2009 Ninth International Conference on Quality Software

ADAM : Web Anomaly Detection Assistant based on Feature Matrix

Sungdeok Cha*, Junsup Lee†, Sangrok Kim‡ and Sanghyun Cho§
*Department of Computer Science and Engineering
Korea University, Seoul, South Korea
Email: scha@korea.ac.kr
†Knowledge e-learning research team
Electronics and Telecommunications Research Institute(ETRI), Daejeon, South Korea
Email: junsup@etri.re.kr
‡Financial Supervisory Service
Seoul, South Korea
Email: srkim@dependable.kaist.ac.kr
§NHN Corporation
Gyeonggi-do, South Korea
Email: shcho@dependable.kaist.ac.kr

***Abstract*—Importance of web security cannot be overemphasized in the era of web-based economy. Although anomaly detection has long been considered a promising alternative to signature-based misuse detection technique, most studies to date used either small scale or artificially generated attack data. In this paper, based on security analysis applied on anonymous www.microsoft.com log of about 250GB, we propose Anomaly Feature Matrix (AFM) as an effective framework to characterize anomalies. Feature selection of AFM is based on the characteristics of well-known (e.g., DDoS) attacks as well as patterns of anomalous logs found in the Microsoft data. Independent security analysis performed on the same data by Microsoft security engineers concluded that 1) We did not miss any major attacks; and 2) AFM is a general enough framework to characterize likely web attacks. In order to assist AFM-based anomaly analysis in large organizations, we implemented an interactive and visual analysis tool named ADAM (Anomaly Detection Assistant based on feature Matrix). Integrated with mapping software such as Virtual Earth, ADAM enables efficient and focused security analysis on web logs.**

***Keywords*-anomaly detection; web security; web data mining; network security;**

I. Introduction

In the era of web-based economy, the importance of web security cannot be overemphasized. While signature-based misuse detection technique is mature and highly effective in defeating known attacks, it is practically useless when new attacks are launched. Anomaly detection has long been considered a promising and complementary approach to traditional misuse detection. Although there have been several experiments on anomaly detection, most studies have common limitations. As ”live data” containing real attacks are difficult to obtain, researchers often launch attacks themselves using known exploit code and measure effectiveness of their ideas. Such artificially generated and known attacks are apparently unrealistic in evaluating effectiveness of anomaly detection techniques.

In this paper, we report results of an experiment in which we analyzed anonymous, raw, and unlabeled web log collected at www.microsoft.com. Microsoft Corporation has generously provided our research group web log containing more than one billion requests whose size is about 250GB. In order to preserve privacy, web logs have been made anonymous. That is, source IPs have been randomly hashed while ”essential characteristics” have been preserved.

Our group’s research goals are to develop: (1) Approaches to identify anomalous web requests which need to be subject to further security analysis; and (2) Effective visualization scheme of anomalies so that engineers can effectively manage a large number of web servers securely. To accomplish the research objectives, we spent nearly eight man-months manually reviewing more than 127 million entries for anomalies while performing automated analysis on the rest. Automated analysis such as applying Snort rules has also been performed.

Anomaly Feature Matrix (AFM) is an effective framework to characterize anomalous patterns we found. Analysis results have been briefed to Microsoft’s security engineers, and independent analysis had been performed on the same data. Feedback has been highly encouraging. They found no major anomalies we missed, and AFM is considered a general enough framework to characterize likely web attacks. Encouraged by positive feedback, we developed a software tool, ADAM (Anomaly Detection Assistant based on feature Matrix), to automate analysis.

Our paper is organized as follows: Section 2 reviews related work. In section 3, we describe some of essential characteristics of web log used in the study. In section 4, we introduce AFM and discuss some of the anomalous web requests. Section 5 explains capabilities and interface im-

1550-6002/09 $26.00 © 2009 IEEE
DOI 10.1109/QSIC.2009.24

plemented in ADAM. It is integrated with mapping software such as Virtual Earth so that security engineers may perform focused security analysis. Section 6 concludes the paper and describes the work in progress.

A. Related Work

There have been several attempts to detect anomalous and suspicious activities in web requests. Snort[**?**], widely used open-source tool, has more than 1,300 signatures on known attacks stored in more than 50 rule sets. However, not all the rules remain effective in typical web server configurations. When applied to IIS web servers, for example, only about 11% of the rules are applicable. More importantly, unless web attacks are analyzed and attack patterns coded as rules, Snort is essentially useless in defending servers from the threat of unknown attacks.

