I. J. Computer Network and Information Security, **2017, 1, 20-30**
Published Online January 2017 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijcnis.2017.01.03

Pre-Processing of University Webserver Log Files for Intrusion Detection

Bukola A. Onyekwelu
Department of Computer Science, Joseph Ayo Babalola University, Ikeji-Arakeji, Osun State, Nigeria
E-mail: baonyekwelu@jabu.edu.ng

B. K. Alese and A. O. Adetunmbi
Department of Computer Science, Federal University of Technology, Akure, Ondo State, Nigeria
E-mail: bkalese@futa.edu.ng, kaalfad@yahoo.com, aoadetunmbi@futa.edu.ng, aoadetunmbi@futa.edu.ng

Abstract—Web Server log files can reveal lots of interesting patterns when analyzed. The results obtained can be used in various applications, one of which is detecting intrusions on the web. For good quality of data and usable results, there is the need for data preprocessing. In this research, different stages of data preprocessing were carried out on web server log files obtained over a period of five months. The stages are Data Conversion, Session Identification, Data Cleaning and Data Discretization. Data Discretization was carried out in two phases to take care of data with continuous attributes. Some comparisons were carried out on the discretized data. The paper shows that with each preprocessing step, the data becomes clearer and more usable. At the final stage, the data presented offers a wide range of opportunities for further research. Therefore, preprocessing web server log files provides a standard processing platform for adequate research using web server logs. This method is also useful in monitoring and studying web usage pattern in a particular domain. Though the research covers webserver log obtained from a University domain, and thus, reveals the pattern of web access within a university environment, it can also be applied in e-commerce and any other terrain .

Index Terms—Web Server Log, Data Preprocessing, Data Cleaning, User Identification, Discretization.

I. Introduction

There is a wide gap between data and information, and bridging this gap is one of the paramount goals in the world of Information Technology research and development. When facts and figures are processed and properly interpreted, it enhances knowledge. Over the years, various techniques have been employed to make data meaningful, one of which is data mining. Reference [1] defines data mining as the application of the computer-based methodology to discover knowledge from data. He also explained that new and valuable information can be sought in large volumes of data, using data mining.

Data in the real world can be inconsistent, noisy and incomplete. Of course, it is a well-known fact that the quality of data determines the quality of information obtained. For data mining to be carried out successfully, preprocessing is highly necessary. Tasks involved in preprocessing include Data cleaning, Data integration, Data transformation, Data reduction and Data discretization. Discretization is a process of quantizing continuous attributes. According to [2], discretization becomes necessary in a situation where the data has some discreet attributes as well as continuous attributes. Data discretization is a method of reducing data and making it simpler, easier to understand, use and explain. Data that is discretized has many advantages over continuous data. Results obtained can be systematically studied and compared. Also, various classification learning algorithms can only deal with discrete data.

In a University system, like any learning environment, there is usually a Metropolitan Area Network (MAN). The University under observation uses a VSAT and Fibre Optics link via radio to provide Internet services to users within the campus. The University runs both wired and wireless connectivity, with several antennas, radios, routers, access points and switches spread all over the campus. The ICT department manages the University Server, and also maintains the University website, a very robust website that is well loaded with lots of information for prospective and actual students, staff, parents and the general public. The site also offers Students' Portals for Course Registration, Result Checking, other registrations, as well as webmail services for staff. As a result of its robust nature, the network traffic to the site is most times heavy. The University server runs Apache 2.2.26 and is set to log according to the general log format.

The web server log files are text files created automatically when a user accesses a web site. The information obtained from the log files include 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 data obtained is passed through different stages of preprocessing and various comparisons are carried out on the results obtained.

The rest of this paper is organized as follows: In section 2, we present a review on Related Work on

various preprocessing techniques. Section 3 explains the method of data collection Section 4 describes the data preprocessing steps undertaken. Section 5 describes the data discretization processes, while Section 6 shows the results and discussions. Section 7 concludes the paper.

II. Related Work

In studying Preprocessing Methods in Web Server Logs, [3] defined web log mining (also known as web usage mining) as the extraction of interesting patterns in web access log, involving three steps, which are data preprocessing, pattern discovery and pattern analysis. They stressed that due to the large volume of irrelevant information in the web log, it is difficult to use in web log mining procedures. Preprocessing cleans up the data, making the information useful as transaction database for mining procedure and also offers structural, reliable and integrated data source to pattern discovery. They established then various phases in the data preparation process, which are data cleaning, user identification, session identification, and path completion.

Reference [4] established the preparation of web log files for web intrusion detection, unifying different formats of different log files using XML format. This integration, also known as information fusion, increases the value of the log research uses. Here, three different formats were employed, which are NCSA format, W3C Extended format, and IIS format. The preprocessing steps are more or less the same, except that they were preceded by the Information fusion. They developed algorithms to convert these formats to XML and implemented the algorithms using visual studio .net, c# language and under windows vista operating system.

In [2], preprocessing is classified into steps. First, some categorical features were mapped to integer values ranging from 1 to N, N being the total number of symbol variation in each feature. Secondly, continuous-value attributes are standardized based on equal bin partitioning, and Boolean feature with values 0 or 1 was left unchanged. Discretization of the continuous attributes in the dataset was based on Entropy, which is a supervised splitting technique exploring class distribution information in its calculation and determination of split-point.

Reference [11] used the method of identifying user navigation pattern by employing the Naïve Bayse algorithm for classification, in order to obtain results of preprocessed web server log. Reference [12] developed a system of Intrusion Detection using web usage mining, by employing various preprocessing steps which include preparation, clean up and symbolization, where long strings are symbolized to aid fast calculations. Other steps defined by [12] are User identification and session identification. On the preprocessed data, [12] went on to identify scanners and employ signature-based detection.

