Int. J. Advance Soft Compu. Appl, Vol. 6, No. 1, March 2014
ISSN 2074-8523; Copyright © SCRG Publication, 2014

Big Data Preprocessing Mechanism for Analytics of Mobile Web Log

You Joung Ham, Hyung-Woo Lee

Dept. of Computer Engineering, Hanshin University,
411, Yansan-dong, Osan, Gyeonggi, Rep. of Korea
e-mail: you86400@hanmail.net, hwlee@hs.ac.kr

Abstract

The number of mobile Web service users has increased steadily as wireless mobile Web-based services have diversified, and existing services are also being provided by Web-based interfaces on smartphones. However, mobile Web users' information can be leaked to attackers as mobile Web services are interrupted by SQL injection and/or parameter injection attacks. Therefore, an efficient preprocessing method for big-data mobile Web log information is needed in order to detect such attacks on mobile Web servers by analyzing the log information created. Since the existing Web log preprocessing method implements sequential exploration of log character strings, it is not suitable to high-speed processing of mobile Web log data. In this paper, an algorithm to classify and index, by field, large-capacity mobile Web log information was developed using the B-Tree structure. Through this algorithm, Web log preprocessing and an improved search function can be provided efficiently, and the proposed method can be applied to detecting attacks on mobile Web servers for analytics of Big mobile Web Log Data .

Keywords: *Web Log, Mobile Big Data, Analytics, Preprocessing, Attack Detection, High-Speed.*

1 Introduction

Local Internet utility rates and the number of local Internet users are in a steadily increasing trend, and age groups and occupational clusters are also becoming more diversified. In addition, many business groups have changed their operational systems to Web-based services following this rapid increase in Web users and Web services. Therefore, the development of mobile devices has caused

an explosive increase in mobile Web connections. Due to such increases in mobile Internet users, monthly Web log quantities for major portal sites constitutes big data of about 50TB, more or less. Web log information can be used as a tool to analyze consumer trends. Using it, we can provide more advanced services for enterprises that use mobile Internet for major business and marketing processes. However, the Web log data collected is not in a form that can be used directly for analysis. Preprocessing is needed in order to analyze trends of users through the Web log and to obtain statistical data from Web sites. Therefore, this paper proposes a Web log preprocessing method via an index method and a B-Tree structure for Web logs in order to improve performance of Web log preprocessing.

Moreover, attempts (and successes) in mobile Web attacks have also increased, along with the quantitative increase in smartphone users [1]. The reason mobile Web services have become the focus of hacker attacks is because service methods of many businesses and business groups have changed to Web-based services, and their dependence on the Web system has also increased with the rapid increase in mobile Web applications. In addition, since mobile Web services have to allow access from outside through port 80 and port 443, the Web server can easily be attacked even though other ports are protected by a firewall.

At present, a Web intrusion detection system (IDS) has been established for Web services in order to shore up their weak points. The existing IDS system uses rule data with IP packets in order to detect attacks. However, it has to detect illegal access from outside by analyzing the Web log data created on the Web server or provide a function to detect abnormalities in order to detect attacks on the Web server [2]. The existing Web IDS system uses Web attack rule information in order to detect outside attacks, inappropriate query transmissions from inside the Web system, or abnormal access information based on the Web log. But since the existing Web IDS system applies a method that just compares the Web attack detection rule without separate preprocessing of the Web log, which is large, it is not able to efficiently cope with a Web attack implemented in real time.

In order to solve the problem, this paper proposes a method to improve the performance of the Web IDS system by implementing preprocessing to enhance the efficiency of attack detection with large-capacity mobile Web log information created by mobile Web users. The proposed method is designed to implement enhanced preprocessing based on an optimized B-Tree. When applying the proposed preprocessing method, a large quantity of Web log information can be processed at high speed, and Web service attacks can be detected and coped with more effectively than with the existing method.

This paper is arranged as follows. Related research in Section 2 looks at utility rates and the reliability of Web services, investigates the Web-based mining method and existing preprocessing methods, and examines the inefficiency of existing preprocessing methods. Section 3 explains the index method for the Web log as proposed in this paper and the preprocessing algorithm through the B-Tree

structure. The subsequent section describes the session classification function using the proposed method. The last section explains the benefits of the proposed method through analysis and evaluation of its performance.

2 Related Works

2.1 Mobile Web Attack

Smartphones, tablets, and similar smart mobile devices have been greatly successful for personal communications and computing, owing to their always-on connectivity, mobility, and usability, and to operating systems that enable a vast market of applications tailored to the needs of mobile users. Smart devices increasingly store and provide access to a range of personal, corporate, financial, and security-related Web data. While availability and ease of access to such data comes via different types of network connections (e.g. Wi-Fi, cellular), the ability to download, install and use mobile applications and the practicality of using networked services while on the go make the smart mobile device an inseparable companion in the modern world. But they also provide the perfect breeding ground for malicious software.

Mobile Web hacking incidents, such as leaking personal information of users and enterprise home page falsification, plus financial accidents, increased suddenly as the number of Web users suddenly increased. Most such incidents come from intrusion into the system through a home page that is open to anybody, instead of by someone hacking into the system.

