Web Forensic Evidence of SQL Injection Analysis

針對 SQL Injection 攻擊鑑識之分析

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

Bernard Chia[2]
National Taipei University
s710183203@mail.ntpu.edu.tw[2]

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

Abstract

In the WEB 2.0 generation, web attack becomes common and widely exploits by the intruders to unauthorized access. According to the survey from OWASP (Open Web Application Security Project's), SQL injection attack (SQLIA) placed the first in the OWASP 2013's top 10 list of cyber threats that web service facing. SQLIA is a technique of inserting SQL meta-characters and commands into web-based input field to change the original meaning of the SQL queries in order to manipulate the execution of the malicious SQL queries to access the databases unauthorized. It unable be detected by firewall or antivirus due to the SQLIA is just injecting meta-character and do not have any malicious. Hence, forensic analysis to find out the evidence attack play an important role to making conclusion about and incident to prove or disprove intruder's guilt. Methodologies forensic analyses of web application that present previously are only simple statistical analysis, parsing capabilities or simple signature matching. Thus, we proposed a method by analyzing the URL request and decode it before analyzing with the rule set that provided by PHPIDS. After that we, cluster these attacks by calculate the distance with every cluster and cluster it with the nearest centroid point. To find the pattern of the SQL injection to cluster these attacks, we apply a method with extracting the SQL keyword as token set form URL request and analyze these request based on K-mean method to find the standard centroid to cluster these attacks.

Keywords: SQL injection, Request decoding, SQL keyword, K-means

摘要

在網路 2.0，入侵者以未經授權的方式存取資料庫內容的網路攻擊被廣泛使用。根據 OWASP 的調查中，SQL 注入攻擊（SQLIA）成為網路攻擊排行榜之冠。SQLIA 是插入的 SQL 元字元和命令以更改原本的 SQL 查詢的內容，以執行惡意的 SQL 查詢來對資料庫進行攻擊。由於 SQLIA 的方式並沒有明顯的惡意特徵，所以 SQLIA 不易被偵測。因此，網路攻擊鑑識分析在此扮演非常重要的角色，以找出攻擊者攻擊資料庫的證據。對於過去所提出的 Web 攻擊分析方法只是一般的統計分析，僅僅只透過語法分析或簡單的特徵比對，效果並不顯著。因此，我們提出了一種方法來分析與分類 SQLIA。首先，我們會將收集到的 URL Request 進行 Decode 動作後，接著利用 PHPIDS 所提供的規則來進行比對，最後我們透過計算各別 SQLIA 與每個攻擊的 cluster 中心之距離來對此攻擊進行分類。為了找出 SQLIA 的特徵模式，我們利用 URL Request 內的 SQL 關鍵字作為特徵值並利用 K-Mean 方法來分析與分類。

關鍵字：網路攻擊鑑識，Decode，SQLIA，K-Mean

1. Introduction

In the WEB 2.0 generation, web attack becomes common and widely exploits by the intruders to unauthorized access. According to the survey from OWASP (Open Web Application Security Project's), SQL injection attack (SQLIA) placed the first in the OWASP 2013's top 10 list of cyber threats that web service facing. SQLIA is a technique of inserting SQL meta-characters and commands into web-based input field to change the original meaning of the SQL queries in order to manipulate the execution of the malicious SQL queries to access the databases unauthorized. Since SQL injection is difficult to detected, therefore web forensic analysis based on log files plays an important role to find out the attack evidence.

There are many techniques to support web forensic analysis; one of the popular techniques is log analysis. Log analysis [1] is one of the tasks in which forensic analysis rely on. Log files are like the black box on an airplane that recorded the event occurred within an organizations system and networks. It recorded some important information token such as IP address, date, request sent, status and bytes send and each have meaning.

In previous research, they are focusing on how to SQL injection but not focus on post-analyzing. Author [2] proposed the improved "require/provide" model which is established cooperation between statistical and knowledge-based model to detect the SQL injection. Even it is a method that can apply in post forensic analysis, but since it too complicated to apply due to the reason it need collect those different types of log files and correlate it and make this way become complicated. The work of [3] is proposed experiment with Bayesian clustering to detect anomalies log line. It might be have highly false negative since it just a simply statistical analysis.

In this paper, we proposed a post forensic methodology to analyze SQL injection attack by analyzing the URL request and decode it before analyzing with the rule set that provided by PHPIDS. After that we, cluster these attacks by calculate the distance with every cluster and cluster it with the nearest centroid point. To find the pattern of the SQL injection to cluster these attacks, we apply a method with extracting the SQL keyword as token set form URL request and analyze these request based on K-mean method to find the standard centroid to cluster these attacks.