Reference [13] also employed all the methods used by others, but included a step known as Path Completion, to take care of the problems created by incomplete paths. Results obtained showed where several Robot hosts and robot user agents where identified. Reference [14] used the Web Personalisation process to customize the content and structure of the site in order to obtain the user's navigation behavior. Steps identified in the Web Personalization process are data collection, data modelling and categorization, data analysis and action determination. In Reference [15], a detailed analysis was carried out on various available web techniques and their uses. Various Web content mining tools were also analysed, with respect to their applications in business environment.

III. Data Collection

Web server log files were obtained from a University web server for a period of six months, and a large volume of web usage data was obtained from the log files. Figure 1 shows a sample of the web usage data.

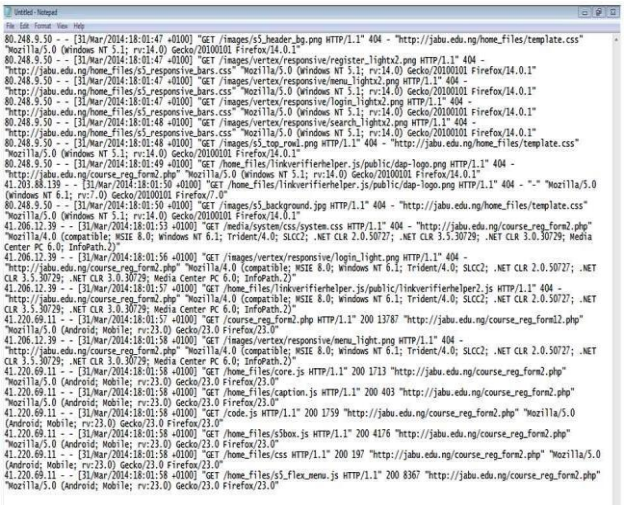

Fig.1. Sample of Web Usage Data

Reference [5] described Web Server logs as plain text (ASCII) files obtained from the server. Here four types of server logs are described, which are, Access Log, Agent Log, Error Log, and Referrer Log. The web server log obtained for this research is the Access Log. A typical example of a few lines of text is

46.148.30.106 - - [31/Aug/2014:18:11:33 +0100] "GET /administrator/index.php HTTP/1.1" 200 2246 "-" "Mozilla/5.0 (Windows NT 5.1) AppleWebKit/537.1 (KHTML, like Gecko) Chrome/24.0.1309.0 Safari/537.17"

46.148.30.106 - - [31/Aug/2014:18:11:34 +0100] "POST /administrator/index.php HTTP/1.1" 303 20 "-" "Mozilla/5.0 (Windows NT 5.1) AppleWebKit/537.1 (KHTML, like Gecko) Chrome/24.0.1309.0 Safari/537.17"

46.148.30.106 - - [31/Aug/2014:18:11:35 +0100] "GET /administrator/index.php HTTP/1.1" 200 2348 "-" "Mozilla/5.0 (Windows NT 5.1) AppleWebKit/537.1

(KHTML, like Gecko) Chrome/24.0.1309.0 Safari/537.17"

The above lines of text can be interpreted as follows:

a. 46.148.30.106 - Client IP address
b. 31/Aug/2014 - Date at which the entry is recorded.
c. 18:11:33: Time at which the entry is recorded
d. Get: Method of HTTP request
e. administrator/index.php -The resource requested
f. 200- It is the status code returned to the user which in this case means that the request is successfully
g. 2246: The bytes sent from the server to the client in response to the user request
h. "Mozilla/5.0 (Windows NT 5.1) AppleWebKit/537.1 (KHTML, like Gecko) Chrome/24.0. 309.0 Safari/537.17": Browser name and version and the operating system.

IV. Data Preprocessing

Data in the real world can be inconsistent, noisy and incomplete. Of course, it is a well-known fact that the quality of data determines the quality of result obtained. This makes the concept of data preprocessing highly necessary. Data preprocessing involves several tasks, which are:

i. Data cleaning – where missing values are filled in, noisy data is smoothened, outliers are identified and removed, and inconsistencies are resolved.
ii. Data integration – where multiple databases, data cubes or files are integrated
iii. Data transformation – where Normalization and aggregation are carried out.
iv. Data reduction – where reduced representation is obtained in volume but the same or similar analytical results are produced.
v. Data discretization – still a part of data reduction but where particular importance is placed on numerical data.

According to [2], in a situation where data is made up of attribute values that are both or either discreet and/or continuous, there is a need for discretization. In this research, data discretization becomes imperative because of the nature and volume of the data. Reference [5] explains with illustration, preprocessing as one major step involved in Web Usage Mining. In this research, the preprocessing steps employed are illustrated in Figure 2 .

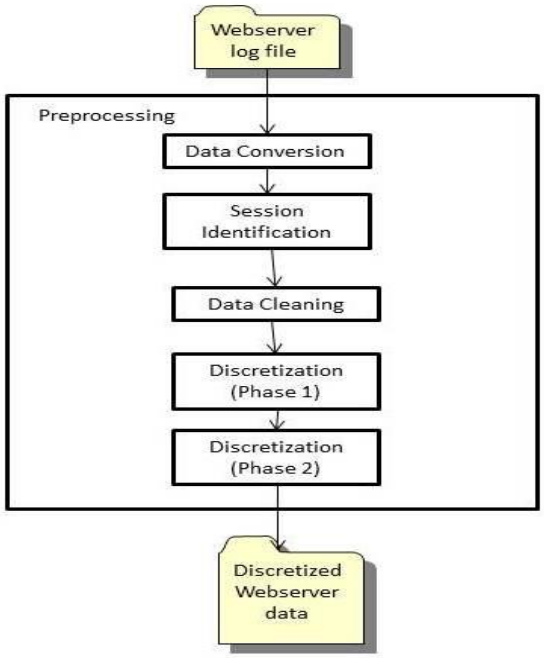

Fig.2. Web Server Log File Preprocessing

According to [6], data cleaning involves the removal of irrelevant references to embedded objects which are unimportant to the analysis. In this research, Data conversion was first carried out by converting the webserver log files from text files to Spreadsheet table formats, where each field was identified and separated. The resulting fields are described in Table 1.

