Concordia Institute for Information System Engineering (CIISE)
Concordia University

INSE 6140 Malware Defenses and Application Security

Project Report:

Vulnerability Analysis of Open-Source Software: WriteFreely

Submitted to:

Dr. Makan Pourzandi

Submitted By:

Student Name	Student ID	Contribution
Juan Diego Aguilar	40169314	Clickjacking Incorrect Caching
Sanjeev Kumar Paul	40155990	CWE-319 Code Smells
Shreyaben Jayeshkumar Desai	40156963	OWASP ZAP, Documentation

Date: April 26, 2021

TABLE OF CONTENTS

I. EXECUTIVE SUMMARY

With the surge of various web applications over the Internet, the security concern of such applications arises in current modern times. However, with the publishment of these applications to the internet, security and privacy breaches become a potential risk. Web security/vulnerabilities scanners and tools aid us to find out and locate the vulnerabilities depending on the given configuration and scope of each one. OWASP has mentioned the top 10 security threats in any web application.[1]

Rank	Types of security threats	Exploitability	Technical Impact
1	SQL Injection	Easy	Severe
2	Broken Authentication	Easy	Severe
3	Sensitive data exposure	Average	Severe
4	XML External Entities (XXE)	Average	Severe
5	Broken Access Control	Average	Severe
6	Security Misconfiguration	Easy	Moderate
7	Cross-site Scripting (XSS)	Easy	Severe
8	Insecure Deserialization	Difficult	Severe
9	Using component with known vulnerabilities	Average	Moderate
10	Insufficient logging and monitoring	Average	Severe

Table 1. OWASP top 10 vulnerabilities in web application

In this table, we can evidence that there are some threats which are simple to exploit and have severe impact on application, in many cases, this exploitation will not be visible to users of the application. Such attacks are practiced daily in various economic sectors such as Banking, E-commerce, Government, Health, and even IT. Therefore, the vulnerability assessment tools take essential part in dealing with exploitation and eventually these tools are helpful in mitigation process.

We were fascinated to discover vulnerabilities in our chosen open-source web application, we experimented with diverse ways and tools for exploitation and later on mitigation of them. Our report and assessment include issues such as clickjacking, cleartext password over HTTP, code smells, incorrect caching directives, autocomplete enabled at login form and some other tests for our target.

II. INTRODUCTION

Nowadays blogging has gained a lot of popularity, since its easier for users to share their thoughts throughout the internet with a single click and most of the services we use have been migrated to the cloud, so now they can be accessed anywhere throughout internet browsers. This makes Web Applications more powerful in terms of their capabilities to serve users, but at the same time they present a bigger challenge for security.

Since the services are publicly available, attackers try to exploit vulnerabilities that developers may have missed. There are many attack vectors that can be misused and may include but are not limited to the following:

- Reliance on untrusted inputs in a security decision
- Weak passwords
- Cross-site scripting (XSS) and forgery
- URL redirection to untrusted sites
- Path traversal
- Bugs

A. Open-Source Web Application

For our project, we chose an open-source web application named "WriteFreely" which is a software for writing blogs and publishing them online. This web application is developed on Go (Golang) an open-source programming language, combined with LESS coding. All mixed together with some plain JavaScript, CSS and HTML on the frontend. Since this is an open-source web application, we found the source-code on GitHub and we worked on version 0.10.0. MySQL and SQLite databases are supported by WriteFreely.

B. Tools Used

Numerous tools for vulnerability assessment are available online and many of them have been tested by us. Two types of tools have been chosen, one for source-code analysis and other for vulnerability scanning as listed below:

Burp Suite[14]**:** It is a complete platform for any web application. It captures the HTTP or HTTPS requests and plays the role of man in the middle between the user and web pages so that the user can change and automate according to the requests.

Vega Scanner[15]**:** It is a free and open-source security scanner and testing platform which is used for testing and authenticating the security of web applications. Vega aids to find vulnerabilities like Cross-site Scripting, SQL injection, Shell injection, etc. Key features are it supports graphical user interface and supports multi-platform.

