Web Server Logs Preprocessing for Web Intrusion Detection

Shaimaa Ezzat Salama
Faculty of Computers and Information, Helwan University, Egypt
E-mail: chaimaa_salama@yahoo.com

Mohamed I. Marie
Faculty of Computers and Information, Helwan University, Egypt
E-mail: mohamedmarie@yahoo.com

Laila M. El-Fangary & Yehia K. Helmy
Faculty of Computers and Information, Helwan University, Egypt
E-mail: {lailaelfangery@gmail.com, yhelemy@yahoo.com}

Received: June 4, 2011 Accepted: June 19, 2011 doi:10.5539/cis.v4n4p123

Abstract

Securing e-commerce sites has become a necessity as they process critical and sensitive data to customers and organizations. When a customer navigates through an e-commerce site his/her clicks are recorded in web log file. Analyzing these log files using data mining reveal many interesting patterns. These results are used in many different applications and recently in detecting attacks on web. In order to improve quality of data and consequently the mining results data in log files need first to be preprocessed. In this paper, we will discuss how different web log files with different formats will be combined together in one unified format using XML in order to track and extract more attacks. And because log files usually contain noisy and ambiguous data this paper will show how data will be preprocessed before applying mining process in order to detect attacks. We will also discuss the difference between log preprocessing for web intrusion and for web usage mining.

Keywords: Web log file preprocessing, Web attacks, Intrusion detection, Log file format

1. Introduction

The destruction of trust in e-commerce applications may cause business operators and clients to forgo use of the Internet for now and revert back to traditional methods of doing business. This loss of trust is being fueled by continued stories of hacker attacks on e-commerce sites and consumer data privacy abuse (N. Kumar Tyagi, A.K. Solanki, S. Tyagi,2010).

Most common attacks on e-commerce sites or on web applications in general are cross site scripting (XSS), SQL injection, denial of service (DOS) and session hijacking (R. Meyer,2008). Traditional protection mechanisms like firewalls were not designed to protect web applications and thus don't provide adequate defense. Current attacks cannot be thwarted by just blocking ports 80 (HTTP) and 443 (HTTPS).

One technique to detect web attacks is to analyze web server log files (R. Meyer,2008) (A. Hamami, M. Ala'a, S. Hasan,2006) (C. Kruegel, G. Vigna,2003). They are text files created automatically when a user accesses a web site. These files record information about each user request. This information includes user IP, the resource user requests, what type of protocol used and others (see section II). Because these log files contain information about user access behavior on a web site, analyzing these files can reveal patterns of web attacks.

The purposes of web mining (analyzing log files using data mining techniques) is to identify potential users for e-commerce, enhance quality of services provided to end users, improve web server performance and others (K.R. Suneetha, Dr. R. Krihnamoorthi,2009). Recently, it has been discovered that web mining can enhance the detection of web attacks, reduce human intervention and reduce false alarms (S. F. Yusufovna,2008) (V. Marinova-Boncheva,2007). But these log files contain noisy and ambiguous data which may affect the results of the mining process. This is why it is important to prepare log files before applying the mining algorithms (G. Castellano, A. M. Fanelli, M. A. Torsello,2007).

Many works have been devoted to preprocess data in log file for web usage mining. But few researches have been developed for preprocessing of log files for web intrusion detection. Web usage mining and web intrusion detection have different targets and thus they differ in the type of data needed for mining process.

This paper will discuss this issue in illustrating how log files will be prepared for web intrusion detection and what is the difference between log file preprocessing for web usage mining and for web intrusion detection. In addition, it will show how to integrate two different log files with different format illustrating what is called information fusion. Information fusion or information integration is the merging of information from disparate sources with different conceptual, typographical, textual representations. The more logs that can be put in context with other logs the more valuable they become. We will discuss the use of XML format to unify the different format of different log files.

This paper is ordered as follows: section 2 presents different log file format, section 3 briefs previous work, section 4 explains the preprocessing process, section 5 compares the preprocessing of log files for web intrusion with the preprocessing of log files for web usage mining, conclusion and future work are mentioned in section 6.