Table 1. Webserver Log Table

Field num	Field Names	Description
1.	Source-IP	This is the IP address of the user making request to the server
2	Date and Time	Gives the exact date and time of the access
3.	The Access Method, Resource, and Service	This field contains the method used, the information requested by the user and the protocol used.
4.	Status Code	This code is sent by the server. Codes beginning with 2 mean that the response is successful, beginning with 3 means redirection, 4 means error caused by the user, and 5 means error in the server
5	Bytes Received	This is the size of the object that the server returns to the user
6.	Destination IP	This is the IP address of the requested service
	Browser/Operating System Platform	This describes the particular browser and the operating system from which the user made the request

For the Session Identification process, the Date and Time field was split into two, Date field and Time field. The file was sorted primarily by Source-IP, then by date and time. The Start Time and End Time of each Source-IP were obtained (in seconds). The aim is to be able to calculate the duration of each Source-IP on the system by finding the difference between the Start time and the end time. The Duration Algorithm is outlined below.

Let S = Source_IP, T_0 = Time, T_1 = Start_Time, T_2 = End_Time, D = Duration

1. Get S
2. If $S = S-1$, Then $T_1 = 0$, else, $T_1 = T_0$. T_1 is converted to seconds
3. If $S = S+1$, Then $T_2 = 0$, else, $T_2 = T_0$. T_2 is converted to seconds
4. $D = (T_2-T_1)+1$. [+1 takes care of the present second]

In other words, a new Start_Time T_1 is obtained whenever the Source_IP changes and End_Time T_2 is obtained when the present Source_IP is different from the next Source_IP.

The third field is also further broken down into three fields, namely, the REQUEST_METHOD, RESOURCE, and SERVICE.

a. REQEST_METHOD: This is the method or request type used in information transfer. It was observed that there are 12 classes, identified in the log files, and shown in Table 2. Therefore, discretizing this field will result in classes of 1 to 12.

Table 2. Types of Request_Method

S/N	TYPES OF REQUEST_METHOD
1	GET
2	POST
3	ACUNETIX
4	COOK
5	DEBUG
6	HEAD
7	NETSPARKER
8	OPTION
9	PROPFIND
10	PUT
11	TRACE
12	TRACK

b. SERVICE: This describes the protocol used by the client, and they are mostly HTTP 1.0 and HTTP 1.1. As a result, classification is done by representing HTTP 1.0 by 1, HTTP 1.1 by 2, and others by 0.

c. RESOURCE: This describes the information requested by the client. This field is categorized based on normal, as well as some anomalous characteristics. Ezeife et al [7] described these anomalous behaviors, and some of them are discussed.

i. Cross Site Scripting: This is said to occur when a web application gathers malicious data from a user through cookie theft. The vulnerabilities are shown in the form of string, for example, < script > alert(document.cookie) < /script >.

ii. SQL-Injection: described as a method used to exploit web applications that accept user data in SQL queries without first stripping possibly harmful characters. In attack mode, SQL keywords are introduced secretly into the username or password, for instance, with the use of dangerous characters like single-quotes or double-quotes.

iii. Unicode attacks – mainly launched against the Microsoft HE web server. Here, the attacker attempts access to shells and scripts by providing a path argument that steps backward (up) in the file tree and then down into a system directory by escaping the offending "\..\" sequence in various ways.

iv. Buffer overrun - recognizable from the prefix and length of the path,and also that they contain long runs of the same character ("A" or "N" respectively).

Some of these attributes are outlined in Table 3

Table 3. Resource Attributes

S/N	Resource characteristics	Attributes
1.	Normal	Normal
2.	Cross Site Scripting	Contains < script >
3.	SQL Injection	Containing single or double quotes
4.	Unicode Attacks	e.g. "\..\", %C0AE%C0AE%C0AF%C0AE%C0AE%C0A
5.	Proxy attacks	http/www appearing in the middle of a url

OS Platform: Some Operating Systems from which the access was launched are also analyzed and categorized into 12. This is shown in Table 4.

Table 4. Operating Systems Platform

S/N	OS Platform
1	Macintosh
2	Windows
3	Linux
4	iOS
5	Android
6	Blackberry
7	compatible; MJ12b
8	J2ME/MIDP
9	UCWEB
10	Series60
11	SpreadTrum
12	Series40
13	AppleWebkit
14	NokiaOS

A platform was developed to carry out the overall Data Cleaning process, using Visual Studio C#. The Algorithm for the Data Cleaning is itemized below:

1. Read the data file.
2. Split each line read from data file delimited by comma (,)
3. Iterate through all the data read from the file line by line
4. Check the Resource (column 7) and do the following:

i. If the resource value contains '<script', replace with Cross Site Scripting

ii. The resource value is double (" ") or single (' ')

quoted, replace with SQL injection
iii. If it contains "\\..\\" or "%C0AE%C0AE%C0AF%C0AE%C0AE%C0A", replace with Unicode Attacks
iv. If it contains http/www, replace with proxy attacks
v. If it contains cgi-bin, replace with Cgi-bin
vi. Otherwise, replace with Normal

5. Check the Services (column 8) and do the following:

i. If it contains 1.0, replace with HTTP/1.0,
ii. Otherwise, replace with HTTP/1.1

6. Check the OS Platform (column 13) and do the following:

i. If it is macintosh, replace with Macintosh
ii. If it is window, replace with Windows
iii. If it is linux, replace with Linux
iv. If it is ios, replace with iOS
v. If it is android, replace with Android
vi. If it is blackberry, replace with Blackberry
vii. If it is compatible or mJ12b, replace with compatible; MJ12b
viii. If it is j2me/midp, replace with J2ME/MIDP
ix. If it is ucweb, replace with UCWEB
x. If it is series60, replace with Series60
xi. If it is spreadtrum, replace with Spreadtrum
xii. If it is series40, replace with Series40
xiii. If it is applewebkit, replace with AppleWebkit
xiv. If it is nokiaos, replace with nokiaOS
xv. Otherwise, replace with iOS

7. Check if all the lines have been read otherwise go to line 3.
8. Save file with the modifications
9. End

The platform created to carry out the Data Cleaning process is shown in Figure 3.