The rest of the paper is organized as following: Section 2 is introducing these previous techniques. Section 3 is describing the detail of the proposed method. Experimental results are discussed in section 4 and section 5 is conclusion.

2. Related Work

Our work is closely related with the following subjects: Detection method and clustering method. Therefore we will introduce previous technique in the following section.

The rule-based strategy defines the static rules which have to be defined by the rules before the analysis can be made. Those rules can be like detection of certain characters or more complex like session fixation attacks. Static rules are defined once and stay the same during the detection phase. They have to be defined and specifically crafted for each application. Static rules are pre-know values like certain input characters, a fixed length of a parameter or an upper limit of a transfer. Even Rule based detection is widely use in detection but it will produce high false negative rate when unknown attack has been detected. [4]

SQL injection can perform in various ways. Therefore, in the field of research, most works have focus on the other solution. For instance, the T. Naoki [5] proposed techniques of SQL injection attack based on the number of characters (alphabets, symbols, numeric characters, etc.) in the input as feature for analysis. Frank S [6] proposed a technique parsing of the attack strings.

The field of log mining is closely related to log file forensics. The work of [7] is proposed experiment with Bayesian clustering to detect anomalies in various kinds of log data. [8], [9] proposed used unsupervised clustering algorithm to find anomalies. The discipline of web log mining and knowledge are discovering as describe in [10].

S. Mukkamala [11] proposed the improved "require/provide" model which is established cooperation between statistical and knowledge-based model to detect the SQL Injection Attack. P. Ning [12] and [13] proposed the alert correlation model based on pre-requisites and consequences of the individual detected alerts. A knowledge data "hyper alert type dictionary" contains rules that describe the conditions where prior behaviors prepare for later ones.

3. Web forensic analysis Design

The proposed Web forensic analysis system consists of 2 stages: Web attack forensic implementation and k-mean clustering. The flow chat of proposed algorithm is given in figure 1.

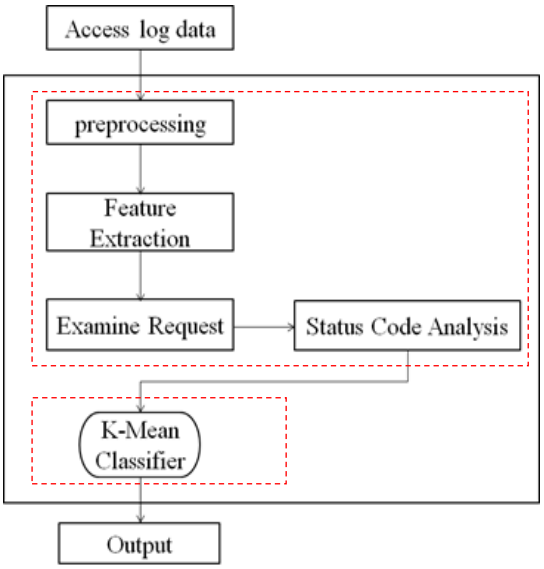


Figure 1: Web Forensic analyze methodology

A. Web Attack Forensic Implementation

This stage is discussed how we process and analyze the server log files to find out the evidence of an attacker. This stage involve is 4 process where is preprocessing, feature extraction, request examination and last is status code analysis.

1. *Preprocessing:* we collect data that we aim to analysis. In this paper, we choose the log files from the web server with the apache web service. Log files is the history of the user's request that send to the server, therefore, it have many clues about the user, it a resource to find out the attack evidence of the intruders. Log provides some information and through the information that provide by the logs, we can know who the intruders is and what request he/she had send to server. After collecting data, then is cluster of each user.

2. *Feature Extraction:* For the forensic techniques based on the request, we used the value of the URL-request and status code of the URL to representing the evidence to detect attack. To perform the request extraction, we refer the following algorithms [14]. Suppose the log file is that consists millions of records where m is the number of request in the log data. Each query in the log files has structure as figure 2 show. Therefore $\boldsymbol{u_i = path_i + q_i}$ and $\boldsymbol{q_i = (a_1, v_1) + (a_2, v_2) + \cdots + (a_i, v_i)}$ where $\boldsymbol{path_i}$ refer to the resource of the web application and $\boldsymbol{a_i}$ is name of parameter and $\boldsymbol{v_i}$ is value of parameter. And we will decode those parameter before analyze.

3. *Request Examination:* Basically, SQL injection is a trick to inject SQL command or query as input mainly in POST and GET method in the webpage. Most of the websites takes parameter from the request and make SQL query to databases. Therefore, parameter of the request is important clue in detect SQL malicious queries. In our approach, we analyze the all the GET request value with the regular expression rule that provided by the PHPIDS. Below show the algorithm that how we compare the request parameter with the rule set.