Attack methods using the Web are becoming highly advanced, and attack frequency has also suddenly increased, such that countermeasures need to be presented. The number of Web-related attack type is more than 4000, and is uncountable if we include unknown attacks. Web systems have various forms of threat factors, such as buffer overflow, session hijacking. Directory Traversal, cross site scripting (XSS) and structured query language (SQL) injection. Typical Web attack methods include cross site scripting, SQL Injection, and denial of service (DoS) and are explained as follows.

1) Cross site scripting attack: An XSS attack transmits malignant code to the user via Web pages, Web bulletin boards, and in Web mail using characteristics that allow the codes to be implemented in the client's browser when the codes are prepared in languages such as JavaScript, VBScript, Flash, ActiveX, XML/XSD or HTML, which are implemented on the client side or are entered with the user's security ID.

2) SQL injection attack: Most current Web sites create SQL queries to access the database using values entered by the user. With a user login process, the site creates a SQL query gate for the user account and password in order to confirm whether the user entered a valid account and password. In this moment, the

attacker may interrupt normal operation by sending a falsified user name and password through the SQL injection method. The following attacks are available using such abnormal SQL queries.

- User authentication could be passed in an abnormal way.
- Data stored in the database could be read at its discretion.
- The system could be manipulated using the system order of the database.

Such weak points are called SQL injection weak points, and they may exist in numerous Web pages where the user can enter data.

3) DoS Attack: DoS (Denial of Service) attack is not an active method that acquires unlawful authority to enter a specific system, but is an attack method that selects the network and system resources as the target. DoS attacks on the Web could be detected through requesting the resources for a specific server based on the Internet Protocol (IP) address of the packet of the sender, and the frequency of error messages. However, now it is difficult to reverse-trace the source because of attacks such as IP spoofing. Spoofing can be checked through preprocessing user identifications when a DoS attack uses spoofing.

A rule-based mobile Web IDS system was presented in order to complement the weak points discussed in the previous section. However, the weak points of a rule-based Web IDS system could be analyzed as discussed in the following sections.

2.2 Mobile Web Log Big Data

Due to the proliferation of smartphones and featured mobile devices, the mobile technology landscape is evolving rapidly. Innovations in mobile technology, platforms and devices continues to grow and is bound to drive demands to extend enterprise applications and Web content to mobile devices. Mobile Web not only brings a paradigm shift in the way business applications are developed, delivered and consumed, but also changes the way businesses serve their customers. In such a competitive economic environment, it has become a business imperative to extend/enable enterprise applications and Web resources to mobile devices [11].

The development of the mobile Web has brought a direct change in consumption by consumers, as well as the ability to share information. According to an America Online (AOL) survey with Internet users of the United Kingdom in 2013, 75% of the respondents selected the AOL search site as the most significant information source when they consider purchasing products. The next significant Web site was a price comparison site selected by 56% of the respondents, while TV and newspapers remained at 34% each, and dependence on sales clerks in stores was only 24%. The Web has gained a reputation for reliability from consumers, not only in terms of quantity but also in terms of quality of information. Local consumers also show similar figures. According to a survey carried out by the Korea Internet Security Agency in 2012, 85% of candidates appeared to read product reviews, postscripts and comments of other users on the

Internet when they are going to purchase a product or service. In particular, the level of influence appeared to be highest among women (95.6%) and youth in their 20s (96.2%). Many enterprises that recognized the significance of online channels have classified users by gender, age, occupation, residential area and preferences, and have used the classifications for their target marketing strategies. Such a classification method is called Web-usage mining [3,4] after aggregating Web log data from the Internet, as seen in Fig. 1.

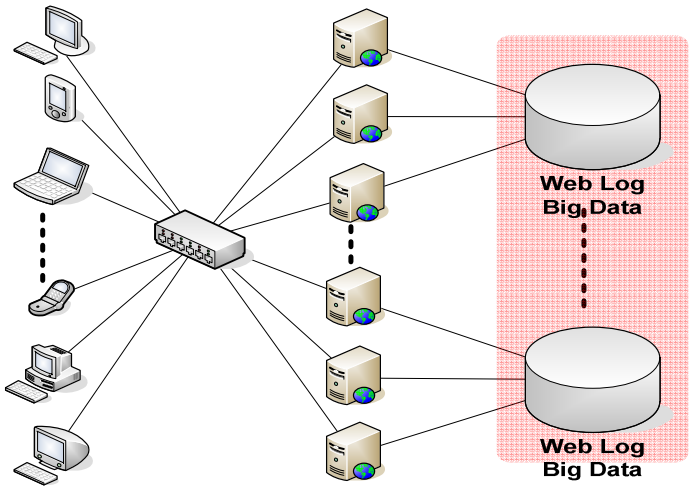

Fig. 1 Mobile Web Log Big Data.

2.3 Mobile Web IDS and its Weakness

Hypertext transfer protocol (HTTP) is one of protocols used most commonly, and one feature is that it does not require much knowledge and can be hacked with simple manipulation and interference with a universal resource locator (URL). Thus, a Web IDS system became a requirement, because the existing general IDS system was not able to actively cope with attacks specialized for Web services.

Rule-based mobile Web IDS is a method to detect attacks through comparison with defined rules by analyzing the Web log data files stored on an internal server. Therefore, the existing rule-based mobile Web IDS is a Host-IDS (HIDS), designed to detect the attack on each device of a Web server, and the rules used in these attacks are created in forms, as shown in Table 1.