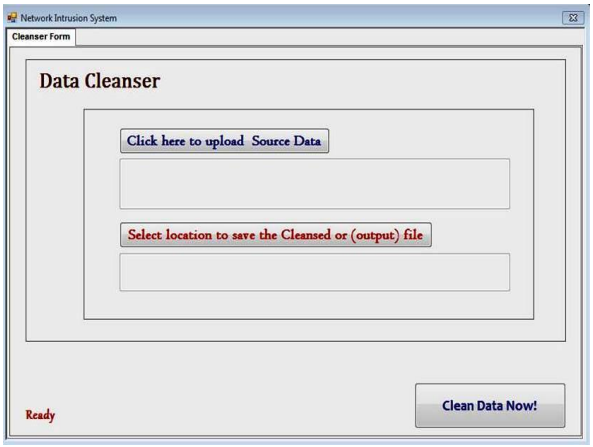

Fig.3. Data Cleaning Platform

The output data was generated in Comma delimited text file format (csv) and is shown in figure 4.

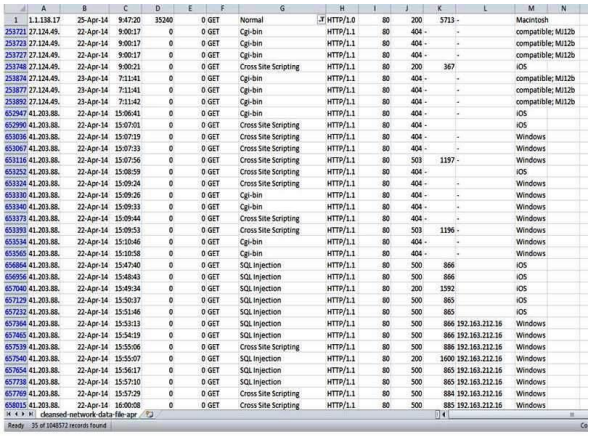

Fig.4. Data Cleaning Output

The output file shows the Source_IP, Date, Time, Start_Time, End_Time, Method, Resources, Services, Port, Status Code, Bytes Received, Destination IP and OS_Platform.

V. Data Discretization

Reference [8] defines Data Discretization as a process of quantizing continuous attributes. This is done mainly to reduce and simplify the data, since discreet attributes are easier to understand, use and expound. Data Discretization is also defined as a set of cuts over domains of attributes, which indicates an important preprocessing task for numeric data analysis [2]. According to [9], the process of discretization transforms a variable, either discrete or continuous, such that it can take a fewer number of values by creating a set of contiguous intervals (or evenly a set of cut points) that spans the range of the variable's values. The main reason for Data Discretization is the fact that learning methods in the field of data mining handle discrete attributes of data better than continuous [10].

In this research, discretization was carried out in two phases, namely

1. Phase 1 - Discretization of fields with discrete values
2. Phase 2 - Discretization of fields with continuous values.

A. Phase 1 - Discretization of fields with discrete values

Here, discrete values (numbers) are computed for each element. From Tables 2 to 4, variations can be drawn up for each of the fields with discrete values, as shown in Table 5 below.

Table 5. Table Showing Variations

S/N	Field	Number of variations
1.	Source_IP	Number of individual Source IP
2.	Method	12
3.	Resources	6
4.	Services	2
5.	OS Platform	14

A platform was also developed to carry out Phase 1 discretization, using Visual Studio C#. The Algorithm is listed below:

1. Select Cleaned file to be discretized
2. Read data
3. Separate data into different categories
4. Select distinct discrete values

   i. Select distinct source_ip, store in array and start indexing from 1 to the last value in the array
   ii. Repeat step (i) above for method, resource, services and os_platform

5. Iterate through the data file line by line

The Platform for the Phase 1 discretization is shown in figure 5. Here, the input file is the output file from the data cleaning process. The resulting output file is shown in figure 6.