Algorithm Request_Analysis ()

```
1.  Input: Log query under test, knowledge base rules set
2.  Output: True if query match with the rules set false otherwise
3.  Begin
4.  {
5.  For each log query v_i in log data
6.      {
7.          Compare with Rule set
8.  If v_i not exists in knowledge base rules set
9.              Return false
10. Else
11.             Return true
12.     }
13. }
14. End
```

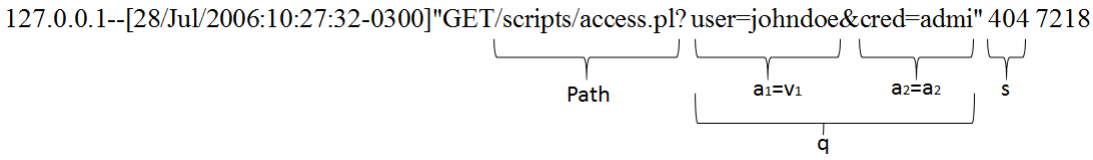
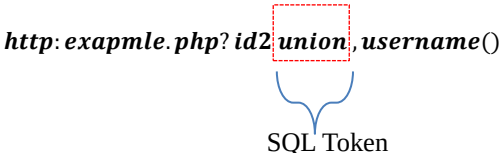


Figure 2: URL log line Request

4. *Status Code Analysis:* In the forensic analysis, we not just consider the request parameter detection, but the HTTP status code is be consider in evidence detection analyze. This is because HTTP is the statuses that refer to the request that send to server. In our approach, we just consider the HTTP status code as 2xx. This is because the request with HTTP status code 2xx is mean that the request is success communicating with the web serve.

B. K-mean Clustering

To clustering these SQL injections that found in log files, we apply k-mean to cluster these attacks. We extract the SQL keyword in logline as feature to cluster these attacks. Example below show we extract SQL keyword to analyze.



Since all the SQL injection attack requests most probably have involve SQL keyword, thus, we extract these keywords from the request. Figure 3 show the SQL keyword that we consider. The clustering job is done via k-mean function. We apply k-mean clustering to determine the center as standard pattern to cluster these attacks. Our method is as follow:
Step 1: Set value of T = $\{\mathbf{t_1}, \mathbf{t_2}, \mathbf{t_3} \ldots \mathbf{t_i}, \mathbf{t_{i-1}}\}$ where T= training set token set
Step 2: Set the value of k=number of cluster in T
Step 3: $\mathbf{D_i}$ is the cluster that divide from T based on value of k
Step 4: Assign initial centroid of every dataset $\mathbf{D_i}$.
Step 5: Take the sample of data $\mathbf{D_i}$ and calculate the Euclidean distance between the sample data and the centroid of each cluster and assign the sample to the cluster with the nearest centroid.
Step 6: Recalculate the centroid of cluster.
Step 7: Repeat step 5 and 6 until the centroid of cluster be constant.
Step 8: The centroid will become the standard centroid of cluster to classify attack.

Keyword	Token Type	Token ID
SELECT	keyword	S
UNION	keyword	U
INSERT	keyword	I
DELETE	keyword	D
CREATE	keyword	C
DROP	keyword	D_r
ALTER	keyword	A
UPDATE	keyword	U_p
AND	keyword	A_n
OR	keyword	O
NOT	keyword	N
--	syntax	S_1
1=1	syntax	S_2

Figure 3: SQL Token Table

4. Evaluation

4.1 URL Request Examination Result

By applying 5-fold cross validation analyze URL request and produce a confusion matrix as show in figure 4. From result, it shows that we detect 4288 out of 4297 from blacklist

data and detect 1 out of 2114 from whitelist data. Experiment result show that the detection rate (TP) of actual SQL injection is 99% and the detection rate (FN) of actual SQL injection just 1%. The experiment also shows that the detection rate (TN) of actual benign is 99.95% and the detection rate (FP) of actual benign is 0.05.

	Prediction (SQL Injection)	Prediction (benign)
Actual (SQL Injection)	4288 (0.99)	9 (0.01)
Actual (benign)	1 (0.05)	2113 (99.95)

Figure 4: URL request with decoding confusion matrix

By applying same dataset above but the difference is analyzing the URL request without decoding. From dataset, we detect 3690 out of 4297 from blacklist and 1 out of 2114 from whitelist. Figure 5 shows the confusion matrix of the experiment. From the table, we noticed that the detection rate (TP) of actual SQL injection is 85.8% and the detection rate (FN) of actual SQL injection is 14.2%. At the same time, the detection rate (TN) of actual benign is 99.95% and the detection rate (TP) of actual benign is 0.05.