Kruegel[**?**] investigated how anomaly detection technique can be applied on web logs. For example, Kruegel developed anomaly detection models based on features such as attribute length, character distribution, or absence of attribute variables and their sequences. While various features can be assigned different weights to optimize performance, it is difficult to determine the ”right” parameter values. As operation heavily relies on heuristics, simple computation (e.g., average of anomaly scores) may not detect sophisticated attacks.

Session anomaly detection (SAD), developed by Cho et al. [**?**], divides a sequence of web requests into sessions. Session characteristics (e.g., page sequences) are compared against those of previous sessions initiated by the same IP, and anomaly score is computed based on assumption sessions would exhibit similar patterns. While probably true in static IP environment, such assumption may not hold in environments such as Web Proxy or Network Address Translation (NAT) is used. In addition, accurate identification of web sessions may prove difficult in some environments.

Web server log visualization serves several purposes. For example, statistical analysis may assist important business decisions. Proper visualization is also critical to security analysis because textual analysis is apparently impractical in corporate environments. For example, Bayesvis [**?**], shown in Figure 1, is a naive bayesian classifier applied to http requests. As it is essentially a color coding scheme applied on web log fields in text format, Bayesvis would be usable only in small organizations. Security administrators in large and complex corporate environment need visualization schemes that offer much higher level of abstraction that Bayesvis does.

II. WWW.MICROSOFT.COM WEB LOG DATA

log used in the study has been collected at Microsoft's web servers for a day. To ensure that there is no privacy violation, Microsoft hashed all the IPs randomly while preserving essential characteristics of its http traffic (e.g., frequency, query contents, or user agents). According to security experts at Microsoft, there are about 5,000 web attacks being attempted at its web servers every day. Microsoft's web servers are known to be the second most frequently attacked web site in the world followed by that of United State's Department of Defense.

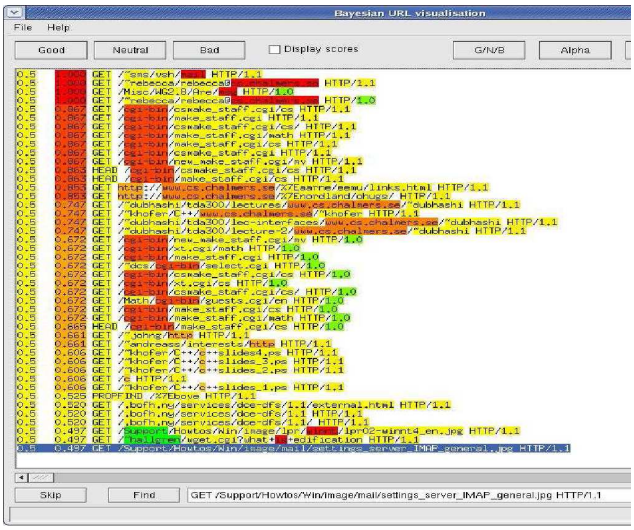

Figure 1. Bayesvis interface

Depending on the amount of web traffic, logs generated every day vary from 50GB to 1TB with 250GB being the average. Number of web log entries often exceeds one billion, and no published study on web anomaly detection used such a large scale and real-world data collected from sites that become the target of frequent attacks. Authors have not been briefed on technical details such as web server topology or page redirection policies, and data were unlabeled. That is, entries had not been marked either normal or anomalous. Preliminary analysis revealed that there are two data centers, named B and T, and each center has five clusters each of which contains eight server farms. Therefore, web log used in this study came from 80 servers.

Web log, stored in IIS format defined by W3C, contains the following fields:

- date-time
- s-computer name (server)
- cs-method (e.g., HEAD, GET, etc)
- cs-uri-query
- c-ip (source IP)
- cs-user-agent (e.g., Browser used by client)
- cs-referrer (site last visited by client)
- cs-host
- sc-status (e.g., error code), sc-substatus, sc-win32-status
- sc-bytes (e.g., number of bytes server sent to client), cs-bytes
- time-taken (e.g., how long it took to serve the request).