2. Web logs

A web server log file is a simple plain text file which records information each time a user requests a resource from a web site. This file is opened when the web services of a server starts and remain open as the server responds to user requests (K.R. Suneetha, Dr. R. Krihnamoorthi,2009)(P. Yeng, Y. Zheng, 2010)(L. Chaofeng, 2006). Web log files provide web administrators with many useful kind of information like:

- Which pages of your web site were requested
- What are the errors that people encounter
- What is the status returned by the server upon user request
- How many bytes sent from the server to the user

Analyzing these data may reveal important patterns. Generally there are four types of server logs (L.K. Joshila Grace, V.Maheswari, Dhinaharan Nagamalai,2011) :

1. Access log file
2. Error log file
3. Agent log file
4. Referrer log file

The first two types are the most commonly used. The agent and referrer logs may or may not be enabled at the server. Access log file contains data of all incoming requests and lets you track and get information about clients of the server. Error log file lists internal server errors. This information enables server administrators to correct site content or to detect anomalous activities. Agent log file provides information about user's browsers, operating system and browser version. Referrer log provides information about the link that redirects visitors to my site. Our work in this paper will focus on web access log files or simply web log files. Figure 1 is an example of one entry in a log file

The interpretation of the fields shown in figure1 is:

1) 2010-11-19: Date at which the entry is recorded.
2) 19:24:24: Time at which the entry is recorded
3) W3SVC1: Name of the server
4) 192.168.1.57: IP address of the server
5) Get: Method of HTTP request
6) sharedoutandabout/InAndOut/Categories.aspx: The resource requested
7) insid=2&langid=1: Parameters associated with the resource requested
8) 80: Port number
9) -: This is the user name if the site require user authentication. If not the hyphen is placed
10) 93.186.23.240: Client IP address

11) Mozilla/4.0+ (compatible;+MSIE+4.01;+Windows+NT): Browser name and version and the operating system.
12) 200: It is the status code returned to the user which in this case means that the request is successfully executed. There are 4 classes of status code as described in table 1 (L.K. Joshila Grace, V.Maheswari, Dhinaharan Nagamalai,2011) .
13) 3223: The bytes sent from the server to the client in response to the user request

These are the general fields that can be recorded in the log file; these fields can be customized (by adding or removing) depending on server administrators needs.

2.1 Log File Format

There are different log file format that differ in the number of parameters and fields recorded in each file and in the format of theses parameters. The most commonly used format are: NCSA common, NCSA combined, W3C extended format and IIS format.

2.1.1 NCSA format

NSCA Common format records basic information about user requests, such as remote host name, user name, date, time, request type, HTTP status code, and the number of bytes sent by the server (w3c consortium, 1995). Items are separated by spaces; time is recorded as local time. Figure 2 shows an example.

Figure 2 means that a client with user name “Fred” with IP address 172.21.13.45 requested with a “Get” method the page called “home.htm” at the date and time previously specified. The request was successfully executed and this is clear from the “200” status code. The object returned to the user was in size of 3401 bytes.

NCSA combined format records exactly the previous data and record in addition the referrer and user agent.

2.1.2 W3C extended format

It stores more information than NCSA format (msdn library, 2011). This log format can be customized that is administrators can add or remove fields depending on what information they want to record. The fields are separated by spaces and the time is recorded as GMT.

Figure 3 shows an example of one record in log file using W3C format.

Interpretation of the log entry in figure 3 is as follows:

#software: This indicates the version of IIS that is running.

#version: this indicates the log file format

#Date: this indicates the recording date and time of the first log entry, which s also the date and time of the creation of the log file.

#fields: this indicates the fields recorded in the file because this format is not standard and fields can be customized so this field is important to indicate the fields that will be recorded in the log file and in which order. Interpretation of the previous record is: at the date and time specified the client with IP 172.22.255.255 requested to download image with name “picture.jpg” from server with IP 172.30.255.255. The request doesn’t contain queries and the status code is 200. The client uses a Mozilla browser and Windows 2000 server operating system.

2.1.3 IIS format

IIS format is a fixed (cannot be customized) format (msdn library, 2011). Fields are separated by commas, making the format easier to read. The time is recorded as local time. The IIS log file format records the following data:

Client IP address, user name, date, time, service and instance, server name, server IP address, time taken, client bytes sent, serve bytes sent, service status code, windows status code, request type, target of operation, parameters. Figure 4 illustrates an example.

The focus of this paper and the combine process will be applied on NCSA format and W3C format log files.