Table 1: Web IDS Log Format.

Format	*Example*
Rule ID	31103
If_sid	31100
URL	|select%20|select+|insert%20|%20from%20|%20where%20|union%20|
Description	SQL Injection

Group	Attack, SQL Injection

The existing preprocessing method detected the attack through following stages and delivered the result of the detection to the administrator:

- Stage 1: Collect the logs.
- Stage 2: Compare the logs collected to the rules.
- Stage 3: Transmit the result from application of the rules to the administrator.

However, the existing Web IDS implements comparison with the rules through sequential search without preprocessing the log files. Thus, its disadvantage is (worst case scenario) that it may present the detection result after a significant amount of time has elapsed. Since the existing system implements detection of an attack based on the rules, an original Web log that is not processed does not provide satisfactory performance. Thus, this paper provides a method of efficient Web IDS performance with rapid detection of attacks through design and implementation of a Web log preprocessing algorithm to detect attacks.

If preprocessing the Web log is implemented, attacks on a Web server, such as script upload and query transmission that uses the weak points in Web services mentioned earlier could be efficiently detected. Therefore, the next section shall study the existing Web log preprocessing method and present a new Web log preprocessing method suitable to Web-attack detection.

Web logs consist of a certain format. The formats are classified mainly as common log format CLF) and extended log format (ELF). The main log format is decided by the Web log configuration fields in Table 2.

Table 2: Web Log Configuration Fields (Selected)

Log Configuration Fields	*Description*
%a	*Remote IP address*
%b	*Transmission capacity (bytes) including header*
%h	*Remote host*
%l	*Remote login ID (if supported)*
%p	*Canonical port no. of server*
%P	*Child process ID (PID)*
%r	*First request line*
%t	*Time format (CLF format)*
%u	*User name at remote location (when authenticated)*
%U	*Requested URL*
%v	*Canonical server name according to client's request*
%V	*Server name according to setting the user's canonical name*

3 Mobile Web Log Preprocessing Method

3.1 Web-Usage Mining and Extraction of Log Field Factors

Mobile Web-usage mining refers to automatically finding the access patterns of Web page users through Web log analysis. Generally, a Web log that constitutes a certain type of format is created on the Web server. The Web log created on the Internet Information Services (IIS) Server or on Apache follows the World Wide Web Consortium (W3C) format. Table 3 explains the W3C extended log format of IIS 6.0 that is used in this paper [5]. Web-usage mining of several log fields in Table 3 requires only the information of certain fields, considering the complex relations between some fields. For example, considerations for detecting abnormal actions by log fields of Table 4 was studied in research for a paper called " Extracting the Characteristics of Abnormal Actions for Detecting the Abnormality of Large Capacity Web Log."

Table 3: Extended Mobile Web Log Format

Field	*Description*	*Default*
date	*Date of activity occurrence*	*Y*
time	*Time of activity occurrence*	*Y*
c-ip	*IP address of client accessing server*	*Y*
cs-user name	*Authorized user ID accessing server (Use "-" when allowing anybody to access)*	*Y*
s-site name	*Internet site that client accessed*	*N*
s-computer name	*Computer of server that created the log*	*Y*
s-ip	*IP address of server that created the log*	*Y*
s-port	*Port of server the client accessed*	*Y*
cs-method	*Requested action (e.g; GET method)*	*Y*
cs-uri-stem	*Web server resources the user accessed (e.g:default.htm)*	*Y*
cs-uri-query	*Client's query information*	*Y*
sc-status	*Action status in HTTP, FTP*	*Y*
sc-win32-status	*Action status in Windows OS*	*N*
sc-bytes	*Bytes that server transmitted*	*N*
cs-bytes	*Bytes that server received*	*N*
time-taken	*Time the action continued (unit: millisecond)*	*N*
cs-version	*Protocol version*	*N*
cs-host	*Contents in host*	*N*
cs(User-Agent)	*Browser information of client*	*Y*
cs(Cookie)	*Cookie information*	*N*
cs(Referrer)	*Reference information of previous site*	*N*
sc-substatus	*Sub information of error code*	*Y*

As seen above, a study on extraction of log field factors shall be conducted in order to apply the Web-usage mining method to the Web log. In addition, preprocessing the Web log, including extraction of the migration route in the Web site, is essential considering the refinement and filtering process of unrelated log information, user classifications, session classifications of classified users and classified sessions. The existing Web log preprocessing is investigated in the next section.

Table 4: Mobile Web Log Field Description

Field	*Description*
IP	*Were many abnormal actions requested by one IP address?*
	Is the IP address used a normal registered one?
date_time	*Were the abnormal actions focused at a certain time?*
	Were the abnormal actions requested cyclically?
User Agent	*Are there many kinds of user authentication for one IP address?*
uri-stem	*Did abnormal actions intensively request only a specific page?*
	Is the requested page popular?
	Is there vulnerability in the requested page?
uri-query	*Is there an attack on a vulnerability in the requested query?*
sc_status	*Are abnormal actions focused on a specific error code?*
time taken	*Is there a serious deviation in time taken?*
Bytes	*Is the deviation in transmission bytes serious?*
cs (referrer)	*Are the abnormal actions requested from a normal referrer?*

3.2 Existing Web Log Preprocessing Method

The existing Web log preprocessing method consists of 5 stages, the first stage is cleaning the log, the second stage is user identification, the third stage is session identification, the fourth stage is path completion, and the fifth stage is formatting.