Table I
EXAMPLE OF DIRECTORY TRAVERSAL ATTACKS

```
\%u002E\%u002E/aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa/../\%u002E\
%u002E\\%u002E\%u002E\\\%u002E\%u002E\\\\%u002E\
%u002E///\%u002E\%u002E/\%u002E\%u002E/\%u002E\%u002E/
\%u002E\%u002E/\%u002E\%u002E/
\%u0045TC/P\%u0041SSWD emph(../ETC/PASSWD)
```

As it is simply impractical to perform manual security analysis on more than 1 billion logs, authors spent nearly 8 man-months (e.g., nearly two months of full-time analysis by four of the authors) to examine logs collected by B0 cluster for representative samples of anomalous logs. When necessary, automated analysis had been applied to the entire log data to derive statistically meaningful data. Logs collected from various servers were evenly distributed. For example, cluster B0 contained more than 127 million entries, and each server collected 18 to 21 million logs. They also exhibited similarity in that 1) The most frequently requested page, default.aspx, had roughly same number of hits; and 2) Similar number of pages had been redirected (e.g., requests to redir.dll page).

III. ANOMALY FEATURE MATRIX (AFM)

When analyzing web logs for anomaly, it is unnecessary to analyze all the fields. For example, s-computername field is apparently irrelevant. Likewise, cs-method or cs-substatus fields need not be subject to anomaly analysis. On the other hand, source IP is an important factor which must be analyzed from proper perspectives. Possible features include degree of concentration, frequency, interval, and validity. For example, frequency analysis on IP can easily detect attempts to launch brute-force DDoS attacks. Validity of IP (e.g., assigned or reserved) is another useful indicator for anomaly. If requests come from supposedly unassigned IP address, they surely deserve attention of security engineers. However, interval analysis of IP values would not reveal any meaningful insights. When applied to the time field and a sequence of requests coming from the same IP, interval analysis can determine if requests are being made periodically. Similarly, validity analysis applied on the query reveals the following buffer overflow and directory traversal (e.g., ../etc/passwd) attacks.

Many of the programmed attacks require comprehensive analysis of all the logs from the same IP to get insights into hostile intent. Precise description of anomalous scenarios often takes a combination of various features. Anomaly feature matrix succinctly represents which fields need to be analyzed from which perspectives. There are various combinations of eight fields (e.g., IP, time, user agent, etc) and four characteristics (e.g., degree of concentration, frequency, interval, or validity) that best characterize anomalies we found while investigating web logs. They are:

Table II
COMBINATIONS OF FIELDS AND CHARACTERISTICS THAT ARE USED IN AFM

Field	**Characteristic**	**Likely anomalies**
IP	Frequency	Brute-force DoS attack?
	Validity	Requests coming from unassigned or reserved IPs are really strange
Time	Degree of Concentration	How about distribution of requests? Many requests flooding in during short period of time?
	Interval	Requests came in periodically? Scanner or web robot in action?
User Agent	Degree of Concentration	How many distinct user agents (browsers) came from an IP? Requests came from browser, messenger, or web robot?
Page	Degree of Concentration	How about preference on pages? Limited to a few pages? Traces of web robots found?
	Frequency	How many times was this page requested by other users? In other words, how popular was the page? Programmed attacks or page injection?
	Validity	Requests made to existing page? Access attempts to known vulnerable resources (e.g.,/_vti_bin/shtml.exe/_vti_rpc)?
Query	Validity	Does the query contain "suspicious" character sequence? Buffer overflow attacks? Directory traversals? XSS?
Status	Degree of Concentration	What was the distribution of status code? Too many error codes would indicate some sort of anomaly
Time Taken	Degree of Concentration	How long did it take to serve the request? Are there requests that took unusually long or short time compared to others? SQL-injection attack taking place?
Bytes	Degree of Concentration	"Odd" (e.g., outside reasonable variations) number of bytes transferred between the clients and servers? Scanning attack?

Elements included in the feature matrix allow investigation from diverse (e.g., user-, content-, and page-based) analysis. For example, when analyzing frequency of IP or interval between the successive requests, each IP (e.g., user) must be reviewed in isolation. When analyzing validity of user agent or query, conclusions can be derived based on the content itself without having to compare against past requests. Likewise, some security analysis must be applied on each HTML page. For example, anomaly on time taken to serve requests and number of bytes transferred is meaningful only when values are analyzed in the context average value associated with the page.

