Contents

[1.0 EXECUTIVE SUMMARY. 2](#_Toc43649353)

[2.0 DATA DESCRIPTION. 2](#_Toc43649354)

[3.0 THE DESCRIPTION OF EACH PART OF LOG ENTRY 2](#_Toc43649355)

[4.0 DATA ANALYSIS 4](#_Toc43649356)

[5.0 HTTP METHODS 4](#_Toc43649357)

[5.1 HTTP Method in Acunetrix. 4](#_Toc43649358)

[5.2 HTTP Methods in Netsparker. 4](#_Toc43649359)

[5.3 HTTP Methods in W3AF. 5](#_Toc43649360)

[6.0 HTTP STATUS CODES. 5](#_Toc43649361)

[6.1 HTTP status codes in Acunetix. 5](#_Toc43649362)

[6.2 HTTP status code in Netsparker 6](#_Toc43649363)

[6.3 HTTP status codes in W3AF. 6](#_Toc43649364)

[7.0 RESULTS 7](#_Toc43649365)

[8.0 CONCLUSION 7](#_Toc43649366)

# 1.0 EXECUTIVE SUMMARY.

This report provides an analysis of web server access logs, data description, discussion of results and conclusions.

A Web application accepting input is vulnerable to the class of attacks known as injection attacks, which include SQL Injection and Cross-Site Scripting attacks.

The methods for analyzing include the usage of python programming modules such as pyspark, ua-parser,arg-parser, pandas.

# 2.0 DATA DESCRIPTION.

The data contains 3 datasets collected from web vulnerability scanners, for example, Acunetix, Netsparker, Web application attack and audit framework.

**An APACHE ACCESS LOG PATTERN**

It was used to obtain fields from the logs for example userid, host, timestamp, response code, protocols and HTTP methods.

The log files used for this work are in the [Apache Common Log Format (CLF)](http://httpd.apache.org/docs/1.3/logs.html#common). The log file entries produced in CLF will look something like this: 127.0.0.1 - - [01/Aug/1995:00:00:01 -0400] "GET /images/launch-logo.gif HTTP/1.0" 200 1839

# 3.0 THE DESCRIPTION OF EACH PART OF LOG ENTRY

* **127.0.0.1.** This is the IP address (or hostname, if available) of the client (remote host) which made the request to the server.
* **(-)"hyphen"**.It indicates that the requested piece of information (user identity from a remote machine) is not available in the output.
* **(- )"hyphen".**It indicates that the requested piece of information (user identity from local logon) is not available in the output.
* **[01/Aug/1995:00:00:01 -0400].**The time that the server finished processing the request. The format is: [day/month/year:hour:minute:second timezone]
  + day = 2 digits
  + month = 3 letters
  + year = 4 digits
  + hour = 2 digits
  + minute = 2 digits
  + second = 2 digits
  + zone = (+ | -) 4 digits
* **"GET /images/launch-logo.gif HTTP/1.0".**This is the first line of the request string from the client. It consists of three components: the request method (e.g., GET, POST, etc.), the endpoint (a [Uniform Resource Identifier](http://en.wikipedia.org/wiki/Uniform_resource_identifier)), and the client protocol version.
* **200 (status code).**This is the status code that the server sends back to the client. This information is very valuable, because it reveals whether the request resulted in a successful response (codes beginning in 2), a redirection (codes beginning in 3), an error caused by the client (codes beginning in 4), or an error in the server (codes beginning in 5). The full list of possible status codes can be found in the HTTP specification ([RFC 2616](https://www.ietf.org/rfc/rfc2616.txt) section10).
* **1839 last entry .**It indicates the size of the object returned to the client, not including the response headers. If no content was returned to the client, this value will be "-" (or sometimes 0).

|  |  |
| --- | --- |
| **Web Vulnerability Scanner** | **Data size** |
| Acunetix | 6539 |
| Netsparker | 7314 |
| Web application attack and audit framework | 3996, |

# 4.0 DATA ANALYSIS

According to the provided datasets.

graphs and detailed tables where generated as described in the provided Jupiter notebooks(New\_Log\_Analysis\_w3af.ipynb,New\_Log\_Analysis\_netsparker.ipynb, New\_Log\_Analysis\_acunetix.ipynb) and visualized in the google data studio dashboards provided below

<https://datastudio.google.com/s/nhoLoMbJkfY>

<https://datastudio.google.com/s/nl0joWY-Z5Q>

<https://datastudio.google.com/s/r2u2AqV8a1c>

# 5.0 HTTP METHODS

## 5.1 HTTP Method in Acunetrix.

|  |  |
| --- | --- |
| **HTTP Method** | **Number** |
| POST | 5499 |
| GET | 1039 |
| **Total** | **6,538** |

## 5.2 HTTP Methods in Netsparker.

|  |  |
| --- | --- |
| **HTTP Method** | **Number** |
| GET | 4004 |
| HEAD | 2038 |
| POST | 1203 |
| OPTIONS | 34 |
| PROPFIND | 34 |
| NETSPARKER | 1 |
| **Total** | **7,314** |

## 5.3 HTTP Methods in W3AF.

|  |  |
| --- | --- |
| **HTTP Methods in W3AF** | **Number** |
| POST | 2656 |
| GET | 1340 |
| **Total** | **3,996** |

# 6.0 HTTP STATUS CODES.

## 6.1 HTTP status codes in Acunetix.

|  |  |
| --- | --- |
| **HTTP status codes in Acunetix** | **Number** |
| 303 | 3407 |
| 200 | 2036 |
| 500 | 706 |
| 404 | 258 |
| 302 | 99 |
| 301 | 18 |
| 400 | 12 |
| 403 | 3 |
| **Total** | **1,096** |

## 6.2 HTTP status code in Netsparker

|  |  |
| --- | --- |
| **HTTP status code** | **Number** |
| 200 | 3784 |
| 500 | 6 |
| 404 | 2602 |
| 302 | 910 |
| 405 | 12 |
| **Total** | **7,314** |

## 6.3 HTTP status codes in W3AF.

|  |  |
| --- | --- |
| **HTTP status code** | **Number** |
| 200 | 2,124 |
| 500 | 513 |
| 404 | 490 |
| 302 | 108 |
| 301 | 77 |
| 403 | 2 |
| 304 | 1 |
| 303 | 681 |
| **Total** | **3,506** |

## 7.0 RESULTS

Machine learning models were produced to detect cross-site scripting and SQLI injection.

Using the datasets provided training data was obtained and a model was created for each dataset testing for the three types of users.

Models tested to deduce results for all types of users, confusion matrices, precision-recall graphs and roc curves were generated to evaluate the performance of the models.

Within the Notebooks, all information was displayed:

* machine-learning/netsparker.ipynb,
* /machine learning/w3af.ipynb,
* /machine learning/acune\_dataset.ipynb.

The results from the visualizations shows presence of attacks in the data files ,this is seen from the various error messages being sent back to the clients which includes the redirects, client errors, server errors and hence the websites should be revised for better services and reducing on the number of attack which causes some of the failures to process the requests**.**

# 8.0 CONCLUSION

More rules are needed to identify cross-site scripting, SQLI injection and vulnerability to enable proper manipulation of data.

According to type1 user detection, a rule-based model was not good enough to detect cross-site scripting and SQL injection

Datasets provided where small and could not facilitate proper machine learning results.