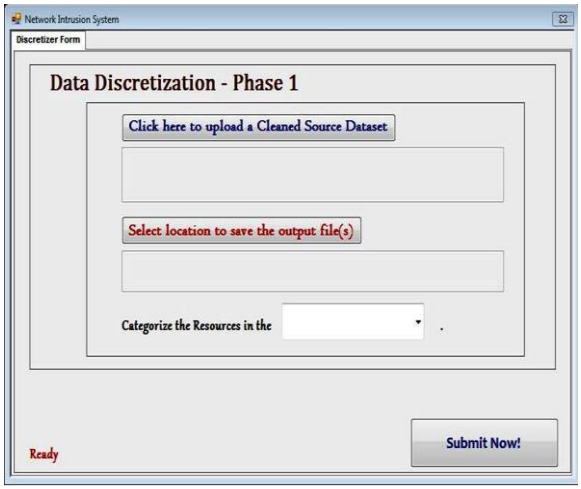

Fig.5. Data Discretization Platform

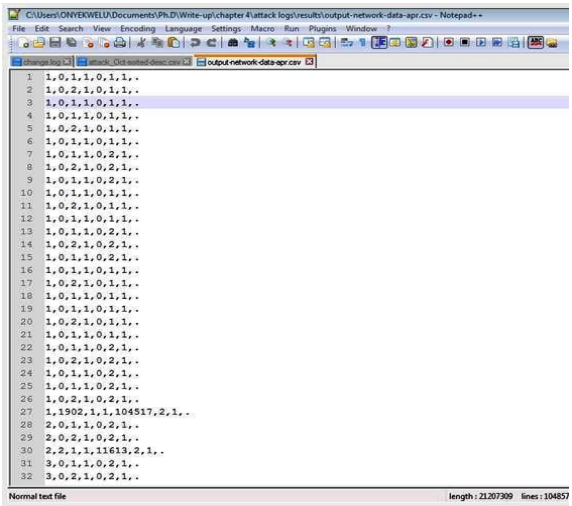

Fig.6. Discretized Output

After discretizing, the unneeded fields are removed, and some of the remaining fields were discretized. Here, the output shows Source_IP, Method, Resources, Service and OS_Platforms as having discrete values, while the Duration and Total Bytes received still have continuous values. These are taken care of in Phase II Discretization stage.

B. Phase 2 - Discretization of fields with continuous values using Equal Width Interval Technique

Equal Width discretization is a simple but straightforward means of discretizing data with continuous attributes. According to Adetunmbi(2008), it is carried out by establishing the range of observed values of a particular attribute, $a \in A$, where $[v^a_{min}, v^a_{max}]$) are the minimum and maximum values of attribute a respectively. The continuous intervals are further divided into k_a sub-intervals of equal sizes, where k_a is a user-determined value, $k_a \in N$. Thus, the width W_a can then be computed as

$$W_a = \frac{v^a_{max} - v^a_{min}}{k_a} \quad (1)$$

At the end of Phase 1, the output obtained still had continuous values in the fields SOURCE_IP and TOTAL_BYTES_RECEIVED, as shown in Figure 6. For the three fields being discretized, k is set to 50, and the equal-width intervals were computed via a Visual C++ code, and results were obtained, as shown in Figure 7.

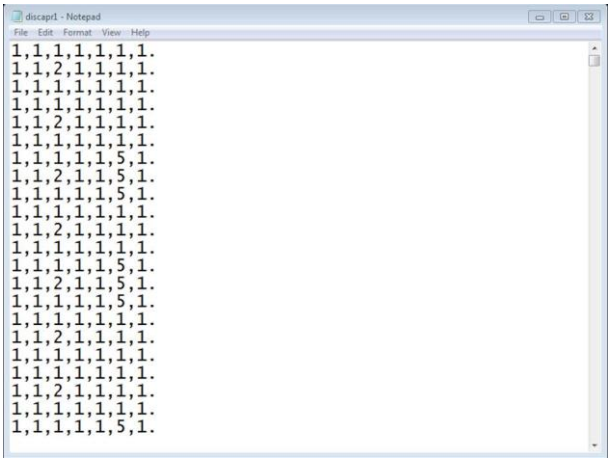

Fig.7. Equal Width Discretization

Here, all the fields are discretized. They have been reduced to 7, that is, Source_IP, Duration, Method, Resource, Service, Total_Bytes received, and OS_Platform.

VI. Results And Discussions

From the discretized output, some analyses were carried out, based on the resource types, comparison of the various resource types based on the number of hits (records), as well as the analysis of anomalous resources based on the Operating System Platforms. Of the six months' web server logs under observation, anomalous resources were identified in 3 months.

6.1. Comparison by Resource Types

Identifying the three months as Month1, Month 2 and Month 3, the tables depicting the rate of the various resources are shown

Table 6. Overall Analysis for Month 1

Code	Resource Type	Number of hits
1	Normal	1,048,539
2	CSS	10
3	SQL Inj	13
4	Unicode	11
	Total hits	1,048,573

For Month1, as shown in Table 6, of the 1,048,573 records obtained, 34 records were shown to be anomalous. Three different types of anomalous records were identified, which are CSS, SQL Injection, and Unicode. The characteristics of these anomalies were discussed in Section 4. The occurrence of these anomalies is minimal. This is more clearly illustrated in figure 8.

Table 7. Overall Analysis for Month 2

Code	Resource Type	Number of hits
1	Normal	697,088
2	CSS	1
	Total hits	697,089

For Month2, as shown in Table 7, of the 697,088 records obtained, 1 record was shown to be anomalous, which is CSS. This is illustrated in figure 9.

Table 8. Overall Analysis for Month 3

Code	Resource Type	Number of hits
1	Normal	1,048,554
2	CSS	14
3	SQL Inj	3
4	Unicode	1
	Total hits	1,048,572

For Month3, as shown in Table 8, of the 1,048,554 records obtained, 18 records were shown to be anomalous,CSS, SQL Inj, and Unicode. Their occurrences are also minimal. This is illustrated in figure 10.