SkipFish[16]**:** It is an active web application tool which creates a site map for the selected website by implementing some recursive and dictionary-based methods. Then the selected website is linked with the output of several operative security checks. It is wholly built on C programming language and that handles the HTTP and nominal CPU footprint through which impressive performance and efficiency is get.

Wireshark[17]: It is a tool that is used to trace packets that are going through your network. It is used to find dropped packets, latency and hacking attempts.

Nikto[19]**:** It is a command line tool, with the aid of which various webservers as well as dangerous files and software can be scanned for the vulnerabilities.

SonarQube[20]**:** It is an open-source platform, developed for continuous static code analysis for bug detection, security vulnerabilities and code smells over 20 programming languages. It runs as a root user and not-root user on windows however on Kali Linux, it runs as non-root user. By running predefined commands on the command prompt user gets scanning results, after completion of scanning user have to login into **localhost:9000** to get the output of the scan.

OWASP ZAP[21]**:** Open Web Application Security Project is a free and most popular security tool on the web network and, it is contributed and supported by OWASP organization, for an open-source project. In terms of making write freely web application secure at an elevated level, we adopted this tool in terms of web app development and penetration testing for the verification of security vulnerabilities.

SQLMap[22]**:** It is also an open-source penetration testing tool through which user can automatically detect and exploits SQL injection flaws and can take control over database servers.

C. SCOPE OF TEST

Our initial scans reported that the WriteFreely was vulnerable to ***SQL injection, Cleartext over HTTP and Clickjacking*** Which has been later-on exploited. Our project and report are based on above mentioned threats and false positive faced. There are some other vulnerabilities found as well by tools which will be listed.

III. SYSTEM CHARACTERIZATION

A. System Environment

We have been working on virtual environment based on Oracle Virtual Box[18], on top of this we have configured a standalone version of Kali Linux Image to run the different tools that will help with scanning, exploiting and mitigating the vulnerability.

Environment	Oracle VM VirtualBox 6.1
OS	Kali Linux; Windows
Server	Apache
Database	MySQL

Table 2. Attributes of the system environment

The first step into testing our application is setting the environment. For this, we virtualize Kali Linux using VirtualBox. During installation we learnt that more than 20GB disk space must be required for completing kali installation. After Kali installation was successfully done, we needed to get MySQL running for backend support of application. For that we had to remove the default MySQL installation from our instance, since it was presenting connection problems. Also, we downgrade MySQL version from 8.0 to 5.7 for tackling the error in connecting database.

After correctly installing the database, we continued with the WriteFreely user and table creation. This database holds all the data related to the application and its configuration. Moving further towards WriteFreely installation, we realized that available documentation was not in detail and some important steps were not covered by it, for example encryption key generation was a very important step into connecting the application with the database. After this, we cloned WriteFreely version 10.0 from its original repository, and started the configuration process.

B. Threat Type Considered (with Severity)

Through our first scanning, vulnerabilities presented below along with their severity and confidence levels were found.

VEGA Report:

Severity: High	**Severity: Medium**	**Severity: Low**
2	1	1
Cleartext Password over HTTP	Possible XML injection	Form Password field with Autocomplete Enabled
SQL injection		

SkipFish Report:

Severity: High	**Severity: Medium**	**Severity: Low**
1	0	3
Incorrect Caching Directives		Incorrect or missing charset
		Generic MIME used
		Incorrect or missing MIME type

SonarQube Report:

Severity: High	**Severity: Medium**	**Severity: Low**
0	167	0
Vulnerabilities	Code smells	

