|  |  |
| --- | --- |
| **DETECTION OF ATTACK-TARGETED SCANS FROM THE APACHE HTTP SERVER ACCESS LOGS**  REPORT | Abstract  This study seeks to obtain data which will help to address machine learning based malware research gaps.  Tumushiime Bob Robert  DATA ANALYSIS |

Contents

[Executive Summary 1](#_Toc43669530)

[Data description 2](#_Toc43669531)

[1.1 What is web log file? 2](#_Toc43669533)

[1.2 Fields of an Access Log 2](#_Toc43669534)

[1.2.1 IP address 2](#_Toc43669535)

[1.2.2 Remote log name 2](#_Toc43669536)

[1.2.3 Authenticated user name 2](#_Toc43669537)

[1.2.4 Timestamp 2](#_Toc43669538)

[1.2.5 Access request 2](#_Toc43669539)

[1.2.6 Result status code 2](#_Toc43669540)

[1.2.7 Bytes transferred 2](#_Toc43669541)

[1.2.8 Referrer URL 3](#_Toc43669542)

[1.2.9 User Agent 3](#_Toc43669543)

[Data analysis 4](#_Toc43669544)

[2.1 Data Structuring 4](#_Toc43669546)

[2.2 Data analysis 4](#_Toc43669547)

[2.2.1 HOSTS 4](#_Toc43669548)

[2.2.2 USERNAME 4](#_Toc43669549)

[2.2.3 METHOD 5](#_Toc43669550)

[2.2.4 STATUS CODES 6](#_Toc43669551)

[2.2.5 URLs 8](#_Toc43669552)

[2.2.6 Attacks launched on servers 8](#_Toc43669553)

[2.3 Feature building 8](#_Toc43669554)

[Results 11](#_Toc43669555)

[Discussion of results 12](#_Toc43669556)

[Conclusions 13](#_Toc43669557)

[References 14](#_Toc43669558)

# Executive Summary

Nowadays, we live in a digital world. The use of computers, laptops, mobile phones, tablets, etc. has spread in any kind of business or people. In addition, majority of them have permanent connection to Internet, or are interconnected with other networks. However, in the last years, the number of automatic cyber security attacks has sharply increased, impacting negatively in business and people. These cyber-attacks include but are not limited to Structured Query Language (SQL) injection and Cross-Site Scripting (XSS)

A large number of communication logs are generated by corporate systems. These are used to monitor the internet traffic that has been flowing towards a particular source. They produce a large number of logs which are then collected and stored. The logs generated by the Systems reveal the states of a running system and contain vital information produced by the system to support diagnosis, hence, the log analysis is a vital method of detecting vulnerability scans, XSS and SQLI attacks.

It is, however, unrealistic and practical to analyze huge amounts of logs manually. This is why using machine learning to automate log analysis is important

The objectives of this study can be summarized as follows:

* To analyze the log data.
* To detect vulnerability scans.
* To detect XSS and SQLI attacks.
* To examine access log files for detections.

# Data description

The data provided is a set of access logs that is supposed to be analyzed for vulnerability scans, XSS and SQLI attacks.



## What is web log file?

Web log file is log file automatically created and maintained by a web server. Every "hit" to the Web site, including each view of a HTML document, image or other object, is logged. The raw web log file format is essentially one line of text for each hit to the web site. This contains information about who was visiting the site, where they came from, and exactly what they were doing on the web site [1].

Different servers have different log formats. Nevertheless the data in this log fragment is pretty typical of the information available.

## Fields of an Access Log

### IP address

"111.111.111.111"

This is the IP address of the machine that contacted a site.

### Remote log name

"-"

This will return a dash unless Identity Check is set on your web server.

### Authenticated user name

"-"

Only available when accessing content which is password protected by web server authenticate system.

### Timestamp

[08/Oct/2007:11:17:55 -0400]

Time stamp of the visit as seen by the web server. -0400 is time zone designator of your web server.

### Access request

"GET / HTTP/1.1"

The request made. In this case it was a "GET" request (i.e. "show me the page") for the file "/" (homepage) using the "HTTP/1.1" protocol.

### Result status code

"200"

The resulting status code. "200" is success. This tells you whether the request was successful or not.

### Bytes transferred

"10801"

The number of bytes transferred. This tells you how many bytes were transferred to the user, i.e. the bandwidth used.