3. Related Work

Many works have been developed and different tools are available to preprocess web log files for web usage mining. Although preprocessing is important either for web usage mining or web intrusion detection, few works have been dedicated to preprocess log files for this latter task. By enhancing the quality of data participating in the mining process, the performance and the mining results will be increased too.

Previous papers agreed on general steps to preprocess log file for web usage mining which are: data cleansing, user identification, session identification, path completion. Other steps may be added or steps may be merged depending on author approach. Figure 5 illustrates these general steps.

Researchers in (K.R. Suneetha, Dr. R. Krihnamoorthi,2009) (L. Chaofeng, 2006) (V. Chitraa, Dr. A.S. Davamani,2010) agreed on data cleansing step that include the deletion of all records that contain image extensions in the URL name. Also all requests that have status code less than 200 and greater than 299 should be removed. L. Chaofeng in paper (L. Chaofeng, 2006) recommends the deletion of robots requests.

Concerning user identification, K.R. Suneetha et al. in paper (K.R. Suneetha, Dr. R. Krihnamoorthi,2009) suggests that different IPs represent different users. If two entries have same IP but with different agents this means that they are different users. L. Chaofeng in paper (L. Chaofeng, 2006) and V. Chitraa et al. in paper (V. Chitraa, Dr. A.S. Davamani,2010) agreed on the previous rules and add another rule which is if two entries have the same IP and the same agent then the site topology must be checked. P. Yeng et al. in paper (P. Yeng, Y. Zheng, 2010) is dedicated only to user identification through inspired rules. Four constraints are used to identify users. These constraints are: IP address, agent information, site topology and time information.

The third step in preprocessing process is session identification. V. Chitraa et al. in paper (V. Chitraa, Dr. A.S. Davamani,2010) suggested two methods which are: *Time oriented heuristics* and *Navigation-Oriented heuristics*. L. Chaofeng in paper (L. Chaofeng, 2006) uses the simple assumption that if the time between 2 requests exceeds specific time (30 or 25 minutes) then this is a new session.

V. Chitraa et al. in paper (V. Chitraa, Dr. A.S. Davamani,2010) considered three approaches in path completion which are: *Reference Length approach*, *Maximal Forward Reference*, *Time Window*.

Regarding preprocessing for web intrusion detection, A. Hamami et al. in paper (A. Hamami, M. Ala'a, S. Hasan,2008) limited the preprocessing process in the conversion from log file to database without any modifications on it. R. Meyer, C. Kruegel et al. and C.J. Ezeife et al. in papers (R. Meyer,2008) (Kruegel, G. Vigna,2003) (C.J. Ezeife, J. Dong, A.K. Aggarwal,2007) respectively although they depend on log files in detecting web attacks they ignored the preprocessing of these files which can reduce their size significantly and enhance the performance of mining process.

4. Preprocessing process

Web server log files store user click streams while navigating a web site. Some of these data are unnecessary for the analysis process and could affect the detection of web attacks. Therefore, preprocessing step comes before applying mining algorithms. Unfortunately, most of the researches in this topic give no details about preprocessing steps. They just mention implicitly that these log files should be converted into suitable format. In this paper we will discuss this issue. Figure 6 illustrates steps involved in preprocessing process. It involves integrating data from multiple log files into one single file with one format. The following subsections will contain details about these steps.

4.1 Data Integration

Sometimes one web site is hosted on different servers; this will create different log files for the same web site. Integrating these log files together makes them more valuable and enables more information extraction. The format of these log files depends on the configuration of the web server. Here we will consider two formats: W3C format and NCSA common format. Log files can be combined either in text format or converted to relational database.

But we will combine them in XML files. XML files are more structured and more readable than text format and they are less complicated and require less storage space than relational database. The tags used to record entries of these two files are: user-IP, user-name, date, time, uri-stem, uri-query, status, bytes-sent.

But date and time fields are recorded in different format in the two log files. NCSA format stores time as local time and W3C format stores time as GMT. To extract consistent information from these log files the time format must be unified. Moreover, date format in NCSA is stored in the following format: DD/MMM/YYYY. Days are recorded in two digits, months are recorded as three letter characters and year is recorded in four digits. Date format in W3C takes the following format: YYYY-MM-DD. Year is recorded first in four digits, months and days are recorded in two digits. Our new standard format will follow the time and date format of W3C.