	Prediction (SQL Injection)	Prediction (benign)
Actual (SQL Injection)	3690 (0.8586)	607 (0.1414)
Actual (benign)	1(0.05)	2113 (99.95)

Figure 5: URL request without decoding confusion matrix

From the confusion matrix above, we notice that the detection rate (TP) of actual SQL injection with decoding is higher than the URL request without decoding. This is because some intruders will encode their malicious code during injection for the purpose to evade the signature detection of Intrusion Detection System (IDS), hence it will affect the effectiveness of detection. This experiment show that URL request with/without decoding will affect the detection.

4.2 K-mean Clustering

We apply four type of SQL injection attack to our testing environment. Figure 6 show the clustering result of 4 type attack. From 1000 sample of request, we success cluster the DELETE request with zero false negative rate. Besides that, the INSERT and UPDATE request also be cluster into proper cluster with zero false negative. For the UNION request, the true positive rate is 98% and the false negative is 2%.

Predict / Actual	DELETE	INSERT	UNION	UPDATE
DELETE	1000	0	0	0
INSERT	0	1000	0	0
UNION	2	0	980	18
UPDATE	0	0	0	1000

Figure 6: clustering Result of 4 type SQL injection attack

5. Conclusion

We using a rule set that provided by PHPIDS to analyze the URL request to find out the attack evidence by the intruders. We decode every URL request before we analyzing the request to avoid the misdetection of the request. Besides that, we also apply the technique K-mean based on the ratio of SQL keyword in the URL request to find out the standard centroid for clustering purpose. At the end of experiment, it show that out proposed is effectiveness in detection and clustering every attack

References

[1] S. Allen, “Importance of understanding logs from an information security standpoint,” Tech. rep., SANS Institute, 2001, Tech. Rep.

[2] Alserhani, Faeiz, et al, "Event-based Alert Correlation System to Detect SQLI Activities," Advanced Information Networking and Applications (AINA), 2011 IEEE International Conference on IEEE, pp.175-182, 2011.

[3] P. Ma, “Log Analysis-based Intrusion Detection via Unsupervised Learning”, Master of Science, School of Informatics,

University of Edinburgh, 2003

[4] I. Koral, Richard A. Kemmerer, and P.A. Porras, "State transition analysis: A rule-based intrusion detection approach," Software Engineering, IEEE Transactions on Vol.21 No.3, pp.181-199, March 1995

[5] T. Naoki, Y. Hiroyuki, and M. Minoru, “Detection for SQL Injection with Anomaly Detection Method, ” The 71th National Convention of ISPJ collection of papers, Vol.3, pp.379–380, 2009

[6] Rietta, Frank S, "Application Layer Intrusion Detection for SQL Injection," Proceedings of the 44th annual Southeast regional conference ACM, pp.531-536, 2006.

[7] P. Ma, “Log Analysis-based Intrusion Detection via Unsupervised Learning”, Master of Science, School of Informatics, University of Edinburgh, 2003

[8] J. Stearley,“Towards Informatic Analysis of Syslogs,” In Cluster Computing, 2004 IEEE International Conference on, pp. 309–318, 2004.

[9] A.J.Oliner, A.Aiken and J.Stearley, “Alert detection in system logs,” In Data Mining, 2008.ICDM’08. Eighth IEEE International Conference on, pp.959–964, 2008.

[10] S. Mukkamala and A.H. Sung, “Identifying Significant Features for Network Forensic Analysis Using Artificial Intelligent Techniques,” In Ternational Journal of Digital Evidence, Vol.1 Isu.4, pp.1–17, 2003.

[11] Alserhani, Faeiz, et al. "Event-based Alert Correlation System to Detect SQLI Activities," Advanced Information Networking and Applications (AINA), 2011 IEEE International Conference on IEEE, pp.175-182, 2011.

[12] P. Ning, Y. Cui, D.Reeves, and D.X. Xu, “Tools and Techniques for Analyzing Intrusion Alerts,” in ACM Transactions on Information and System Security, pp.273--318, May 2004

[13] P. Ning, Y. Cui, D. Reeves, “Constructing Attack Scenarios through Correlation of Intrusion Alerts,” in Proceedings of the 9th ACM Conference on Computer & Communications Security, Washington D.C., pp245--254, November 2002

[14] Kruegel, Christopher, and G. Vigna, "Anomaly detection of web-based attacks," Proceedings of the 10th ACM conference on Computer and communications security, pp.251-261, 2003.