Effectiveness of AFM as a general framework to char-

Fields / Characteristics	User Attributes			Request Attributes				
	IP	Time	User Agent	Page	Query	Status	Time Taken	Byte
Degree of concentration		▲	▲	▲		▲	□	□
Frequency	▲			▲				
Interval		▲						
Validity	○		○	○	○			

Figure 2. Anomaly Feature Matrix (▲ : user-based, ◯ : content-based, and □ : page-based analysis)

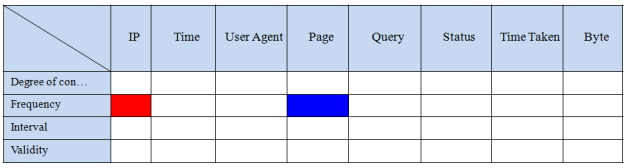

Figure 3. Frequent visits(red marked) to unpopular pages(blue marked)

acterize web attacks becomes apparent only when several anomaly feature elements are combined, and the following are some of the interesting anomalies we found out from www.microsoft.com web log.

Frequent visits to unpopular pages : While it is nothing unusual if a user requests some pages that other users are rarely interested in, it is apparently anomalous if all or most of the requests are exclusively directed to "unpopular pages." In order to detect such anomalies, one must compare average popularity index of pages. In AFM, the following features are combined to detect anomaly.

For each IP, we compute the popularity index of the pages as follows. Suppose that there are 1,000 requests overall and that requests to pages A, B, and C are 4, 5, and 10, respectively. Each page's popularity index is 0.004, 0.005, and 0.01. If pages A and B are requested twice and three times, respectively, popularity score would become 0.0046 (e.g., (0.004*2+0.005*3)/5). If a user exhibits anomalous behavior by requesting only unpopular pages, anomaly score would become low enough to attract attention of security administrators. In our data, among 127 million entries in B0 cluster, we found an IP which made 21,050 requests, and all requests were directed to the least popular page in our data. Further analysis revealed an interesting pattern shown in table 2. Requested pages were /exchange/dbeatty1/Contacts/Emily+Laurie-#.EML, where # was filled with numbers whose exact meanings are unknown to us. It appears that requests were generated by web robots.

Excessive number of distinct user agents : Because typical users run only browser they are most familiar with, most IPs are expected to have one or a small number of user agent values. In our data, the average number of distinct user agents per IP was about two. Exceptions can certainly occur

Table III
EXAMPLE OF SUSPICIOUS ACCESS TO UNPOPULAR PAGES

2006-05-30 19:11:14.000 SERVERB02 HEAD /exchange/dbeatty1/Contacts/Emily+Laurie-41756.EML 84.78.23.216 CFNetwork/1.1 www.microsoft.com 302 0 3 352 160 109

2006-05-31 01:11:03.000 SERVERB02 HEAD /exchange/dbeatty1/Contacts/Emily+Laurie-135269.EML 84.78.23.216 CFNetwork/1.1 www.microsoft.com 302 0 3 352 161 109

2006-05-31 06:28:48.000 SERVERB06 HEAD /exchange/dbeatty1/Contacts/Emily+Laurie-54189.EML 84.78.23.216 CFNetwork/1.1 www.microsoft.com 302 0 3 352 160 109
(Note: IP value was randomly hashed to guarantee anonymity.)

if an IP is used as a proxy server or NAT. However, we found an IP which had 17,875 distinct user agents scattered in more than 3.5 million requests made in 24 hours. Further analysis revealed that (1) web robots generated more than 58,000 of the requests (e.g., between 4,000 and 7,200 requests per hour), and (2) user agent field contained values referring to both Internet Explorer as well as web robot at the same time. That is, the following user agent fields were uses:

"Mozilla/4.0+(compatible;+MSIE+6.0;+Windows+NT; +MS+Search+4.0+Robot)"
or
"Mozilla/4.0+(compatible;+MSIE+5.0;+Windows+98;+DigExt)"

While errors in proxy server setting may have caused such traffic, they appear highly anomalous.

Excessive number of distinct user agents : Interactive users are extremely unlikely to request the same pages repeatedly when they encounter errors. "Smart" web robots are unlikely to exhibit such behavior, either. However, in our 250GB data, there were more than 20,000 distinct IPs whose entire requested resulted in error codes (i.e., 404) being returned. In particular, one IP made more than 7,600 requests to the following page during about 4 hour period.

