International Journal of Science and Engineering Vol.4 No.4(2014):25-31

An Efficient Web Log Filter Scheme for Web Abnormal Behaviors

Chinyang Henry Tseng[1]
National Taipei University
tsengcyt@mail.ntpu.edu.tw[1]

Shin-Yun Chang[2]
National Taipei University
s710183136@webmail.ntpu.edu.tw[2]

Tong-Ying Juang[3]
National Taipei University
juang@mail.ntpu.edu.tw[3]

摘要

隨著科技的快速發展，網路相關服務愈發複雜與趨向多變化。為了更好地保護這些網路服務的安全與隱私，主動進行異常檢查與分析的動作就變得相當重要。故，因應日常安全檢查為事後主動追蹤的需求，我們提出一種針對 Web 異常行為的 Web Log 篩選機制，結合挖掘已知攻擊行為與未知異常行為兩種層面的資料，並藉這些資料之互補性，達成較快且不失過多準確度的目的。此機制首先利用 PHPIDS 的特徵規則進行正則表示式比對，再進行資料預處理挑出可疑的 Log 及轉化 HTTP 請求欄位形成特徵矩陣，使用隨機投影進行降階，並利用馬氏距離找出異常者且為其計算異常分數。若找出的該筆 Log 太過相異將被標記為異常者，直至全數 Log 檢測完為止。為了有更切合的結果，此機制利用了現實公司的真實資料測試，並實驗出適切之參數。除此之外，此機制尚有簡單容易實現、快速且不失過多準確度、無需清理訓練資料的優點。

關鍵詞：Web Log 分析, Web Log 篩選, Web Log

Abstract

With the rapid development of technology, network services are becoming more complex and changeful. To protect the security and privacy of these network services, check and analyze abnormal behaviors actively becomes very important. In order to meet the need of check and analyze abnormal behaviors actively for routine security check, we try to find the data of known attacks and anomaly behaviors, propose a web log filter scheme for web abnormal behaviors which aims to quickly anomaly detection and also provides accuracy. The scheme uses the signature rules of PHPIDS to match, preprocesses network logs to find suspicious logs and form a feature matrix, reduces the dimensionality of matrix using random projection, uses Mahalanobis distance to identify outliers and calculate an anomaly score of the outliers. If the log line is too different, we flag it anomaly, until all of the logs are checked. In order to get the better outcome, we use the data of real-world company to test the scheme and find the suitable parameter. In addition, the advantage of the scheme is simple to implement easily, fast and without losing too much accuracy and does not need to clean training data.

Keywords: *Web Log Analysis, Web Log Filter, Web Log.*

1 Introduction

With the rapid development of technology, over the past few years, the Internet is becoming an emerging important tool for humans. However, because of the properties of Internet: transfer quickly, unknown user, difficult to trace actively, the malicious unknown user who abuses the network services is becoming numerous. When the police know the accident which is about web abnormal behaviors, they may have the less evidence to proof the attack behaviors of the unknown user and they could not trace the user actively.

In order to protect the privacy of people, the police propose a plan that they daily check and analysis the logs from the Internet companies. Although they spend more human resource checking the log is abnormal or not, they may not get a result that the problem is the huge amount of data. For the purpose of checking the logs for a result, the overhead of human resources would be spent very much.

With a view to improving the problem, the study proposed a web log filter scheme for web abnormal behaviors. The main purpose of the scheme is to decrease the large amount of unnecessary data to improve the quality of data quickly for follow-up artificial judgment and to reduce the overhead of human resources. Even it would be replaced part of the human resources for part of works of the police.

*Corresponding author: tsengcyt@gm.ntpu.edu.tw

DOI：10.6159/IJSE.2014.(4-4).05

2 Related Works

In this section, we discuss the related works about the web log filter.

The study about web log analysis for web attack behaviors is not a widely discussed topic, even the topic is important and effective for the detection of intrusion behaviors.

Emilie Lundin Barse and Erland Jonsson[1] proposed that the importance and the effectivity of the study about web logs for attack detection. They also propose a framework for testing the need of attack detection with web logs. At last, they verify that the web log is effective for the attack detection.

Then, with the network services are becoming more complex and changeful, the discussion of study based on web intrusion behaviors is also increased.

The study[2] is proposed an algorithm of combining the different format of Apache Log. It is also proposed that we must have different preprocess process for the data to deal with the two kinds of purpose: the data preprocessing about the detection of web intrusion behaviors and the data preprocessing about the analysis of behavior that user browses the web.

Another Study[3] is proposed that log file must be divided into four categories. It must be selected the log file of the suitable category for different purpose.