- Stage 1 (Cleaning Log)—it is part of preprocessing to delete log information that is not necessary for analysis. In general, multimedia files that constitute the Web pages (e.g., jpg, jpeg, gif and swf), have no relation to the user's intentions because they are automatically requested when the user requests the Web page. So this is unnecessary information for Web-usage mining [6,7]. Therefore, the information should be deleted. Such log deletions can increase efficiency by reducing the log size to 1/20th or 1/40th of its original size.

- Stage 2 (User Identification)—this stage identifies the user using Web log information. It normally identifies the user using c-ip and c (User Agent) field information. However, one problem is that it is not able to identify the different users who use the same computer, simply from c-ip and user agent information,

when a proxy server is used. Existing approaches [3,4] assumed that several users use the same computer if the page newly requested for identifying the user is not directly linked to the page visited immediately before or after the check.

- Stage 3 (Session Identification)—the session does not show whether or not the user is under login. The session ID shall be given from the time when the user accesses the Web page with the browser. Therefore, this stage is most important in preprocessing for analyzing the migration route and preferences in each user's Web page. Previous work [4] introduced the method to detect the session through the Web log. It classified the action occurring in 15-minute cycles as the same session, considering the cycle of actions by each user as classified in the user identification stage.
- Stage 4 (Path Completion)—this traces the migration route of the user through the Web pages using the classified session information and normally implements the route connection process of the migration made by pressing the Back or Forward buttons in the browser.
- Stage 5 (Formatting)—this converts the format to a form of information suitable to analyzing the Web log. The type of format is subject to the analysis method to be used.

A Web log obtained through the above stages can be useful data for the Web-mining method. However, since the data form of the Web log gets more complicated, and its size also becomes large due to recent development and changes in the Web environment, the existing preprocessing method urgently needs to be improved. Fig. 2 is the existing algorithm block diagram for mobile Web log preprocessing.

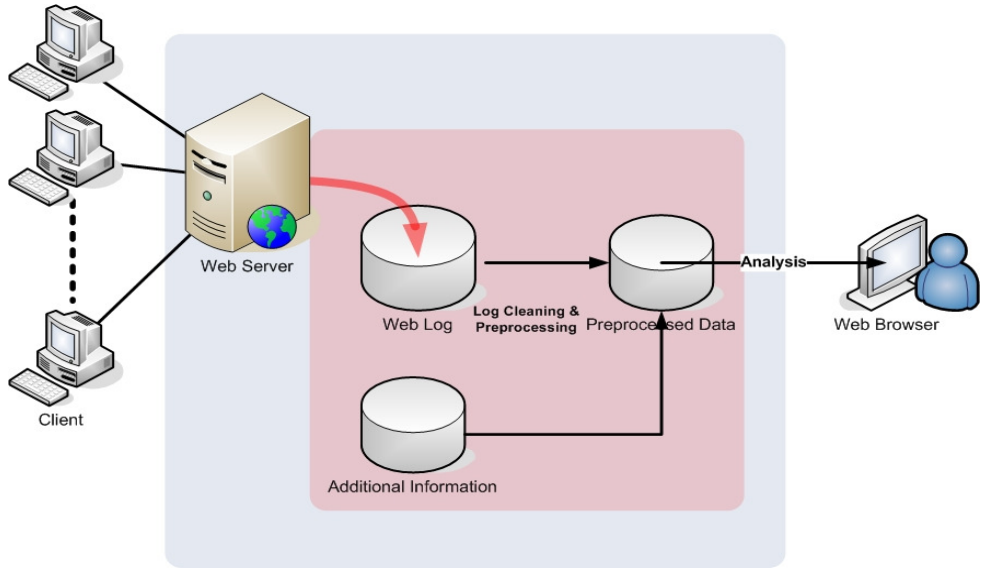

Fig. 2 Big Data Mobile Web Log Preprocessing Procedures.

3.3 Inefficiency of Existing Web Log Preprocessing Method

The Web log has become large in size due to developments on the Web. It is inefficient to preprocess such a large-capacity Web log with the existing method. The file-type suffixes in the log information have to be screened in order to implement the first stage (cleaning the log) of the existing preprocessing method. For example, all log information where the file-type suffix in the filename is .gif or .jpeg could be deleted. However, the existing preprocessing method implements the log cleaning process through sequential scanning of the whole field, as seen in Table 4. Therefore, it is more inefficient than implementing the log cleaning process by indexing the overlapped log information by extracting the *sc-uri-stem* field from Table 3, which is the method proposed in this paper.

In order to create the new information using each field's information from the Web log, we need a search function for the results of the classification method by field and specific character string. For example, log information that shows requests for specific resources in a certain cycle could be reported as a Web scan attack. The result could be detected by the search function in the results from the date and time field by extracting the *cs-uri-stem* field. However, the existing Web log preprocessing method is inefficient for mining that uses such complex log fields because there is no exploration function by field unit and search in the results. In this section, a new Web log preprocessing method is proposed in order to improve such inefficiencies in the existing preprocessing method.

The existing rule-based Web IDS detected Web attacks through exploration of sequential string exploration of access log files. However, an invalidate input attack on the Web is provided via query door requests to the server by the client. Therefore, the method proposed in this paper divides the unnecessary exploration process of each field using a divide-and-conquer method when comparing the rules, and improves the existing algorithm using the binary exploration algorithm for rule comparison.