The conversion from NCSA format to XML file format is presented in algorithm 1. The algorithm reads each line in the log file and extracts each word from that line and stores it in an array called entries_array. These words represent fields or attributes stored by the log file. Algorithm 1 calls another algorithm for time conversion which is converTime. This latter algorithm will call the date conversion algorithm.

Algorithm 1: (NCSA format to XML format)

Input: Log file in NCSA format

Output: XML file containing data from log files

```
Begin
1. Open log file
2. While not end of file
    a. Read log file line in string L.
    b. For each attribute A in L
        i. Entries_array[index]=A
    c. If entries_array[index]!=4 then /* it is not the date and time entry*/
        i. Open XML file
        ii. Add XML node and its corresponding value from entries_array
       Else
        i. Call converTime(entries_array[4])
End
```

The fourth entry of the array contains the date and time attribute of the log file. The first three entries are the user IP address, the identity of the client and the user name if authentication is required.

The time conversion algorithm converts the local time of the log file to GMT. We will get benefit of how NCSA format writes the time field in converting this local time to GMT. Let's look at the following example:

[8/apr/1997:17:39:04 -0800]

An additional piece of information recorded with date and time indicates the difference between this local time and GMT. This piece of information is "-08". This means that there is a difference of 8 hours between the local time and GMT. If the sign recorded is "+", we will add specified hours to the existing time to convert it to GMT. converTime algorithm will show how this conversion process works.

converTime algorithm (string str):

Input: string containing time from log file

Output: time in GMT

```
Begin
    1. Extracts time part from str
    2. Split the time into hours, minutes, seconds variable , the time difference and the sign
    3. If sign="+" then
        i. hours=hours + time difference
        j. if hours>24 then
                a. hours=hours-24
                b. dayadd=1
        k. else
                a. dayadd=0
    4. Else
        i. hours=hours- time difference
        j. if hours<0 then
                a. hours=24+hours
                b. dayadd=-1
        k. else
                a. dayadd=0
    5. Save the new time hh/mm/ss
    6. Open XML file and store the date in its corresponding tag
    7. convertDate(string str, int dayadd)
End
```

The variable “dayadd” is used to store the number of days that will be added or subtracted from the original day in the log file in case that the hours will exceed 24 hours or will be less than 0.

convertDate algorithm is responsible for conversion of date from its current format to the following format: YYYY-MM-DD. Month is recorded in digits not in characters.

convertDate(string datestr, int dayadd) algorithm:

Input: string containing date from log file

Output: date in new format

```
Begin
    1. Extract the date part from datestr
    2. Split the date into day, month, year variables
    3. Replace the three characters of the month by its corresponding two digits
    4. Day=day + dayadd
    5. Place the previous three variables in the new order: year-month-day in the format of YYYY-MM-DD
    6. Open XML file and store the date in its corresponding tag
End
```

The conversion process of log file in W3C format is presented in algorithm 2. As mentioned in section 2, log file with this format can be customized. This means that fields are not fixed and their order is not specified as in NCSA log files. #field attribute is added to the log file for this purpose. So before extracting the attributes from the log file and storing them in XML file the algorithm should first learn the stored attributes and their order.

Algorithm 2: (W3C format to XML format)

Input: log file in W3C format

Output: XML file

```
Begin
1. Open W3C log file format
2. Read #field line in string f
3. For each attribute A in f
    i. Attribute_array[index]=A
4. While not end of file
    a. Read log file line in string L
    b. For each attribute A in L
        i. Entries_array[index]=A
    c. For each element E in attribute_array
        i. If E[index]=any of the required tags in XML then
            1. Open XML file
            2. Create tag with name in E[index]
            3. Value of the tag=entries_array[index]
End
```

The attribute_array stores recorded attributes in log file. Then the algorithm checks if the elements in the attribute_array array are all required or not. If they are required then the algorithm opens the XML file and creates a node with name containing in the array. In addition of storing attributes name, the algorithm stores user clicks in another array called entries_array like algorithm 1 do. The value of the tag created in XML is taken from that array.

These previous two algorithms have been implemented using visual studio .net, c# language and under windows vista operating system. After running algorithm 1 and algorithm 2 we end up with an XML file containing the two log files in one format. Figure 7 illustrates a snapshot of this XML file.