"2006-05-31 00:36:48.000 SERVERB03 GET /rads/scripts/RADS.dll QueryProduct2 89.24.81.222 McAfeeInstantUpdater/1.0 www.microsoft.com 404 0 64 0 178 421"

IV. ANOMALY DETECTION ASSISTANT BASED ON FEATURE MATRIX (ADAM)

ADAM is a software tool we implemented to assist security engineers detect web anomalies early. That is, ADAM does not make definitive conclusion on the occurrence of web attacks, and other security tools (e.g., signature-based intrusion detection systems or firewalls) are expected to block apparent and known attacks. ADAM's user interface, shown in figure 1, is designed to be intuitive while remaining faithful to the approach described in this paper.

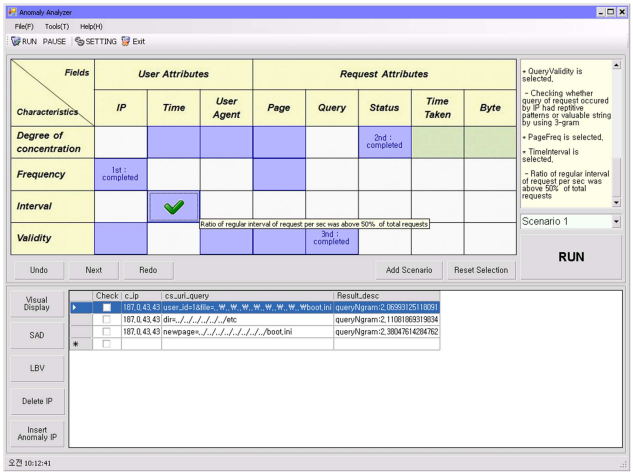

Figure 4. ADAM interface

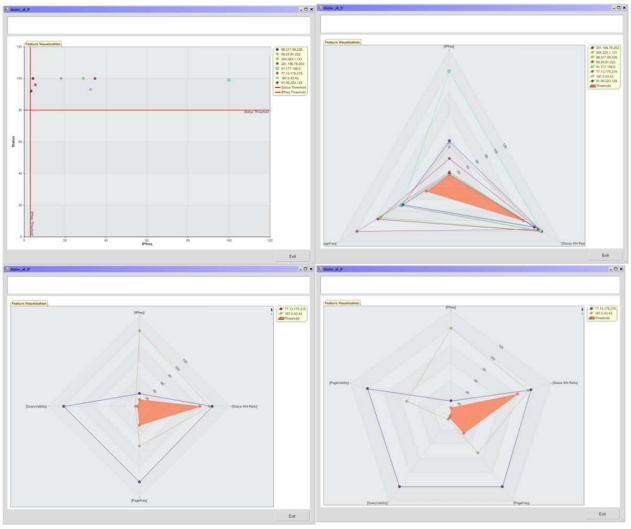

Figure 5. Visualization of anomalous requests with respect to the threshold values

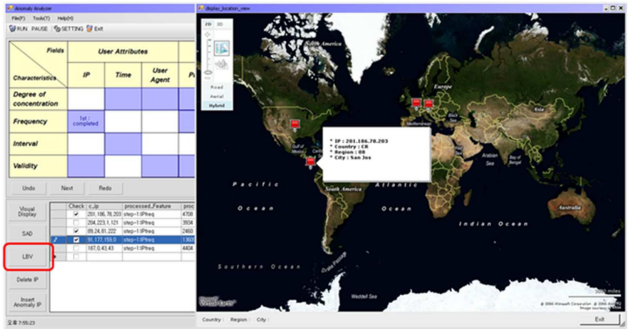

Figure 6. Geographical display of anomalous requests (Integrated with Microsoft's Virtual Earth)

Web logs first need to be parsed and entered into a database. Users then may choose to investigate if any of the predefined anomalous scenarios we found in Microsoft data are present. In doing so, users may choose the same setting we found to be satisfactory in Microsoft data. Such threshold values have been determined to minimize the amount of web logs that are subject to in-depth security analysis. Therefore, there are no universally applicable rules as to what the optimal threshold values are. Each organization may start with default values associated with the tool and gradually customize them while accumulating experience. Upper right corner of the display is reserved to inform users of various settings and analysis in progress.