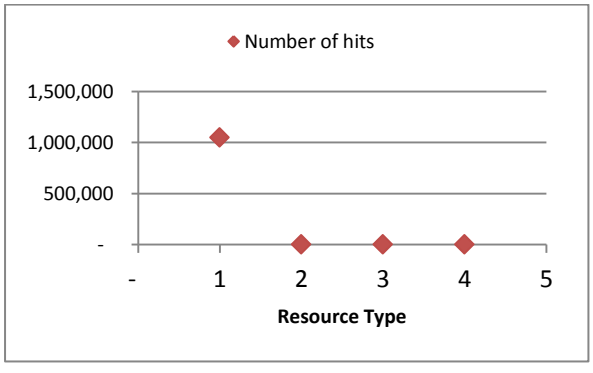

Fig.8. Analysis of Month 1

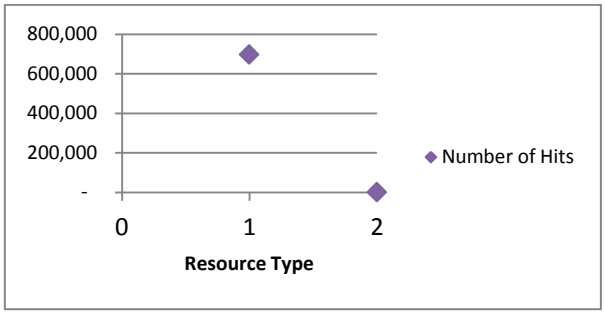

Fig.9. Analysis of Month 2

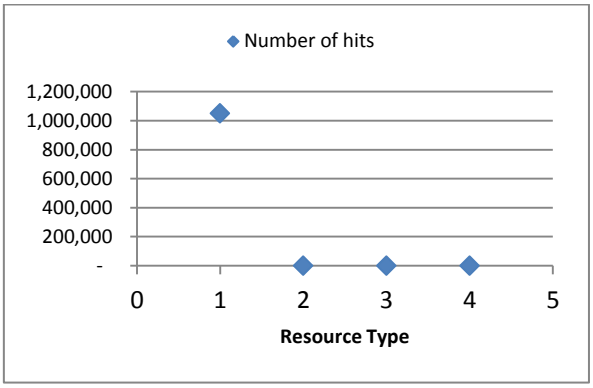

Fig.10. Analysis of Month 3

From tables 6 to 8 and figures 8 to 10 above, it is clear that of the 5 months under observation, anomalous resources were observed in 3 months. However, from the above, we can see that anomalous traffics are very minimal, compared to normal traffic.

6.2. Anomalous Resource Types Comparison

Having observed that the anomalous traffics are considerably minimal in the 3 months, a comparison was also made based on the different anomalies. This comparison was carried out for Month 1 and Month 3. For Month 2, there was only one other resource type apart from NORMAL, and this presents no basis for comparison. The comparisons are shown in Tables 9 and 10.

Table 9. Anomalies Comparison for Month 1

IP	CSS	SQLI	UNICODE	Total
IP5724	6	1	0	7
IP7270	4	12	10	26
IP8072	0	0	1	1
Total	10	13	11	34

From Table 9, we observe that the anomalous requests were carried out from three definite sources, IP5724, IP7270, and IP8072. Of the three sources, IP7270 was prominent, having 76.47% of the anomalies. IP5724 had 20.59%, while IP8072 had 2.94%. This is further illustrated in figure 11.

Table 10. Anomalies Comparison for Month 3

IP	CSS	SQLI	UNICODE	Total
IP2263	1	0	0	1
IP2412	4	0	0	4
IP5883	9	3	0	12
IP7791	0	0	1	1
Total	14	3	1	18

Table 10 likewise shows anomalies from 4 different sources, IP2263, IP2412,IP5883 and IP7791. Their percentage rates are as follows:

IP2263 = 5.56%
IP2412 = 22.22%
IP5883 = 66.66%
IP7791 = 5.56%

This is illustrated in Figure 11.

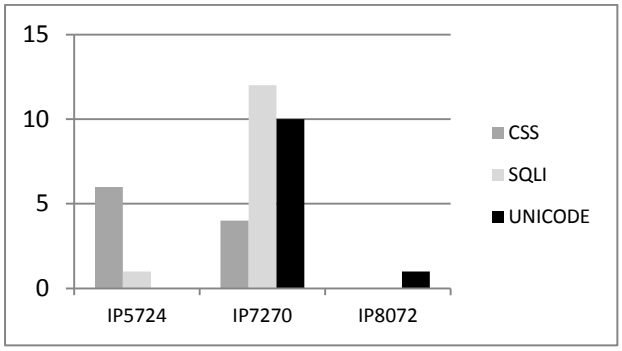

Fig.11. Comparison for Month 1

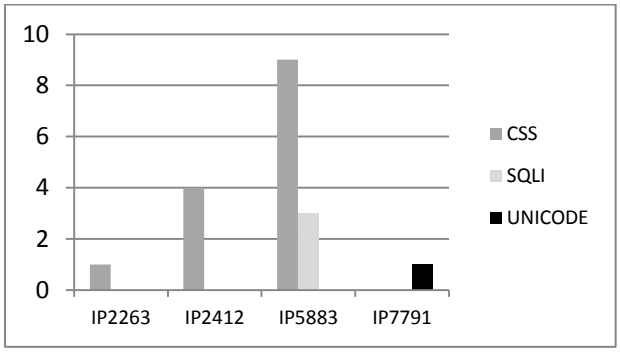

Fig.12. Comparison for Month 3

From the comparisons above, it is clear that the anomalies are varied. In Month 1, each of the 3 anomalies featured from two sources. However, SQL Injection, had the highest number of hits. On the other hand, In Month 3, out of the four different sources from where anomalies were recorded, two were Cross Site Scripting, one had both CSS and SQL Injection, while the last source recorded only Unicode attacks. CSS had the highest hit.

6.3. Comparison based on Operating System

As earlier illustrated in Table 4, the Operating System Platforms from which anomalies were observed are,

s/n	OS Platform
1	Macintosh
2	Windows
3	Linux
4	iOS
5	Android

For Month 1, it was observed that the anomalies were launched from 3 of the Operating Systems listed above. Likewise, Month 3 shows that the anomalies were launched also from 3 different operating systems. These are shown in Tables 11 and 12 respectively, and illustrated in Figures 13 and 14 respectively.

Table 11. Comparison Based On Os For Month 1

	OS2	OS4	OS5
CSS	4	1	6
SQLI	10	3	0
UNICODE	6	5	0

The data in Table 11 shows that, of the 11 occurrences of CSS in Month1, 54.55% came from OS1, which represents Android Operating Systems running on Mobile devices, 36.37% from OS2, which represents the Windows Operating System running either on Mobile devices or Personal Computers, and 9.08% from OS4, which represents iOS running from Apple devices. There is a slight variation with SQL Inj, which recorded 76.92% of its occurrences on the Windows Operating System running either on Mobile devices or Personal Computers, and 23.08% occurrences on iOS. The Unicode showed almost the same pattern, with 54.55% of its occurrences on the Windows Operating System running either on Mobile devices or Personal Computers, and 45.45% occurrences on iOS.