At last, the study[4] is proposed a data preprocessing architecture about the detection of intrusion behaviors. It is also proposed a concept that uses the architecture to real-time detection.

In addition, a study[5] proposed an implemented program for identifying the malicious user that it extracts the keywords of attacks by analyzing web server logs in recent years.

3 Design Model

In this section, the overall scheme designs and used method are explained. The filter scheme consists of following phase:

- Data acquisition
- Filter the log of known attack behaviors
- Filter the log of unknown abnormal behaviors
- Output the results

First, data acquisition must be prepared before the scheme. We use the data which collected from a network. In this study, IIS server access logs are used.

After acquisition the data, in order to filter the log of known attack behaviors, we use the filter rules of PHP Intrusion Detection System (PHPIDS) to compare logs with regular expression. Then, we preprocess the data and extract all of the appeared characters of Uniform Resource Locator (URL) to form numerical feature matrix of the preprocessed data. The dimensionality of feature matrix is reduced using random projection. Subsequently, the suspicious score for each data point from the whole dataset can be calculated using the Mahalanobis distance. We will use the suspicious score to filter the log of unknown abnormal behaviors. Finally, we combine the data logs of two phases and output the results which are sorted with the suspicious unknown user, as illustrated in Fig. 1.

3.1 Data acquisition

In our research, we use IIS server access logs. The data comes from a real-life company web server. Actual intrusion attempts and other abnormal log lines are included in the data. We examine all of the log lines from web log files. A single log line uses the following format:

2013-02-03 22:16:27 W3SVC735099204 203.73.140.20 GET /faq.aspx page=1&fag=99999999%20oR%207=7 80 - 94.123.126.206 Mozilla/4.0 200 0 0

This format is called W3C Extended Log File Format [12], which is the default format of IIS web log formats. These logs contain various information about the network traffic, such as date, timestamp, client IP, server IP, URL, HTTP status code and the amount of transferred bytes. These information may contain different intrusion attempts, such as SQL injections, especially in the HTTP request part. For our research, the string about HTTP request from the logs will be extracted to use.

The more detail of used data and used methods are described in the following subsections.

3.2 Filter the log of known attack behaviors

In the phase of filter the log of known attack behaviors, we use the filter rules of PHP Intrusion Detection System (PHPIDS) to compare all of the log lines with regular expression

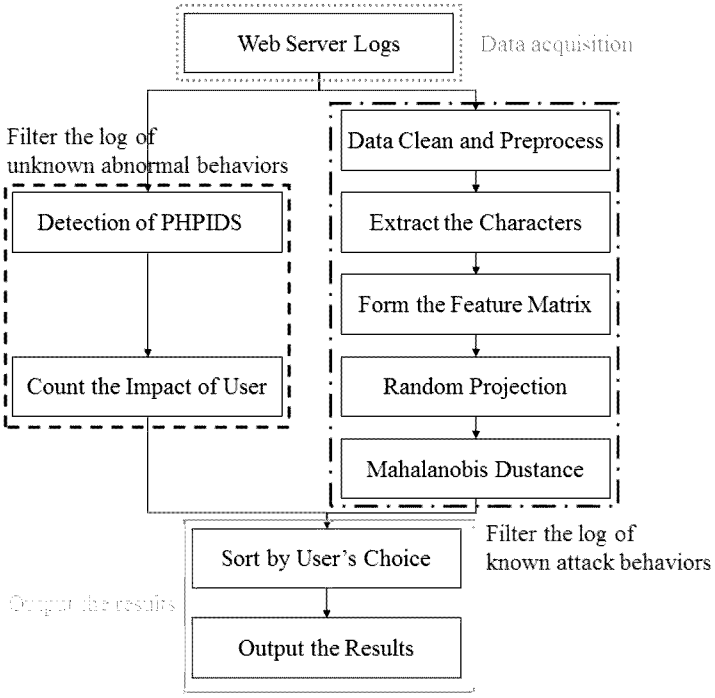

Fig. 1 Overall scheme designs

and find the suspicious unknown user with the log of known attack behaviors.

PHPIDS is an intrusion detection system for PHP environment. It uses the pre-defined filter rules to compare all of the data which is inputted from user and uses the numerical impact to give the compared data a tendency for known attack behaviors. The pre-defined filter rules of PHPIDS can be seen in Fig. 2.

In our study, we use rules, tags and impact from the filter rules file to compare the logs. First, we use rules to compare two fields from the logs, which are cs-uri-stem and cs-uri-query, record tags and impact for the log line if the two fields of the log line matches the rules and flagged the log line as abnormal log.