4.2 Data Cleansing

The purpose of data cleansing process is to remove noisy and unnecessary data that may affect the mining process. The input for this step is the XML file which contains the combined log files. The data cleansing includes the following steps:

1) Remove logEntry nodes that contain in uri-stem child node extensions like jpg, gif, css. This step is common with cleansing process in web usage mining
2) Unlike preprocessing web usage mining, status with code 400 series and 500 series should be kept because they may be considered as anomaly actions. Users that cause many errors are subject to suspect.
3) Remove logEntry nodes with successful status (200 series) and with "-"in uri-query node. Because probability of requests with such characteristics to contain web attacks is almost zero. Although query string associated with URL requests may include web attacks like XSS or SQL injection, these URL may execute successfully if these attacks are not detected. This is why any parameters should be examined even if its status is 200 series.

4.3 User Identification

In the e-commerce context unlike other web based domains user identification is a straightforward problem as in most cases customers must login using their unique ID. We are interested in anonymous users in case they cause a lot of errors or their uri-query node is not empty. It is not important to identify the identity of this user, what is important is to detect attack he/she triggers if there.

4.4 Session Identification

For known users, i.e. users logged in e-commerce sites with user name, session identification is time oriented. If the time between requests exceeds 30 minutes, this means the start of new session (L. Chaofeng, 2006) (G. Shiva, N.V. Suba, U. Dinesh,2010) (V. Sathiyamoorthi, Dr. V. Murali,2009). For anonymous users, what is important is to detect which IPs cause error in small period of time.

Unlike in web usage mining, after the preprocessing process no need to convert data in XML file to relational database. We can use XML file as input to the mining process for the detection of web attacks.

4.5 Detected attacks from log files

In the previous section we identified how log files can be preprocessed to be ready for detection of intrusions. In this section we will illustrate how the previous preparation of log files help in detection of different types of intrusion.

The log file in the format of XML stores field about the status of the requested URL, this can help in:

1) Brute force attack: the user that triggers a lot of errors in small period of time is suspicious to brute force attack. This can be detected through checking of the status field.
2) Checking status field help in identifying malicious users that trigger a lot of errors while browsing the site

Checking the uri-query field stored in log files help in detection of:

1) SQL injection, XPath injection, XSS attacks: these attacks can be detected from checking the uri-query field for dangerous keywords that may reveal the occurrence of these attacks. This checking is necessary even if the status field doesn't contain error series code.
2) The change in parameters order or the absence of some required parameters for the same page may indicate anomaly actions.

Log files also store data about the returned bytes upon a user request this helps in:

Anomaly behavior can be detected when user requests a page and its return bytes are different from other requests for the same page.

5. Differences between log files preprocessing for web intrusion and web usage mining

The previous section described attacks that can be detected from the log files that have been preprocessed. More attacks can also be detected when applying data mining techniques. Because intrusion detection and web usage mining have different purposes, the preprocessing process of log files is also different for each purpose. Table 2 summarizes these differences to better understand the difference between preprocessing process for web intrusion detection purpose and for web usage mining. For example, request with error code are deleted for web usage mining because they are of no use for this purpose but the situation is different for web intrusion detection. For the latter case, it is very important to keep these requests with error code in order to detect probable attacks.

6. Conclusion and future work

In this paper, we presented new approach in preprocessing of web log files for web intrusion detection. We discussed the different steps in this process and the differences in these steps from web usage mining. In addition, we illustrated how to combine two log files with different formats in one standard format using XML. We provided two algorithms to combine those log files. These algorithms have been implemented using c# code and under windows vista operating system.

Future work for our research is to consider all other formats for web log files in the combining process even with the user defined format.

References

A. Hamami, M. Ala'a, S. Hasan. (2006). Applying Data Mining Techniques in Intrusion Detection System on Web and Analysis of Web Usage, *Information Technology Journal,* 2006.

C.J. Ezeife, J. Dong, A.K. Aggarwal. (2007). SensorWebIDS: A Web Mining Intrusion Detection System, *International Journal of Web Information Systems,* volume 4, pp. 97-120, 2007

C. Kruegel, G. Vigna. (2003). Anomaly Detection of Web-based Attacks, *CCS,* 2003.

G. Castellano, A. M. Fanelli, M. A. Torsello. (2007). Log Data Preparation for Mining Web Usage Patterns, IADIS International Conference Applied Computing, 2007

G. Shiva, N.V. Suba, U. Dinesh. (2010). Knowledge Discovery from Web Usage Data: A survey of Web Usage Pre-processing Techniques, Springer, 2010.