### Referrer URL

"http://www.google.com/search?q=log+analyzer&ie=utf-8&oe=utf- 8&aq=t&rls=org.mozilla:en-US:official&client=firefox-a"

The referring url. This is the page the visitor was on when they clicked to come to this page. Usually this will mean that this page has a link to yours, but sometimes this is simply the page the user was looking at when they typed in your address into their browser, or clicked on your address in some other software such as a news reader or an email client.

### User Agent

"Mozilla/5.0 (Windows; U; Windows NT 5.2; en-US; rv:1.8.1.7) Gecko/20070914 Firefox/2.0.0.7"

The "User Agent" identifier. The User Agent is whatever software the visitor used to access this site. It's usually a browser, but it could equally be a web robot, a link checker, an FTP client or an offline browser.

# Data analysis



## Data Structuring

The web logs provided were in text form so the web logs were turned into data frames for easy analysis using pandas. The log file lines are split according to their fields and the data is saved in tabular form. The column titles are given according to the field of a web log line.

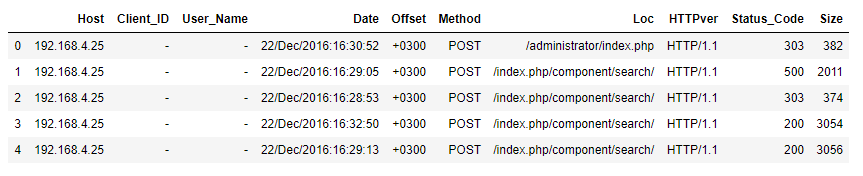


Figure 1 part of the data after being converted to Data Frame

## Data analysis

This part contained data normalization, pattern recognition and feature building.

All the URLs in the “**Loc**” column were turned to lower case to ease pattern recognition. Also the datatype of the “Size” column was changed from “**O**” to “**int**”

For each dataset created i.e. Acunetix\_df, Netsparker\_df and w3af\_df a summary of the datasets was obtained and some of the interesting insights included

### HOSTS

The hosts that accessed server are:

Captured by acunetix:

192.168.4.25

Captured by netsparker:

192.168.4.164

Captured by w3af:

192.168.4.163

192.168.4.25

192.168.1.20

### USERNAME

User\_Name in the w3af\_df

'-'

'admin'

192.168.4.163 logged on as admin

### METHOD

The POST request method requests that a web server accepts the data enclosed in the body of the request message, most likely for storing it. [2]

The HTTP GET request method retrieves information from the server [2].

The HTTP HEAD method requests the headers that are returned if the specified resource would be requested with an HTTP GET method. Such a request can be done before deciding to download a large resource to save bandwidth [2].

The HTTP OPTIONS method is used to describe the communication options for the target resource. The client can specify a URL for the OPTIONS method, or an asterisk (\*) to refer to the entire server [2].

PROPFIND — used to retrieve properties, stored as XML, from a web resource [2].

**Acunetix\_df**

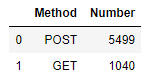


Figure 2 Methods recorded and their respective frequencies

**Netsparker\_df**

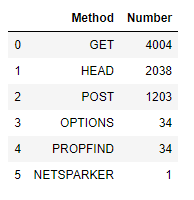


Figure 3 Methods recorded and their respective frequencies

**w3af\_df**

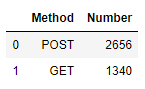


Figure 4 Methods recorded and their respective frequencies

### STATUS CODES

Status codes are issued by a server in response to a client's request made to the server.

All HTTP response status codes are separated into five classes or categories. The first digit of the status code defines the class of response, while the last two digits do not have any classifying or categorization role. There are five classes defined by the standard [3]:

* 1xx informational response – the request was received, continuing process
* 2xx successful – the request was successfully received, understood, and accepted
* 3xx redirection – further action needs to be taken in order to complete the request
* 4xx client error – the request contains bad syntax or cannot be fulfilled
* 5xx server error – the server failed to fulfil an apparently valid request

**Acunetix\_df**

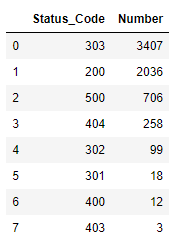


Figure 5 Status codes recorded and their respective frequencies

Majority of the requests were redirected.

303 See Other (since HTTP/1.1)

The response to the request can be found under another URI using the GET method. When received in response to a POST (or PUT/DELETE), the client should presume that the server has received the data and should issue a new GET request to the given URI.

**Netsparker\_df**

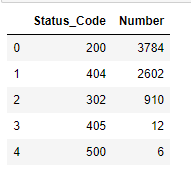


Figure 6 Status codes recorded and their respective frequencies

Most of the requests were successful.