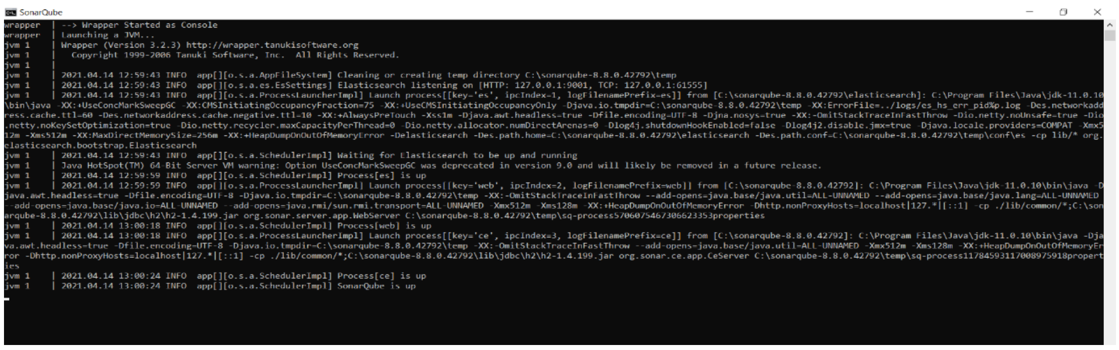
IV. <u>RESULTS</u>

A. Threat Assessment:

i) Source Code Analysis:

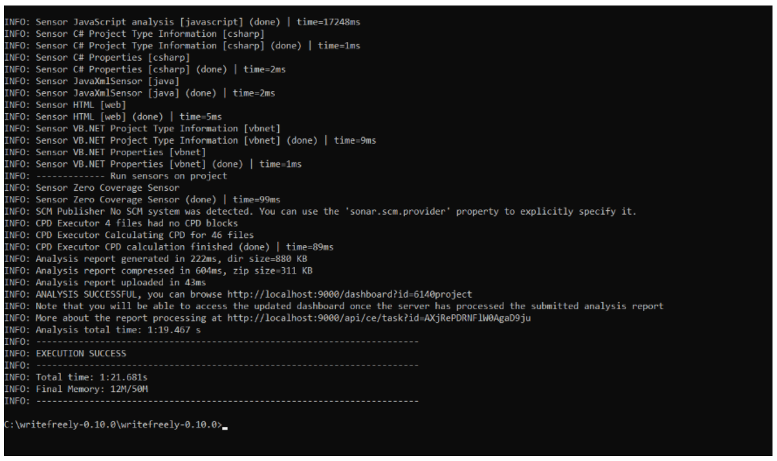
SONARQUBE:

1. Download SonarQube on Windows OS as non-root user and install it inside C Drive.
2. Download SonarScanner from official website and config the sonar-scanner. Properties and move it to the root directory of your source code.
3. Execute the StartSonar.bat file from bin folder.
4. Change directory to Source code folder in Command prompt and write sonar-scanner.bat. Output of the analysis will be sent to the localhost:9000 where user finds out whether there are bugs, smells or vulnerabilities present in code.