K.R. Suneetha, Dr. R. Krihnamoorthi. (2009). Identifying User Behavior by Analyzing Web Server Access Log File, *IJCSNS,* 2009

L. Chaofeng. (2006). Research and Development of Data Preprocessing in Web Usage Mining, International Conference on Management Science and Engineering

L.K. Joshila Grace, V.Maheswari, Dhinaharan Nagamalai. (2011). Analysis of web logs and web user in web mining, *IJNSA,* 2011

msdn library. (2011). [online] available at: http://msdn.microsoft.com/en-us/library/ms525807(v=vs.90).aspx

N. Kumar Tyagi, A.K. Solanki, S. Tyagi. (2010). AN Algorithmic Approach To Data Preprocessing In Web Usage Mining, *International Journal of Information Technology and Knowledge Management,* 2010.

P. Yeng, Y. Zheng. (2010). Inspired Rule-Based User Identification, LNCS 6440, pp. 618-624

R. Meyer. (2008). Detecting Attacks on Web Applications from Log Files, Sans Institute, InfoSec reading room, 2008, [online] available at: http://www.sans.org/reading_room/whitepapers/logging/detecting-attacks-web-applications-log-files_2074

S. F. Yusufovna. (2008). Integrating Intrusion Detection System and Data Mining, International Symposium on Ubiquitous Multimedia Computing, 2008

V. Marinova-Boncheva. (2007). Applying a Data Mining Method for Intrusion Detection, International Conference on Computer Systems and Technologies, 2007

V. Chitraa, Dr. A.S. Davamani. (2010). A Survey on Preprocessing Methods for Web Usage Data, *IJCSIS,* 2010.

V. Sathiyamoorthi, Dr. V. Murali. (2009). Data Preparation Techniques for Web Usage Mining in World Wide Web- An Approach, *International Journal of Recent Trends in Engineering,* 2009.

w3c consortium. (1995). [online] available at :www.w3.org/Daemon/User/Config/Logging.html, 1995

Table 1. Classes of status code

Success	200 series
Redirect	300 series
Failure	400 series
Server error	500 Series

Table 2. Differences between log files preprocessing for web intrusion and for web usage mining

	General web usage	**Our Intrusion detection**
Combine log file	Same format	Different format
Data cleaning	- delete request with sound and images extension - delete request with error series status - keep other entries	- delete request with sound and images extension - essential to keep request with error series status - remove request with no parameter and with success status
User identification	Very important to identify users. Different heuristics are developed for this purpose	For application such as e-commerce or e-learning user identification is straightforward problem. In general, web intrusion main interest is to classify requests as legitimate or attack rather than identifying the user making the request
Session Identification	Different algorithms developed for such task because it is important to track user clicks in each session in order to understand user behavior and react upon this.	It is not that important. What is more important is to identify that user sends malicious requests either in same session or in different sessions. This is why we use simple way for session identification which is time oriented method
After preprocessing	Log files must be converted to relational database in order to apply data mining algorithms.	XML file will not be converted to relational database. We can use XML file as input to the mining algorithms

```
2010-11-19 19:24:24 W3SVC1 192.168.1.57 GET /sharedoutandabout/InAndOut/
Categories.aspx insid=2&langid=1    80 -    93.186.23.240 Mozilla/4.0+
(compatible;+MSIE+4.01;+Windows+NT) 200 3223
```

Figure 1. One entry in a log file

```
172.21.13.45 - fred [08/Feb/2010:16:20:14 -0800] "GET /home.htm HTTP/1.0" 200 3401
```

Figure 2. Entry in NCSA log file format

```
#software: Microsoft Internet Information Services 6.0
#version: 1.0
#Date: 2002-05-02 17:42:15
#Fields: date time c-ip cs-username s-ip s-port cs-method cs-uri-stem cs-uri-query sc-status cs(User-Agent)
2002-05-02 17:42:15 172.22.255.255 - 172.30.255.255 80 GET /images/picture.jpg – 200
Mozilla/4.0+(    compatible;MSIE+5.5;+Windows+2000+Server)
```

Figure 3. Entry in W3C log format

```
192.168.114.201, -, 05/15/11, 7:55:20, W3SVC2, SERVER, 172.21.13.45, 4502, 163, 3223, 200, 0, GET,
/index.htm, -,
```