3.3 Filter the log of unknown abnormal behaviors

In order to improve the quality of data, the raw data must be preprocessed. We preprocess and clean the raw data with the two steps: user identification and data clean.

In user identification, we use the two fields: c-ip and cs(User-Agent) from the raw data and check these fields from all of the log lines of the whole data. While we check these fields of the log line, an ip address will be regarded as a user if we find a new ip address. However, an ip address with different cs(User-Agent) will be regarded as a new user.

After identifies the user, we clean the logs from raw data which matches the following conditions:

We clean the log line by checking the cs-uri-stem field. If the filename extension is image, music, movie or style sheet, such as .jpg, .png, .gif, .swf and .css, the log line will be deleted.

We save the log line by checking the sc-status field. If the status with code 400 series or 500 series, the log line should be kept because

they may be considered as anomaly actions which are caused many errors by user. The user is subject to suspect.

We clean the log line by checking the cs-method, cs-uri-query, sc-status field. If the method is GET, the uri-query is '-' and the status with code 200 series, the log line will be deleted.

Then, we extract the character from the preprocessed data. We use the two fields: cs-uri-stem and cs-uri-query of the logs to find the extracted character which is not repeated. We define these characters as an index to a matrix. Subsequently, we count the frequency of these characters from all of the log lines to form a vector of a matrix. We can get the vector list to form a feature matrix until all of the vector of the log lines would be counted.

In order to reduce the dimensionality of the feature matrix, we use random projection (RP)

```
<filter>
    <id>32</id>
    <rule><![CDATA[(?:[^\w\s=]on(?!g\&gt;)\w+[^=_+-]*=[^S]+(?:\W|\&gt;)?)]]></rule>
    <description>Detects possible event handlers</description>
    <tags>
        <tag>xss</tag>
        <tag>csrf</tag>
    </tags>
    <impact>4</impact>
</filter>
<filter>
    <id>33</id>
    <rule><![CDATA[(?:\<\w*:?\s(?:[^\>]*)t(?!rong))|(?:\<scri)|(<\w+:\w+)]]></rule>
    <description>Detects obfuscated script tags and XML wrapped HTML</description>
    <tags>
        <tag>xss</tag>
    </tags>
    <impact>4</impact>
</filter>
<filter>
    <id>34</id>
    <rule><![CDATA[(?:\<\/\w+\s\w+)|(?:@(?:cc_on|set)[\s@,"=])]]></rule>
    <description>Detects attributes in closing tags and conditional compilation tokens</description>
    <tags>
        <tag>xss</tag>
        <tag>csrf</tag>
    </tags>
    <impact>4</impact>
</filter>
```

Fig. 2 The filter rules of PHPIDS

that the goal is to project high-dimensional data into a lower-dimensional space using a random matrix [6]. The idea is based on Johnson-Lindenstrauss lemma [7]. It states that points can be projected to a randomly generated subspace and still the distances between points are approximately preserved.

Given the original data with d dimensions, the new subspace has k dimensions so that $k \ll d$. If the original data matrix is X_{dxN} and the randomly generated matrix is R_{kxd}, the random projection of the data can be calculated using the following equation [6].

$$X_{kxN}^{RP} = R_{kxd} X_{dxN}$$

The actual creation of the random matrix R is the most important phase of the method. Basically, R should be orthogonal, but the process of orthogonalization is computationally expensive. However, Hecht-Nielsen[8] proposed: "There exists a much larger number of almost orthogonal than orthogonal directions in a high-dimensional space". Based on the result, R could be almost orthogonal.

Instead of using Gaussian distributed variables, a much simpler probability distribution has been proposed by Achlioptas [9]:

$$r_{ij} = \sqrt{3} \times \begin{cases} +1 & with\ probability\ \frac{1}{6} \\ 0 & with\ probability\ \frac{2}{3} \\ -1 & with\ probability\ \frac{1}{6} \end{cases}$$

In this study, we compute the random matrix with this distribution, as proposed by Achlioptas [9].

Then, We use the projected matrix that the suspicious score for each data point from the whole dataset can be calculated using the Mahalanobis distance. Mahalanobis distance[10] is a distance metric. This distance metric takes into account the correlations of the data. We use the method to create basically a suspicious score.

If we have a data set X with an individual data vector being $X = (x_1, x_2, \cdots, x_N)^T$, as well as mean $\mu = (\mu_1, \mu_2, \cdots, \mu_N)^T$ and covariance matrix S, the Mahalanobis distance score for each data point can be formally defined as follows [11]:

$$D_M = \sqrt{(x-\mu)^T S^{-1} (x-\mu)}$$

Finally, we set a threshold for these scores that flag whole data points as abnormal.