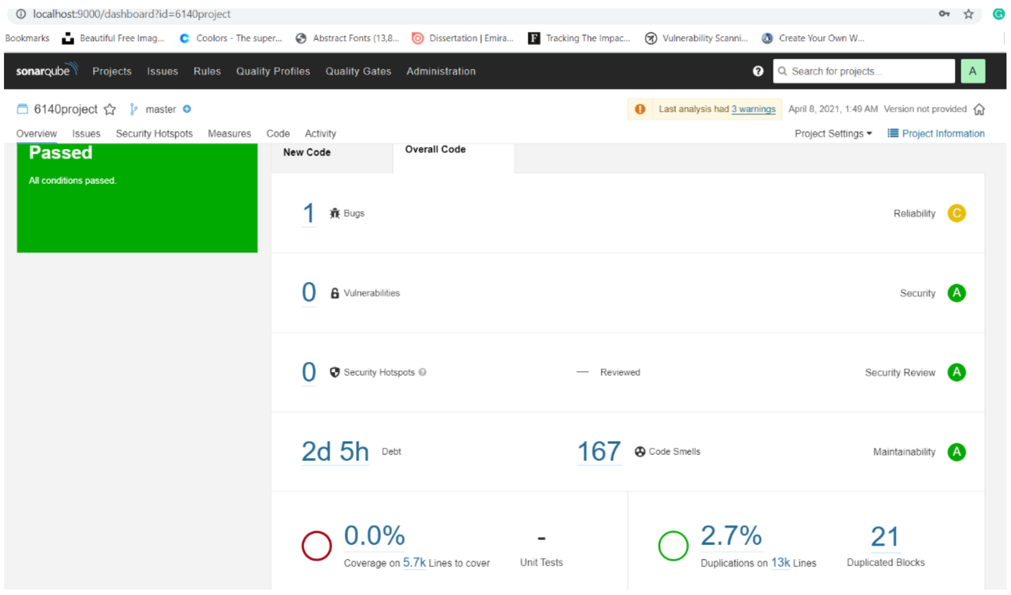
```
wrapper  | --> Wrapper Started as Console
wrapper  | Launching a JVM...
jvm 1    | Wrapper (Version 3.2.3) http://wrapper.tanukisoftware.org
jvm 1    |   Copyright 1999-2006 Tanuki Software, Inc.  All Rights Reserved.
jvm 1    |
jvm 1    | 2021.04.14 12:59:43 INFO  app[][o.s.a.AppFileSystem] Cleaning or creating temp directory C:\sonarqube-8.8.0.42792\temp
jvm 1    | 2021.04.14 12:59:43 INFO  app[][o.s.a.es.EsSettings] Elasticsearch listening on [HTTP: 127.0.0.1:9001, TCP: 127.0.0.1:61555]
jvm 1    | 2021.04.14 12:59:43 INFO  app[][o.s.a.ProcessLauncherImpl] Launch process[[key='es', ipcIndex=1, logFilenamePrefix=es]] from [C:\sonarqube-8.8.0.42792\elasticsearch]: C:\Program Files\Java\jdk-11.0.10
\bin\java -XX:+UseConcMarkSweepGC -XX:CMSInitiatingOccupancyFraction=75 -XX:+UseCMSInitiatingOccupancyOnly -Djava.io.tmpdir=C:\sonarqube-8.8.0.42792\temp -XX:ErrorFile=../logs/es_hs_err_pid%p.log -Des.networkadd
ress.cache.ttl=60 -Des.networkaddress.cache.negative.ttl=10 -XX:+AlwaysPreTouch -Xss1m -Djava.awt.headless=true -Dfile.encoding=UTF-8 -Djna.nosys=true -XX:-OmitStackTraceInFastThrow -Dio.netty.noUnsafe=true -Dio
.netty.noKeySetOptimization=true -Dio.netty.recycler.maxCapacityPerThread=0 -Dio.netty.allocator.numDirectArenas=0 -Dlog4j.shutdownHookEnabled=false -Dlog4j2.disable.jmx=true -Djava.locale.providers=COMPAT -Xmx5
12m -Xms512m -XX:MaxDirectMemorySize=256m -XX:+HeapDumpOnOutOfMemoryError -Delasticsearch -Des.path.home=C:\sonarqube-8.8.0.42792\elasticsearch -Des.path.conf=C:\sonarqube-8.8.0.42792\temp\conf\es -cp lib/* org.
elasticsearch.bootstrap.Elasticsearch
jvm 1    | 2021.04.14 12:59:43 INFO  app[][o.s.a.SchedulerImpl] Waiting for Elasticsearch to be up and running
jvm 1    | Java HotSpot(TM) 64-Bit Server VM warning: Option UseConcMarkSweepGC was deprecated in version 9.0 and will likely be removed in a future release.
jvm 1    | 2021.04.14 12:59:59 INFO  app[][o.s.a.SchedulerImpl] Process[es] is up
jvm 1    | 2021.04.14 12:59:59 INFO  app[][o.s.a.ProcessLauncherImpl] Launch process[[key='web', ipcIndex=2, logFilenamePrefix=web]] from [C:\sonarqube-8.8.0.42792]: C:\Program Files\Java\jdk-11.0.10\bin\java -D
java.awt.headless=true -Dfile.encoding=UTF-8 -Djava.io.tmpdir=C:\sonarqube-8.8.0.42792\temp -XX:-OmitStackTraceInFastThrow --add-opens=java.base/java.util=ALL-UNNAMED --add-opens=java.base/java.lang=ALL-UNNAMED
--add-opens=java.base/java.io=ALL-UNNAMED --add-opens=java.rmi/sun.rmi.transport=ALL-UNNAMED -Xmx512m -Xms128m -XX:+HeapDumpOnOutOfMemoryError -Dhttp.nonProxyHosts=localhost|127.*|[::1] -cp ./lib/common/*;C:\son
arqube-8.8.0.42792\lib\jdbc\h2\h2-1.4.199.jar org.sonar.server.app.WebServer C:\sonarqube-8.8.0.42792\temp\sq-process5706075467306623353properties
jvm 1    | 2021.04.14 13:00:18 INFO  app[][o.s.a.SchedulerImpl] Process[web] is up
jvm 1    | 2021.04.14 13:00:18 INFO  app[][o.s.a.ProcessLauncherImpl] Launch process[[key='ce', ipcIndex=3, logFilenamePrefix=ce]] from [C:\sonarqube-8.8.0.42792]: C:\Program Files\Java\jdk-11.0.10\bin\java -Dja
va.awt.headless=true -Dfile.encoding=UTF-8 -Djava.io.tmpdir=C:\sonarqube-8.8.0.42792\temp -XX:-OmitStackTraceInFastThrow --add-opens=java.base/java.util=ALL-UNNAMED -Xmx512m -Xms128m -XX:+HeapDumpOnOutOfMemoryEr
ror -Dhttp.nonProxyHosts=localhost|127.*|[::1] -cp ./lib/common/*;C:\sonarqube-8.8.0.42792\lib\jdbc\h2\h2-1.4.199.jar org.sonar.ce.app.CeServer C:\sonarqube-8.8.0.42792\temp\sq-process1178459311700897591 8propert
ies
jvm 1    | 2021.04.14 13:00:24 INFO  app[][o.s.a.SchedulerImpl] Process[ce] is up
jvm 1    | 2021.04.14 13:00:24 INFO  app[][o.s.a.SchedulerImpl] SonarQube is up
```