Table 12. Comparison Based On Os For Month 3

	OS1	OS2	OS3
CSS	5	0	9
SQLI	0	0	3
UNICODE	0	0	1

Unlike Month1, the comparison based on Operating Systems in Month3 is varied. As a result, no fixed conclusion can be drawn from this. The comparisons are further illustrated in figures 13 and 14.

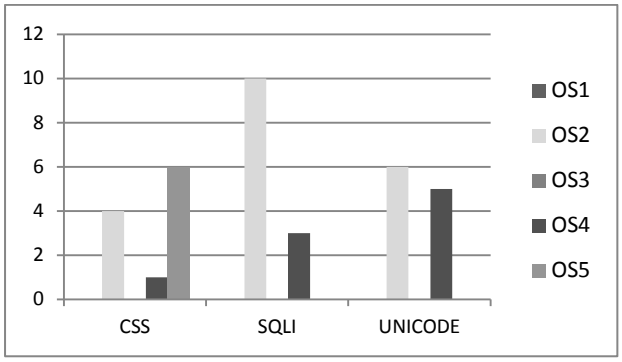

Fig.13. Comparison based on OS for Month 1

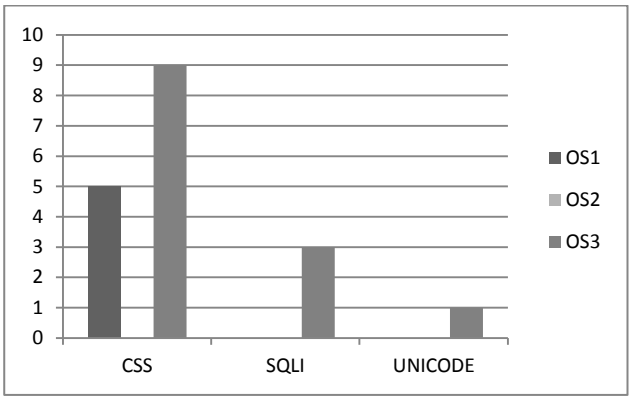

Fig.14. Comparison based on OS for Month 3

The above results show clear analysis of web server log files to detect anomalies, which could be web attacks. More detailed analysis of these files can also reveal patterns of anomalies. Due to the noisy and ambiguous nature of web server log files, they are difficult to analyze and interpret. It is therefore certain that improving data quality, thereby enhancing data mining results can be achieved through preprocessing. Good data mining results can in turn aid in the detection of web attacks and other anomalies while reducing false alarms.

Comparing the web usage data, from the raw state all through the various preprocessing stages, it becomes clear that with each preprocessing step, the data becomes clearer and more usable. At the final stage, the data presented offers a wide range of opportunities for further research.

VII. Conclusion

Preprocessing web server log files provides a standard processing platform for adequate research using web server logs. Also, it can be used to monitor and study web usage pattern in a particular domain. This research covers webserver log obtained from a University domain, and thus, reveals the pattern of web access within a university environment. It is applicable in e-commerce terrain and any other domain, as the case may be. From the results illustrated in the various graphs in this framework, the high level of security in the particular University web server, within the months under investigation, is clearly seen.

References

[1] Kantardzic, M. (2011). Data Mining: Concepts, Models, Methods, and Algorithms, Second Edition. Institute of Electrical and Electronics Engineers. John Wiley & Sons, Inc.

[2] Adetunmbi, A. O. (2008): Intrusion Detection Based On Machine Learning Techniques, A Ph.D. Theses in the Department of Computer Science, Federal University of Technology, Akure, Nigeria.

[3] Dhawan, S., and Lathwal, M. (2013). Study of Preprocessing Methods in Web Server Logs. *International Journal of Advanced Research in Computer Science and Software Engineering. Volume 3, Issue 5. ISSN: 2277 128X*.

[4] Salama, S. E., Marie, M. I., El-Fangary, L. M. and Helmy, Y. K. (2011). Web Server Logs Preprocessing for Web Intrusion Detection. *Computer and Information Science, Vol. 4, No. 4.*

[5] Grace, L.K J, Maheswari, V., and Nagamalai, D (2011). Analysis of web logs and web user in web mining, *International Journal of Network Security & Its Applications (IJNSA), Vol.3, No.1, http://arxiv.org/ftp/arxiv/papers/1101/1101.5668.pdf* Accessed on November 10, 2014.

[6] Revathi, T., Mohana, R. M., Sasanka, C. SKumar, K. J., Kiran, B. U. (2012). An Enhanced Pre-Processing Research Framework for Web Log Data. *International Journal of Advanced Research in Computer Science and Software Engineering Research, Volume 2, Issue 3. ISSN: 2277 128X. http://www.ijarcsse.com /docs/papers/March2012/volume_2_Issue_3 /V2I300119.pdf.* Accessed on November 15, 2014.

[7] Ezeife, C. I., Dong, J., and Aggarwal, A. K. (2008): SensorWebIDS: A Web Mining Intrusion Detection System. *12th International Database Engineering and Applications Symposium {(IDEAS} 2008), September 10-12, 2008, Coimbra, Portugal}, http://dblp.uni-trier.de/rec /bib/conf/ideas/EzeifeEA08.* Accessed on May 8, 2015.

[8] Liu, H., Hussain, F., Tan, C., Dash, M. (2002). Discretization: An Enabling Technique. *Data Mining and Knowledge Discovery, 6, 393–423, Kluwer Academic Publishers.*

[9] Lustgarten, J., Visweswaran, S., Gopalakrishnan, V., and Cooper, G. (2011). Application of an efficient Bayesian discretization method to biomedical data, in *BMC Bioinformatics. http://www.biomedcentral. com/1471-2105/12/309.* Accessed on August 9, 2014.