3.4 Output the results

In the phase of output the results, we use the outcome from the forward steps to sort the suspicious level of malicious unknown user. First, we find the highest impact of logs, the highest suspicious score and count the number of abnormal logs of these users. Then, we sort the logs by user's choice. We set four situations for user: only sort known attack logs, only sort unknown abnormal logs, combine the logs of these phases and sort known attack logs first, combine the logs of these phases and sort unknown abnormal logs first. User can choose the sort which would be observed. Finally, we output two results: the sorted logs by user's choice and the whole raw data of these malicious unknown users.

4 Evaluations

In order to evaluate the efficacy of the scheme, we use a program which is written by us. We use it as our experimental environment. In addition, we use the server access log and security incident report from Nokia ICM to form a result by artificial judgment. The result is a standard which is used to verify the experimental result.

The number of test data is 196186. The number of the abnormal logs which is artificial judged is 94362. The range of timestamp of the logs is about four days. After we preprocess the raw data, the number of character which we extract is 584. It is the dimensionality of the feature matrix. We also have two variables which we have to decide: k, threshold. We discuss the variables in the following sections.

The evaluation of our study is based on the following performance metric:

Match Rate (MR, %): the number of filtered logs which is abnormal by artificial judgment / the number of abnormal logs which is artificial judged.

Mistaken Select Rate (MSR, %): the number of normal logs which is artificial judged / the number of filtered logs.

4.1 The Value of k

First, we set the default threshold as 5. We observe the change of MR when k is 2 to 7, as illustrated in Fig. 3.

As illustrated in Fig. 3 and Table. 1, the MR is higher if the k increases. However, the MR increases less when k increases to 6 and the rate of growth of used memory is uneconomical. When k increases to 7, the rate of growth of used memory is almost zero. Thus, we decide k as 5.

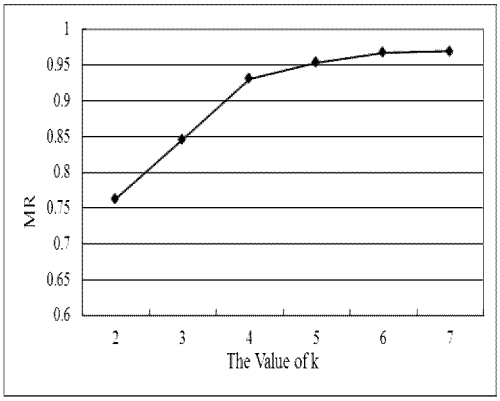

Fig. 3 the change of MR when k changes

Table. 1 the result of MR and used memory

	MR	Size of Used Memory
k = 2	0.76324	365MB
k = 3	0.84622	382MB
k = 4	0.93086	449MB
k = 5	0.95343	462MB
k = 6	0.96762	472MB
k = 7	0.96917	504MB

4.2 The Value of Threshold

First, we set the k as 5 from the forward section. We observe the changes of MR and MSR when threshold is 0.4 to 0.8, as illustrated in Fig. 4 and Fig. 5.

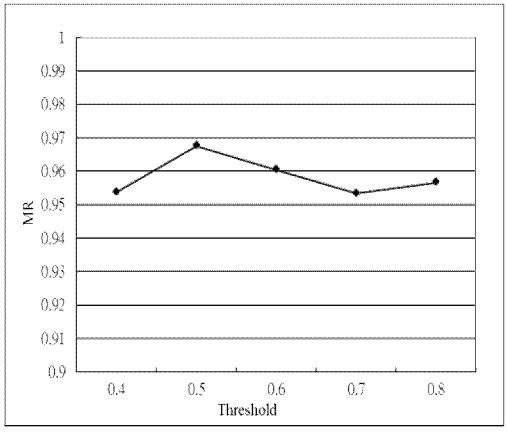

Fig. 4 the change of MR when k changes

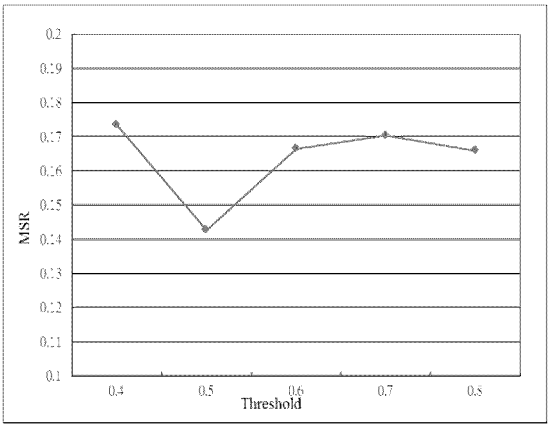

Fig. 5 the change of MSR when k changes

Table. 2 the result of MR and MSR

	MR	MSR
Threshold = 0.4	0.9538	0.1734
Threshold = 0.5	0.96762	0.1427
Threshold = 0.6	0.96040	0.1665
Threshold = 0.7	0.95343	0.1703
Threshold = 0.8	0.95662	0.1659

Because of the purpose of our study, we need the MR is more higher and the MSR is more lower. As illustrated in Fig. 4, Fig. 5 and Table. 2, the MR is highest and the MSR is lowest when the threshold is 0.5. Thus, we decide threshold as 0.5.