4 Proposed Method

The high-speed Web log preprocessing algorithm proposed in this paper reduces the amount of log information innovatively using the field unit index method, considering the parsing method by field and overlapped character strings by loading the log information. We propose an algorithm that uses a divide-and-conquer strategy on the huge Web log using a multi-thread method. The preprocessing method proposed in this paper processes the overlapped strings in order to efficiently compare the rules with the existing Web log data file. The process of overlapped strings, considering the characteristics of strings, is a method to index the overlapped strings in the table configured by each field of text log files. Therefore, the log files are divided into parts, and problems are preprocessed into overlapped string units, which showed a performance

improvement of about 50%. In this case, each field of the log files is stored to index information such that it can be converted to a complete string. In addition, it improves the performance of the existing preprocessing process by configuring the field unit index information based on the B-tree structure.

4.1 Mobile Web Log Loading Module

The IIS log is a large-sized block of text information. However, the single mass of information can be divided into each line, and the lines can be divided into several fields, because each line consists of fields, and each field has relevant information. Therefore, we need to solve problems by dividing up the information, instead of directly searching such a large-sized block of information. As mentioned earlier, the method is to index the information of each field and convert it into a B-Tree that is optimized for search. It is necessary to convert the information directly to B-Tree when the Web log is loaded for analysis. Therefore, the above tasks and loading of the Web log should be done at the same time. Based on Web log data, we can get preprocessed data as seen in Fig. 3 to extract meaningful messages from the big data Web log data set.

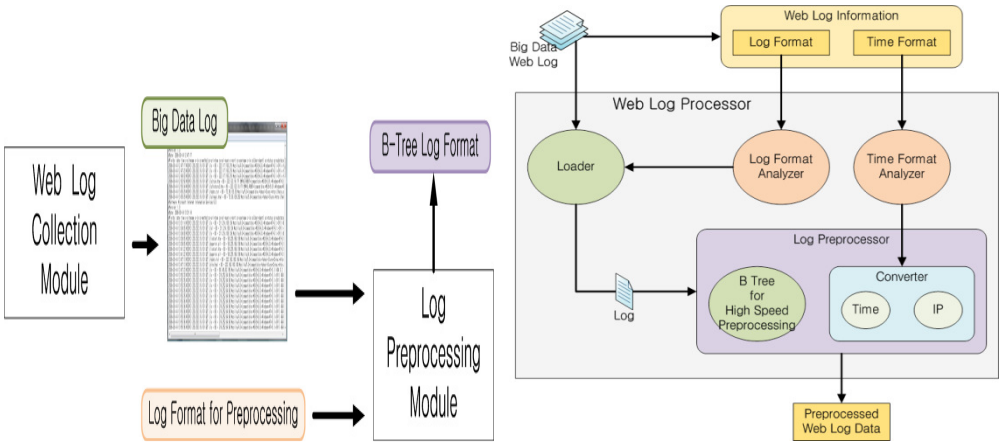

Fig. 3 High-Speed Preprocessing of Big Data Mobile Web Log.

4.2 Overlapped String Filtering Method

The filtering of overlapped strings should be executed as the load function for the Web log is implemented. The standard template library (STL) [8] map consists of keywords and data, configuring them into B-Tree through comparison of keywords. The keywords are designated as the field information, and the data are designated for an index of the field information. If the information from Key5 enters the map, the map should not accept the information, but rather the value of Second shall be set to 'False' in pair information because the data already exist in the position where the information is to be placed.

4.3 Index Processing Module for Big Data Query

Comparison of numbers is faster than that of character strings when comparing some data in a program because the program has to compare the numbers 10 times in order to compare the 10-character strings. The function of the program could be improved if the comparison is made only one time by allowing the index to accept 10-character strings. In order to do this, the character string is converted to a number array by dividing one line of the log, which is made with a single block of text, into each field unit and then providing each divided character with its own index number, configured in B-Tree and based on the overall system module shown in Fig. 4.

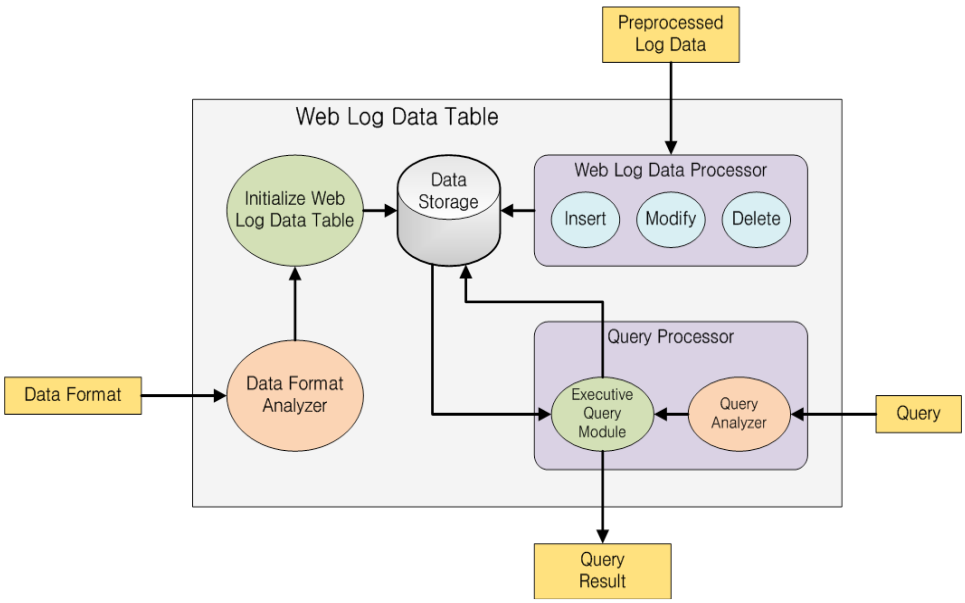

Fig. 4 Big Data Indexing and Querying.

