes it to find GET tag and logs the user Web request which includes requested URL, event date and time, current user logged in and name of the workstation. Proposed application creates and maintains activity logs at two different locations, one at the local host and other on the logging server. On local host, log file is created by application at run time by using the "WinDir" system variable values which is "C:\Windows\" for Windows XP and for Windows 2000 its is "c:\winnt", thus making it independent of underlying operating system.

Once activity is logged in a local log file, application will send the activity record to the local logging server. Before sending log file to the network logging server, proposed application checks status of connection with the logging server. If the connection with logging server is active it sends the activity log to the file maintained at server side, otherwise it records activity in local log file and starts network monitoring again.

In forensic investigation, date and time information can be crucial in solving cases by analyzing and auditing logs. However, system date and time can be altered by the user or system time might not be configured properly. To cater for this problem proposed application first validates the current system date and time with GMT before logging any activity. At the start of each network event the HTTP-Sniffer gets GMT time from local server and compares it with system time to compute the difference. If difference is '0', system time is correct and is logged; otherwise the difference is added to current time.

3.2. Comparative Analysis

3.2.1. Test Bed Configuration: Web forensic tools discussed in Section II and proposed application were tested and compared on a computer with Intel (R) Celeron (R) CPU 2.13 GHZ, running Microsoft Windows XP Professional, Service Pack 3 , Internet Explore version 6.0.2900.2180, with default settings. Kaspersky Antivirus 6.0 for workstation with latest updates and Antivirus and Anti Hackers setting at default level were used.

3.2.2. Anti-Forensic Application: Anti-forensics tools were configured to clear Temporary Internet files, Cookies, Registry Keys, Index.dat files, Slack space, free space and files were overwritten 3 times by using DOD 5220.22-M standard. Then, comparative analysis was carried out to monitor the performance of existing tools with proposed one and following results were obtained as shown in table 3:

Table 3. Anti-forensics tools analysis

Web Atrifacts files / Anti-Forensic Tools	IE Index.dat	IE Cookie	Registry	History.dat	Proposed-Sniffer log file
Tracks Eraser Pro	Cleared	Cleared	Cleared	Cleared	Unchanged
Secure Clean	-do-	-do-	-do-	-do-	-do-
Eraser	-do-	-do-	-do-	-do-	-do-

As listed in table 3 history files created by IE and Mozilla Firefox were cleared by wiping tools. However, History file created by application proposed in this paper was not detected and cleared.

3.2.3. Test Results: Existing Applications: Pasco and Web Historian takes '.dat' files as input. After running anti-forensic tools, both these web activity reconstructing tools were given .dat file as input. Both of the tools failed to recover the activity records.

3.2.4. Test Results: Proposed Application: The activity log file created by the proposed application was not detected and cleared by disk-wiping tools. Disk-wiping software has no way of knowing which legitimate file created by various applications should be eliminated [11]. The sniffer programs are difficult to detect, unless they are designed to advertise their existence, the proposed Sniffer application was not detected by the kaspersky antivirus software with latest updates.

3.3. Performance Analysis

Application proposed in this paper is light weight in terms of its CPU, RAM and storage requirements. Instead of analyzing and parsing data of each packet flowing through the network, it analyzes only outbound TCP packets with TCP data field >0 (some of the network sniffers like Netcat can bind a shell with any user specified port as backdoor), thereby creating less overhead on host machine.

3.4. Security Analysis

Sniffers are used to capture and log network packets. Typically, the packet sniffer only captures packets that were intended for the machine in question. However, if placed into promiscuous mode, the packet sniffer is also capable of capturing all packets traversing the network regardless of their destination. By placing a packet sniffer on a network in promiscuous mode, a malicious intruder can capture and analyze all of the network traffic. This compromises the security of data that flows across network. Also logs created by network sniffers contain entire packet information which might not be helpful for forensic investigation thus complicating the task. HTTP Sniffer proposed in this paper uses raw stream socket, captures and analyzes only those packets

destined for a particular host. This scheme not only reduces processing and storage overhead but also reduces the risk of data leakage in case the system is compromised.

4. Conclusion

The solution presented in this paper has resolved two major problems encountered by digital investigators, recovering deleted and overwritten data and determining time of the occurrence of the activity. The scheme presented here records Web activity at two different locations might seem to be creating redundancy of data but can be helpful in situations where one of the log file is deleted or corrupted thus saving investigator's time and effort. HTTP-Sniffer proposed in this paper captures and processes only outgoing IP packets without creating overloaded CPU or storage space problems. However, it provides wealth of knowledge for enterprise web policy violation and forensic analysis.

5. References

[1] Internet and Chat Forensics- Presentation Transcript [online], available from "http://www.slideshare.net/bshavers/internet-and-forensics-286361.html".

[2] Anti Forensics: making computer forensics hard [online], available from "http://ws.hackaholic.org/slides/AntiForensics-CodeBreakers2006-Translation-To-English.pdf".

[3] Web Browser Forensics, Part1 [online], available from "http://www.securityfocus.com/infocus/1827".

[4] Forensic Analysis of Internet Explorer Activity Files [online], available from "http://www.foundstone.com/us/pdf/wp_index_dat.pdf".

[5] Pasco, IE Index.dat viewer [online], available from "http://www.sourceforge.net/projects/odessa".

[6] Galleta , IE Cookie viewer [online], available from "http://www.foundstone.com/us/resources/proddesc/galleta.htm-4".

[7] Web Historian [online], available from "http://mandiant.invisionzone.com".

[8] Tracks Eraser Pro, Web activity cleaner[online], available from "http://acesoft.net/";

[9] Secure Clean, a web activity cleaner[online], available from "http://www.whitecanyon.com"

[10] Hypertext Transfer Protocol – HTTP/1.1 [online], available from "http://www.ietf.org/rfc/rfc2616.txt".

[11] Micheal A. Caloyannides, "Privacy Protection and Computer Forensics", Second Edition, Artech House, 2004, pp. 12-36