There are several features of ADAM to assist security administrators perform analysis efficiently.

- Backward and forward steps in analysis (e.g, undo and redo) allow customized anomaly analysis. Each criteria further narrows down candidates, and the order of applying anomaly features is important because results would become different if same features were applied in different order. Users can always register customized anomaly scenarios.
- Lower half of the window always displays the logs that fit the pattern specified so far, and users may request visual displays on selected requests at any time. In figure 3, emphasis is on illustrating why selected requests are considered anomalous with respect to the threshold values. Depending on the number of anomaly features applied, different graphs will be generated. Lines or shaded regions represent the threshold values used in anomaly decision. Therefore, nodes further away from the center are considered more anomalous. If desired, each line or node can be selected to display the corresponding requests in tabular format. ADAM is also integrated with mapping software (e.g., Microsoft's Virtual Earth) to display where potentially hostile requests are coming from.

Preliminary experience with ADAM has been highly positive in that users found the interface self-explanatory. Unfortunately, possibility of providing real-time or near-real-time anomaly detection using the proposed approach appears remote because it takes too much time to parse web logs, upload to relational database, and visualize the query results. However, as indicated in the paper's title, ADAM is effective as an assistant in guiding security engineers to 1) select a small amount of web logs that deserve to be investigated more thoroughly, 2) issue an arbitrary sequence queries using intuitive and graphical interface, and 3) visualize results so that degree of anomaly as well as the location where the requests are coming from are displayed.

V. Conclusion

Although theory on anomaly detection is nothing new, it is often ignored in practice due to 1) relatively high degree of false alarms; and 2) lack of proper tool support. Most

anomaly detection study on web logs had limitation in that experiments were conducted using "artificially generated" logs containing traces of "known attacks." Our study makes the following important contributions to the research community.

- We used anonymous 250GB web log data collected at www.microsoft.com. As the second most frequently attacked site in the world, Microsoft's web log is a rich source of real anomalies. Based on manual security analysis conducted on more than 127 million entries, we have identified 10 anomalous patterns, and an independent security analysis by Microsoft's engineers confirmed quality and validity of our analysis.
- We proposed Anomaly Feature Matrix (AFM) as a framework that is effective and general enough to characterize anomalies that may arise in the future. AFM promotes careful, balanced, and coordinated analysis of several aspects of data (e.g., degree of concentration, frequency, interval, and validity) on relevant fields.
- ADAM is a useful tool to automate AFM-based anomaly analysis, and it provides powerful visual display capability. When analyzing a large number of web requests, visualization is essential in effectively combating sheer complexity and volume associated with data.

Subsequent to research on anomaly feature matrix, we have conducted an empirical study on how to effectively identify various web robots. Other research activities in progress include composite attribute vector (CAV) in which we use vector-based display on how each session's behavior is similar to that web robots or interactive (and assumed normal) users.

ACKNOWLEDGMENT

This work was partially supported by Defense Acquisition Program Administration and Agency for Defense Development under the contract and also partially supported by Korea SW Industry Promotion Agency (KIPA) under the program of Software Engineering Technologies Development and Experts Education. This research was supported by the MKE (Ministry of Knowledge Economy), Korea, under the ITRC (Information Technology Research Centre) support programme supervised by the IITA (Institute of Information Technology Advancement) (IITA-2008-(C1090-0801-0032)). This work was supported by the IT R&D program of MKE/MCST/IITA, [2008-S-025-02, Development of Elemental Technology for Promoting Digital Textbook and u-Learning]. This research was also supported by a grant from Korea University.

REFERENCES

[1] Snort, http://www.snort.org/

[2] C. Kruegel, G. Virgna, W. Robertson, *A multi-model approach to the detection of web-based attacks*, Computer Networks: vol. 48, no. 5, pp. 717-738, 2005

[3] Sanghyun Cho, Sungdeok Cha, *SAD : Web Session anomaly detection based on parameter estimation*, Computer & Security, pp. 312-319, 2004

[4] Stefan Axelsson. *Combining a bayesian classifier with visualisation: Understanding the IDS*, Proceedings of the 2004 ACM workshop on Visualization and data mining for computer security, pages 99-108, 2004

[5] IIS W3C Extended Log Format, http://www.loganalyzer.net/log-analyzer/w3c-extended.html