200 OK

Standard response for successful HTTP requests. The actual response will depend on the request method used. In a GET request, the response will contain an entity corresponding to the requested resource. In a POST request, the response will contain an entity describing or containing the result of the action.

**w3af\_df**

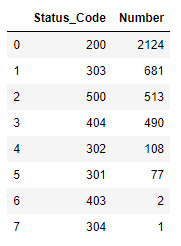


Figure 7 Status codes recorded and their respective frequencies

Most of the requests were successful.

### URLs

Most emphasis was put on the URLs as they provide information on what was accessed and what was done.

Regex function was used to identify the attacked URLs.

### Attacks launched on servers

#### XSS

Cross Site Scripting attacks work by embedding script tags in URLs/HTTP requests and enticing unsuspecting users to click on them, ensuring that the malicious java script gets executed on the victim's machine [4].

These attacks leverage the trust between the user and the server and the fact that there is no input/output validation on the server to reject java script or other active code characters.

Simple attacks contain HTML tags.

#### SQL Injection

Code injection can be any type of code like SQL, LDAP, XPath, XSLT, HTML, XML and OS command injection [4].

Here, we are focusing on the most prevalent injection, the SQL injection. For SQL injections to work, the attacker has to jump out of the original SQL statement.

This is usually done by the single quote (') or the double dash (). The single quote acts as a delimiter for an SQL query; the double dash is the comment character in Oracle and MS SQL.

## Feature building

After the attacks were identified using Regex patterns, they were compared to their respective data frames which resulted in to a new column “**State**” that identifies the attack as “**1**” and the normal as “**0**”.

The URLs were further analyzed and it was seen that the average length of attack URLs was greater than the average length of the normal URLs. This resulted in to a new column “**URL\_Length**”.

**Acunetix\_df**

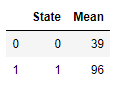


Figure 8 mean length of URLs

**Netsparker\_df**

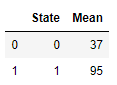


Figure 9 mean length of URLs

**w3af\_df**

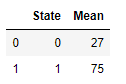


Figure 10 mean length of URLs

It was also observed that most of the attack URLs had more than one parameter compared to the normal URLs where more than 90% of them have just one parameter for all the logs.

This led to a new column “**Parameters**”.

**Acunetix\_df**

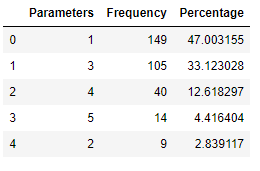


Figure 11 attack

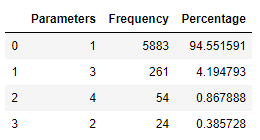


Figure 12 Normal

**Netsparker\_df**

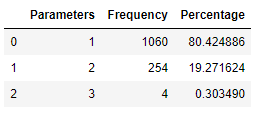


Figure 13 attack

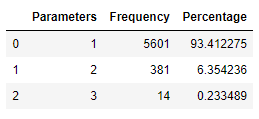


Figure 14 normal

**W3af\_df**

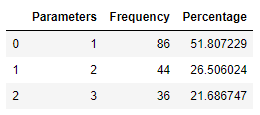


Figure 15 attack

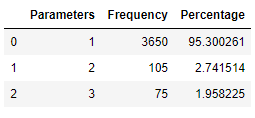


Figure 16 Normal

# Results

## Model setup

After the new features were created i.e. the URL\_Length and the number of Parameters, the data frame was encoded using dummies in pandas for the Method and Status code columns. The logs were analyzed differently throughout and the results are given according to the logs.

When the final Data sets to be used for modeling were successfully arrived at they were split in to train and test data set. The model is fit on the training score and used to predict the test data set.

Several classifications algorithms were considered and they include:

* Logistic Regression
* Linear Discriminant Analysis
* K Nearest Neighbors
* Support Vector Machine
* Decision Trees
* Random Forest
* Ada boost
* Gradient Boosting