For example, the log information below should be stored in memory as sequences of indexed numbers.

```
2013-03-14 18:40:15 W3SVC1 203.252.26.188 GET /ip - 80 -
58.145.82.195
Mozilla/4.0+(compatible;+MSIE+8.0;+Windows+NT+6.1) 404 0 2
```

1	22	1	1	1	8	0	1	16	8	4	1	2

4.4 B-Tree based Configuring Module

B-Tree uses the map that is one of the containers of the STL. The map has the keys and values and configures B-Tree using the keys, while the keys have to

have unique values. In addition, B-Tree may obtain the values by exploration, using the keys in order to access the values. The reasons the map is used are as follows: 1) The condition and value of the search are classified, and 2) the exploration and insertion can be done quickly due to configuration by B-Tree. When the condition itself is the field information and the key, the information can be searched quickly, and the index numbers allocated to each field's information can be stored in the value. In addition, the index numbers can be inserted quickly by B-Tree, which configures each field from the text information (which is large but without finalized sizes) when calling the Web log.

However, there is one problem where it is not possible to gain access with an index number. In order to solve this problem, the vector, which is a container of the STL, is also used. The vector can be referred to as a variable length array. Inserting the element in the middle may lower the function, but inserting the element in the last stage has an advantage in that fast and random access is available. Since an index number is allocated next to the previous one when new field information enters the tree, the field information is added at the end of the vector so that the field information can be entered into the index, which is the same as the field information and the same as the index number inserted into B-Tree. That is, each field consists of two structures; one is built with B-Tree in order to be optimized for search based on the field information using the map, and another has a list structure that can access the field information directly using the index number, if known. Fig. 5 shows the two structures.

The reason we store the information with two structures is to have optimal performance in both structures. When starting the search, it basically searches with field information; in this case, the information is searched in B-Tree, which is optimized for exploration and which obtains the relevant index and all works afterwards are processed with the obtained index numbers only. It is not necessary to search again in B-Tree after obtaining the index number, but may refer to the value with the index number only.

But in this case, the efficiency of the memory is lowered because the same value is stored twice. In order to maximize the efficiency of the memory, the actual field information is stored only in B-Tree, and the field information the list vector has is configured with a pointer that shows the field information stored in B-Tree.

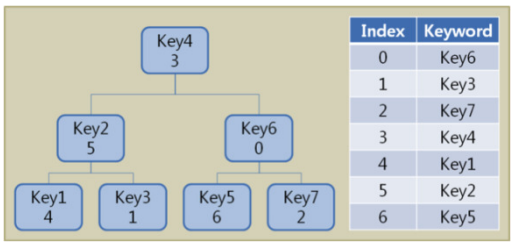

Fig. 5 B-Tree Structure for High-Speed Preprocessing.

4.5 Mobile Web Log–based Attack Detection Method

This paper proposes an algorithm that divides and conquers the text log using a multi-thread method. Divide-and-conquer means to divide a problem (of size *n*) to be solved into several smaller partial problems provided that, in this case, it is necessary to divide it so that the answer to the original problem can be obtained from answers to several smaller problems, as seen in Fig. 6.

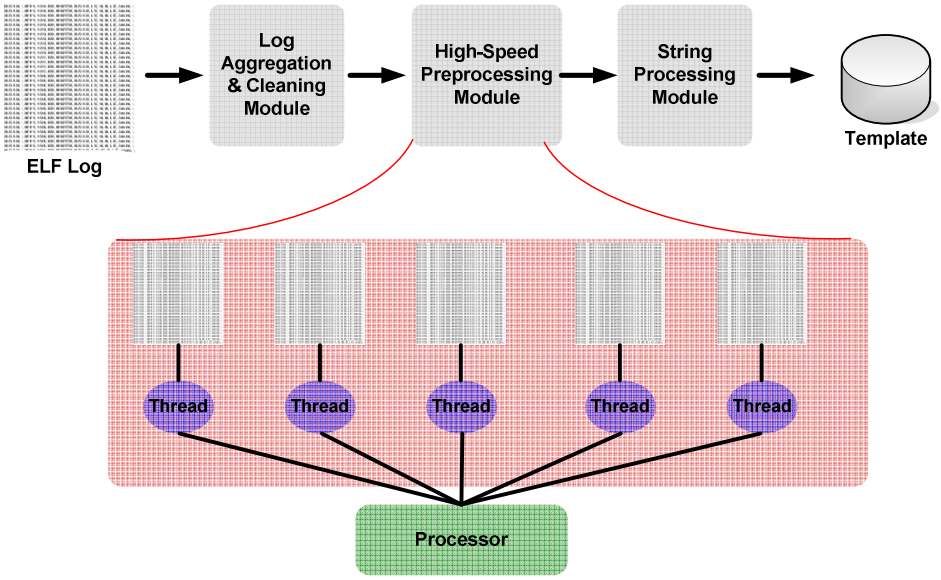

Fig. 6 Thread-based Parallel Processing for Preprocessing of Mobile Web Log.