[10] Grzymala-Busse J.W., & Stefanowski J. (2001). Three Discretization Methods for Rule Induction, *International Journal of Intelligent Systems, 16,* 29-38.

[11] Chandrama, W., Devale, P. R., Murumkar, R. (2014). Data Preprocessing Method of Web Usage Mining for Data Cleaning and Identifying User Navigational Pattern, *IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 1 Issue 10, ISSN 2348 – 7968.*

[12] Mabzool, M., and Lighvan, M. Z., (2014) Intrusion Detection System Based On Web Usage Mining, *International Journal of Computer Science, Engineering and Applications (IJCSEA) Vol.4, No.1.*

[13] Kharwar, A. R., Naik, C. A., Desai, N. K. (2014), A Complete PreProcessing Method for Web Usage Mining, *International Journal of Emerging Technology and Advanced Engineering, Volume 3, Issue 10, ISSN 2250-2459.*

[14] Losarwar, V., and Joshi, M. (2012), Data Preprocessing in

Web Usage Mining, *International Conference on Artificial Intelligence and Embedded Systems (ICAIES'2012) July 15-16, 2012 Singapore*.

[15] Upadhyay, G. M., and Dhingra, K. (2013), Web Content Mining: Its Techniques and Uses, *International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 11, ISSN: 2277 128X*.

Authors' Profiles

Onyekwelu, Bukola Abimbola Place & Date of Birth: Ado-Ekiti, Nigeria, 05 Dec. 1970.

Educational Background:

- *Ph.D. in Computer Science,* Federal University of Technology, Akure, Ondo State., Nigeria - 2015
- *M.Tech in Computer Science,* Federal University of Technology, Akure, Ondo State, Nigeria-2009
- *Post-Graduate Diploma in Computer Science,* Federal University of Technology, Akure, Ondo State, Nigeria - 2004
- *B.Tech in Computer Science,* Federal University of Technology, Akure, Ondo State, Nigeria-1991

Research/Area of Interest: CyberSecurity, Intrusion Detection, Science, Technology, Engineering and Mathematics (STEM) Initiatives,

Working Experience

- **Lecturer II,** Department of Computer Science, College of Natural Sciences, Joseph Ayo Babalola University, Ikeji-Arakeji, Osun State. October 2013 to date
- **Assistant Lecturer,** Department of Computer Science, College of Natural Sciences, Joseph Ayo Babalola University, Ikeji-Arakeji, Osun State. March 2010 to Sept 2013
- **Lecturer (Part-time),** Department of Computer Studies, Federal Polytechnic, Ado-Ekiti, Ekiti State. May to August 2009

Membership of Professional Bodies

- Member, Society of Digital Information and Wireless Communications (SDIWC)
- Member, Computer Professionals of Nigeria, MCPN (Membership Number – 003460/2012)
- Member, Nigerian Computer Society, NCS (Membership Number – 08327)
- Member, Nigerian Women In Information Technology, NIWIIT, (Membership number - 00118)
- Member, Organization for Women in Science for the Developing World (OWSD)

ALESE, Boniface Kayode Place & Date of Birth: Ilawe – Ekiti, Nigeria. 21 December, 1969

Educational Background:

- *Ph.D. in Computer Science,* Federal University of Technology, Akure, Ondo State., Nigeria - 2004
- *M.Tech in Computer Science,* Federal University of Technology, Akure, Ondo State, Nigeria-2000
- *B.Tech in Industrial Mathematics,* Federal University of Technology, Akure, Ondo State, Nigeria-1997
- *NCE in Mathematics/Physics,* Ondo State College of Education, Ikere-Ekiti, Ekiti State, Nigeria - 1990

Research/Area of Interest: CyberSecurity, Intrusion Detection.

Working Experience

Professor, Department of Computer Science, Federal University of Technology, Akure, Nigerai

Membership of Professional Bodies

- Member, Nigeria Computer Society, MNCS (Membership Number 05310)
- Member, Institute of Electrical and Electronics Engineering (Computer Society) New York, MIEEE (Membership Number 80661453)
- Member, Association for Computing Machinery; New York, MACM (Membership Number 0273409)
- Member, International Association of Engineers (Membership Number 135763)
- Member, Infonomics Society, United Kingdom
- Computer Professional of Nigeria (CPN) Registered (Reg No 003686/2012)
- Member, Cyber Security Experts Association of Nigeria
- Member, Information Systems Security Association, Virginia, United States (Membership Number 38991819)
- Member, Information Systems Audit and Control Association, United States of America (Membership Number, 964629)

Dr. A.O. Adetunmbi MIEEE, MCPN Adebayo Olusola Adetunmbi received Bachelor of Technology (B.Tech), Master of Technology and Ph.D degrees in Computer Science from the Federal University of Technology, Akure, Nigeria. He is a Reader in the Department of Computer Science, Federal University of Technology, Akure. He was a Post Graduate Research Fellow of the Institute of Computing Technology, China between March 2006 to March 2007 under the CAS-TWAS Postgraduate fellowship programme, and a visiting scholar to Massachusetts Institute of Technology, Boston under the MIT International Science and Technology Initiatives (Empowering the Teachers Program) co-sponsored by Total and Google. He has published a number of articles at both local and International reputable journals. His

research interests are Information Security, Machine Learning and Natural Language Processing. He is a member of Computer Professional of Nigeria (CPN) and IEEE.

How to cite this paper: Bukola A. Onyekwelu, B. K. Alese, A. O. Adetunmbi,"Pre-Processing of University Webserver Log Files for Intrusion Detection", International Journal of Computer Network and Information Security(IJCNIS), Vol.9, No.1, pp.20-30, 2017.DOI: 10.5815/ijcnis.2017.01.03