## Training phase

The results obtained from running these classifiers are below

**Acunetix Log**

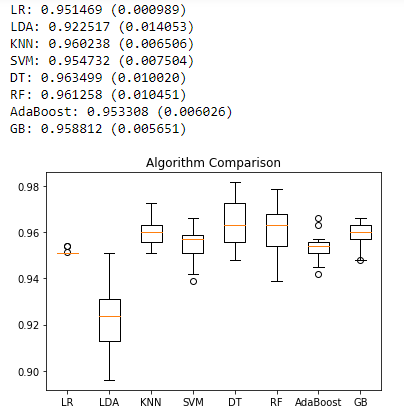


Figure 17 Box plot of the score of different algorithms

It is observed that the Decision Tree classifier performed the best

**Netsparker log**

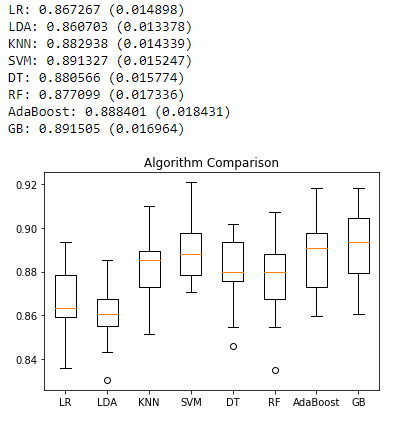


Figure 18 Box plot of the score of different algorithms

The support vector machine performed best for the Netsparker log.

**W3af log**

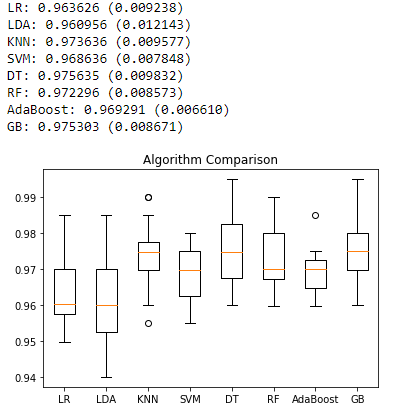


Figure 19 Box plot of the score of different algorithms

The Decision trees algorithm performed best in the w3af log.

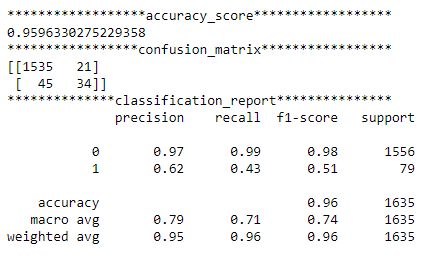
# Discussion of results

## Prediction scores and evaluation

After comparing the performances of the algorithms on the different data set the decision was to move ahead with the decision trees algorithm since it had performed well on two of the three logs and even on the log where it wasn’t the best the deviation in score was little.

The prediction results of all the logs are done using the Decision Trees Algorithm

**Acunetix log**



An accuracy score of 96% was obtained.

The confusion matrix is output in a 2by2 matrix that shows

1535 True Negative

21 False Positives

45 False Negatives

34 True Positives

**Accuracy**

96%

This is obtained from

**Recall**

Attempts to answer “What proportion of the actual positives was identified correctly”

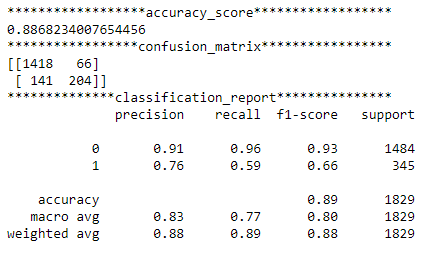
**Precision**

Attempts to answer “What proportion of positives identifications was actually correct”

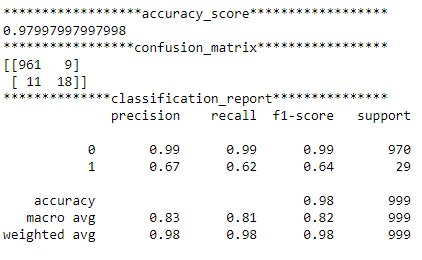
**F1 score**

It is a measure of the tests accuracy. It is the harmonic mean of the precision and recall.

**Netsparker log**



**W3af log**



# Conclusions

The highest accuracy score was 97%, the algorithm can further be improved by using more complex features can be built to make the model more accurate to be used in the real world applications.

# References

|  |  |
| --- | --- |
| [1] | N. W. L. Analyzer, "Nihuo Web Log Analyzer," Nihuo Web Log Analyzer , 2020. [Online]. Available: https://www.loganalyzer.net/log-analysis-tutorial/hit-visit-pageview.html. [Accessed June 2020]. |
| [2] | M. Org, "MDN web doc moz://a," [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods. |
| [3] | wikipedia, "Wikipedia," [Online]. Available: https://en.m.wikipedia.org/wiki/List\_of\_HTTP\_status\_codes. |
| [4] | R. Meyer, "Detecting Attacks on Web Applications from Log Files," 2008. [Online]. Available: https://www.sans.org/reading-room/whitepapers/logging/detecting-attacks-web-applications-log-files-2074. |