Figure 4. Entry in IIS log file format

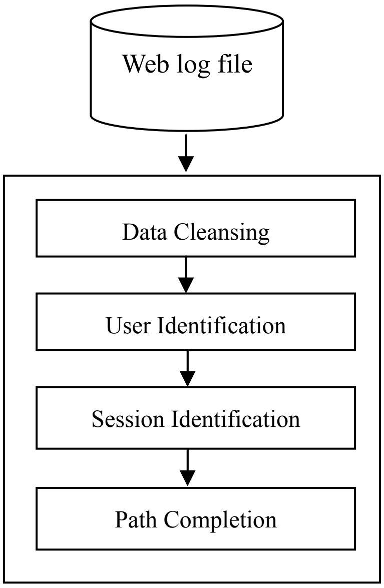


Figure 5. General Preprocessing steps

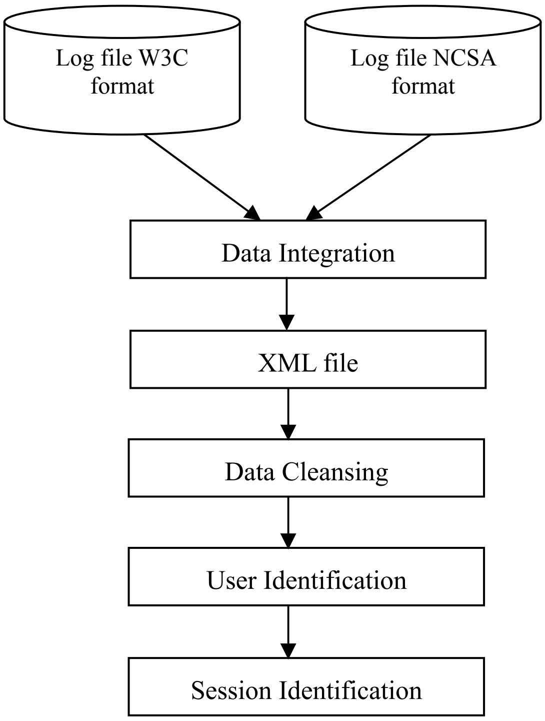
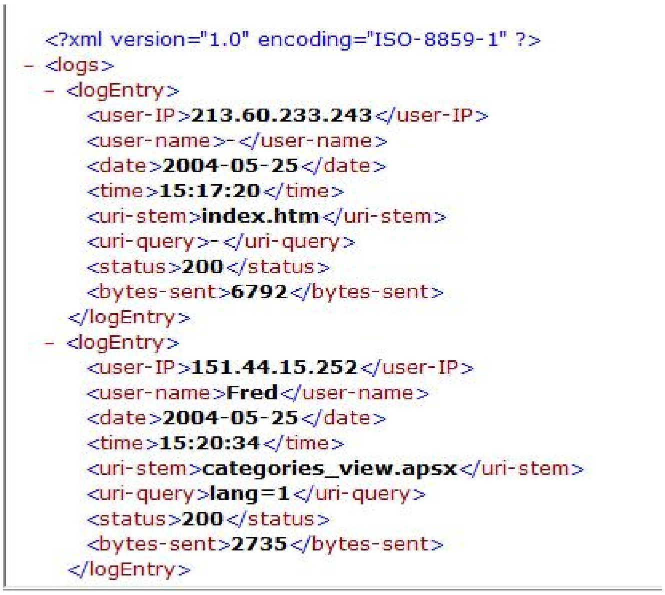


Figure 6. Log files preprocessing



```
  <?xml version="1.0" encoding="ISO-8859-1" ?>
- <logs>
  - <logEntry>
      <user-IP>213.60.233.243</user-IP>
      <user-name>-</user-name>
      <date>2004-05-25</date>
      <time>15:17:20</time>
      <uri-stem>index.htm</uri-stem>
      <uri-query>-</uri-query>
      <status>200</status>
      <bytes-sent>6792</bytes-sent>
    </logEntry>
  - <logEntry>
      <user-IP>151.44.15.252</user-IP>
      <user-name>Fred</user-name>
      <date>2004-05-25</date>
      <time>15:20:34</time>
      <uri-stem>categories_view.apsx</uri-stem>
      <uri-query>lang=1</uri-query>
      <status>200</status>
      <bytes-sent>2735</bytes-sent>
    </logEntry>
```

Figure 7. Snapshot of XML file