The preprocessing method proposed in this paper processes the overlapped strings in order to compare the rules efficiently by dividing the log file. The fields of the log file that are targets to be compared in this paper include the IP address (%a) of the remote client; transmission quantity (%b), including header; the first requested line (%r); and the requested URL (%U). The log formats divided into each field are preprocessed into the threads proposed in this paper.

The process of overlapped strings considering the characteristics of strings is a method to index the overlapped strings in the table configured by each field of text log files. Therefore, the log files divided into each partial problem were preprocessed into overlapped string units, which showed a performance improvement of about 50%. In this case, each field's information from the log files is stored as index information such that it can be converted to a complete string. Fig. 7 is the overall structure for session classification and correlation analysis on the big data Web log in order to detect attacks.

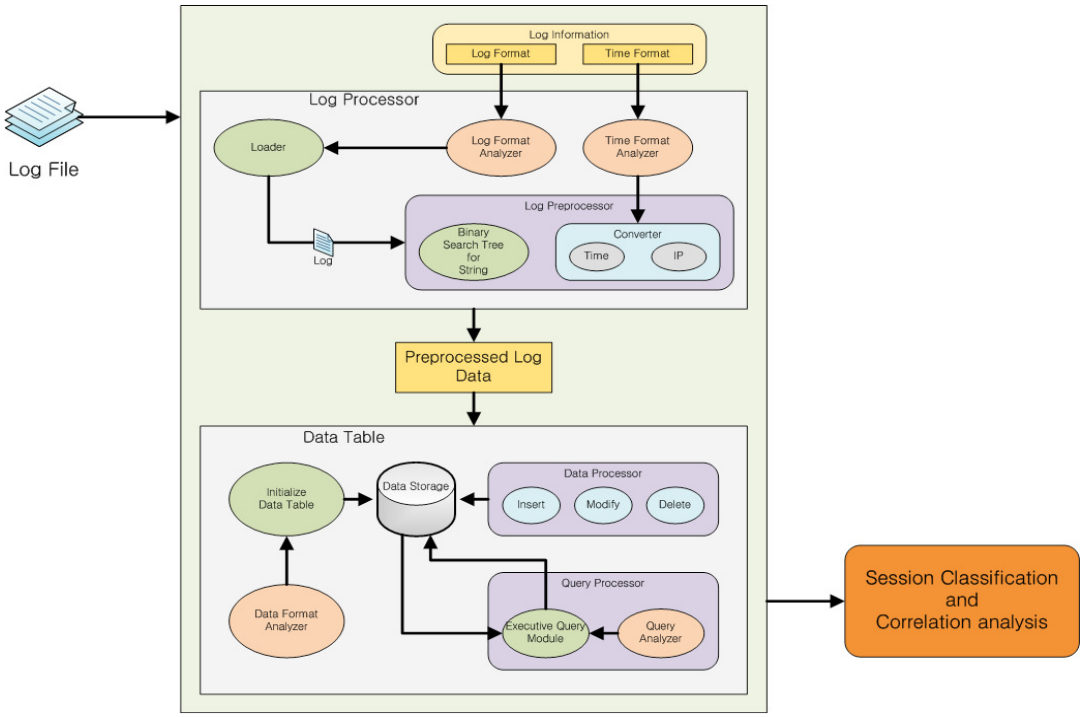

Fig. 7 Big Data Mobile Web Log Preprocessing for Detecting Attacks.

The existing rule-based Web IDS detected Web attacks through sequential string exploration of the access log files. However, an invalidated input attack on the Web is provided via query door request to the server by the client. Therefore, the method proposed in this paper divides the unnecessary exploration process into each field using a divide-and-conquer method when comparing the rules, and improves the existing algorithm using the binary exploration algorithm for rule comparison.

Since the strings that are targets of actual comparison are aligned in the exploration method for rule comparison proposed in this paper, the comparison with actual rules shall be made by exploration of the binary tree. SQL injection and parameter injection attacks can be detected through comparison of URLs requested by the client. Therefore, the process handles the strings of the URLs out of the existing rules as the target to compare.

Each field's information obtained by the divide-and-conquer method proposed in this paper is compared with rules using the binary exploration method. However, the administrator will not be able to obtain the exact information for the log under attack if only the divided information is included when transmitting to the administrator a detection result showing an attack. Therefore, the text log is included when transmitting the detection result to the administrator.

5 Performance Evaluation

The experiment for this research starts with the collection of the Web log. The IIS log used in this research was the home page log for computer-related departments at local universities recently collected over one year. The Web log preprocessing algorithm of this paper focused on a reporting function for the future instead of real-time analysis. The method proposed in this paper divided the large-size log file by fields with high speed using multi-threads. Table 5 shows the comparison results of performance between the existing preprocessing method and the proposed preprocessing method.