4.3 Discussion

We use the experiment result to decide the value of two variables. Finally, the k is decided as 5 and the threshold is decided as 0.5. We use the two variables to experiment again and we get the result as follows:

Table. 3 the final result of the scheme

	Abnormal (Filter)	Normal (Filter)
Abnormal (Judge)	91307	3055
Normal (Judge)	15202	86589

The MR is 0.96762 that is about 96.7%.

The MSR is 0.14273 that is about 14.3%.

In summary, we can know the best of the parameter for k and threshold. We will apply these parameters in the scheme can help people to decrease the raw data largely for the follow-up artificial judgment. It could decrease the overhead of daily security check, which problem is that the data would be made very much every day.

5 Conclusions

With the network services become complex and various, the information and the privacy which are stored in network become more valuable. In order to protect the security and privacy of these network services, daily security check and analysis actively is very important. Based on the purpose, we propose a web log filter scheme for web abnormal behaviors that combines the filter method of two phases: filter the log of known attack behaviors and filter the log of unknown abnormal behaviors. And then, we sort the result by user's choice and output two kinds of results. At last, although we choose a method which computation and implement are not complicated, the filtered data is most matched the judged data. The experiment result shows that the scheme has the ability for reducing the overhead of daily security check and analysis by artificial judgment. Thus, we resolve the problem of artificial security check, which is difficult to skip unnecessary data from the amount of raw data.

References

[1] E. L. Barse, and E. Jonsson, "Extracting attack manifestations to determine log data requirements for intrusion detection," Computer Security Applications Conference, 2004. 20th Annual, pp. 158-167, 2004.

[2] S. E. Salama, M. I. Marie, L. M. El-Fangary, and Y. K. Helmy, "Web Server Logs Preprocessing for Web Intrusion Detection," Computer and Information Science, vol.4, no.4, pp. 123-133, 2011.

[3] L. K. Joshila Grace, V. Maheswari, and D. Nagamalai, "Analysis of Web Logs and Web User in Web Mining," International Journal of Network Security & Its Applications (IJNSA), Vol.3, No.1, pp. 99-110, 2011.

[4] P. V. Patil., and D. Patil., "Preprocessing Web Logs for Web Intrusion Detection," International Journal of Applied Information Systems (IJAIS) Proceedings on International Conference and workshop on Advanced Computing 2013, vol.ICWAC, No.3, pp. 11-15, 2013.

[5] H. Jelodar, "Identification of Internet Hackers' Attacks through Keywords by Analyzing the Server Log File," ARPN Journal of Sys-

tems and Software, Vol.3, No.6, pp. 102-105, 2013.

[6] E. Bingham and H. Mannila, "Random projection in dimensionality reduction: applications to image and text data," in Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, pp. 245–250, 2011.

[7] W. Johnson and J. Lindenstrauss, "Extensions of lipschitz mappings into a hilbert space," Contemporary mathematics, vol. 26, no. 189-206, pp. 1–1, 1984.

[8] R. Hecht-Nielsen, "Context vectors: general purpose approximate meaning representations self-organized from raw data," Computational intelligence: Imitating life, pp. 43–56, 1994.

[9] D. Achlioptas, "Database-friendly random projections," in Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems. ACM, pp. 274–281, 2011.

[10] P. C. Mahalanobis, "On the generalized distance in statistics," Proceedings of the National Institute of Sciences (Calcutta), vol. 2, pp. 49–55, 1936.

[11] R. De Maesschalck, D. Jouan-Rimbaud, and D. L. Massart, "The mahalanobis distance," Chemometrics and Intelligent Laboratory Systems, vol. 50, no. 1, pp. 1–18, 2000.

[12] Microsoft TechNet(1996) - W3C Extended Log File Format (IIS 6.0). Retrieved March 23, 1996, from http://www.microsoft.com/technet/prodtechnol/WindowsServer2003/Library/IIS/676400bc-8969-4aa7-851a-9319490a9bbb.mspx?mfr=true