Table 5: Performance Evaluation

Preprocessing / *Items*	*Existing Method*	*Proposed Method*
Detection of Attack	x	○
Log Analysis and Division	x	○
Use of Multi-Threads	x	○
Log Cleaning	○	x
Creation of Template	△	○
For Mobile Web Log	x	○
High-Speed Preprocessing	△	○

The experiment for this proposed algorithm carried out the comparison test of string search performance according to the log size through 8 different log sizes, from 500,000 log lines to 4,000,000 log lines. In addition, the reference field during searches was set to search all fields. The target program for process performance comparison assessment of this algorithm is Log Parser [9]. This is a search program using text-based query provided by Microsoft and is generally used for analyzing the Web log.

Table 6: Performance Evaluation

Method / *Size*	*Text Log Search*	*Proposed Method*	*Log Parser*
500,000	7.329	1.543	7.880
1,000,000	16.550	5.062	17.470
1,500,000	28.269	6.105	28.610
2,000,000	39.201	7.509	41.120
2,500,000	52.720	9.249	48.200
3,000,000	61.483	10.031	62.830
3,500,000	81.348	12.852	83.234
4,000,000	96.342	16.345	98.885

Size – Number of lines and preprocessing time (sec.)

According to the results in Table 6, the Web log search time of the proposed model is in proportion to Web log size and is not so different from existing models in comparison with text log load. However, the results of tests with the preprocessed log file (PLF)-type log load converted after preprocessing by the proposed method in this paper shows an innovatively improved search performance. PLF format is a preprocessed template for text logs. Some might find that a template converted to PLF shows relatively little difference in search time according to log size because of application of the index method considering the overlapped strings and the B-Tree structure method. In addition, the proposed algorithm may have the effect of reducing the log file size by 1/3 through preprocessing the text log. Indexing the overlapped strings could have such result.

This experiment improved search performance of the Web log and the efficiency of the physical space by developing the high-speed preprocessing algorithm for the Web log file through the index method and B-Tree configuration. Accordingly, it is possible for us to enhance the performance of SQL injection attack detection compared with the existing method [10] by using the proposed Web log session classification and verification of values entered by users.

6 Conclusions

Web services have developed rapidly due to a sudden increase in Internet utilization. However, development for the security area is at a snail's pace. The seriousness for Web security is sufficiently recognized due to suddenly increased hacking attacks on Web services. At present, there exist many systems for Web security, but the network-based IDS/IPS among them is not a Web attack detection system suitable to a Web service environment because it is not possible to recognize the attack through analysis of request and reply logs between server and user due to the characteristics of Web services.

Thus, this paper improved efficiency of rule comparison through indexed mobile Web logs and preprocessing of overlapped strings using multi-threads in order to enhance detection efficiency of attacks on mobile Web IDS systems. In detail, the algorithm proposed in this paper is optimized for improvement of mobile Web log preprocessing performance. Since it strengthens search performance to detect specific strings in mobile Web logs, the expected effects and utilizations include improvement of performance when detecting the attack, identifying the user and session, and discriminating by IP address. Therefore, improvement of classification performance is expected from applying the proposed algorithm to pattern classification of large-capacity mobile Web logs and the data-usage mining method. Additionally, this paper improved the efficiency of rule comparison through Web log division and preprocessing of overlapped strings using multi-threads in order to enhance attack-detection efficiency on mobile Web IDS systems.

ACKNOWLEDGEMENTS.
This work was partially supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (Grant # 2012R1A1A2004573).

References

[1] Santosh Shakya, Anju Singh, Divakar Singh, "A Time Efficient Algorithm for Web Log Anaysis," *International Journal of Computer Applications*, Vol. 75, No. 9, pp.23-30, 2013

[2] Slva S. Slvatha Sindhu, S. Geetha, A. Kannan, "Decision tree based light weight intrusion detection using a wrapper approach," *International Journal of Expert System with Applications,* Vol. 39, Issue 1, pp.129-141, 2012

[3] Aye, T.T, "Web log cleaning for mining of web usage patterns," *3rd International Conference on Computer Research and Development (ICCRD),* Vol.2, pp.490-494, 2011

[4] Yuankang Fang, Zhiqiu Huang, "An Improved Algorithm for Session Identification on Web Log," *WISM 2010*, LNCS 6318, pp.53-60, 2010

[5] http://www.loganalyzer.net/log-analyzer/ w3c-extended.html

[6] Priyanka Patil, Ujwala Patil, "Preprocessing of web server log file for web mining," *World Journal of Science and Technology*, Vol. 2, No. 3, 2012

[7] Sheetal A. Raiyani, Shailendra Jain, "Efficient Preprocessing technique using Web log mining," *International Journal of Advancements in Research and Technology,* Vol.1, No. 6, pp.59-63, 2012

[8] Standard Template Library Programmer's Guide http://www.sgi.com/tech/stl/

[9] Download details Log Parser 2.2 http://www.microsoft.com/downloads/details.aspx?FamilyID=890cd06b-abf8-4c25-91b2-f8d975cf8c07&displaylang=en

[10] Daniel B. Cid, Log Analysis for Intrusion Detection, http://www.infosecwriters.com/text_resources/pdf/Log_Analysis_DCid.pdf

[11] Jitendra Maan, "Mobile Web – Strategy for Enterprise Success," *International Journal on Web Service Computing (IJWSC),* Vol.3, No.1, pp.45-53, 2013