Fig 1: StartSonar.bat



```
INFO: Sensor JavaScript analysis [javascript] (done) | time=17248ms
INFO: Sensor C# Project Type Information [csharp]
INFO: Sensor C# Project Type Information [csharp] (done) | time=1ms
INFO: Sensor C# Properties [csharp]
INFO: Sensor C# Properties [csharp] (done) | time=2ms
INFO: Sensor JavaXmlSensor [java]
INFO: Sensor JavaXmlSensor [java] (done) | time=2ms
INFO: Sensor HTML [web]
INFO: Sensor HTML [web] (done) | time=5ms
INFO: Sensor VB.NET Project Type Information [vbnet]
INFO: Sensor VB.NET Project Type Information [vbnet] (done) | time=9ms
INFO: Sensor VB.NET Properties [vbnet]
INFO: Sensor VB.NET Properties [vbnet] (done) | time=1ms
INFO: ------------- Run sensors on project
INFO: Sensor Zero Coverage Sensor
INFO: Sensor Zero Coverage Sensor (done) | time=99ms
INFO: SCM Publisher No SCM system was detected. You can use the 'sonar.scm.provider' property to explicitly specify it.
INFO: CPD Executor 4 files had no CPD blocks
INFO: CPD Executor Calculating CPD for 46 files
INFO: CPD Executor CPD calculation finished (done) | time=89ms
INFO: Analysis report generated in 222ms, dir size=880 KB
INFO: Analysis report compressed in 604ms, zip size=311 KB
INFO: Analysis report uploaded in 43ms
INFO: ANALYSIS SUCCESSFUL, you can browse http://localhost:9000/dashboard?id=6140project
INFO: Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
INFO: More about the report processing at http://localhost:9000/api/ce/task?id=AXjRePDRNFlW0AgaD9ju
INFO: Analysis total time: 1:19.467 s
INFO: ------------------------------------------------------------------------
INFO: EXECUTION SUCCESS
INFO: ------------------------------------------------------------------------
INFO: Total time: 1:21.681s
INFO: Final Memory: 12M/50M
INFO: ------------------------------------------------------------------------

C:\writefreely-0.10.0\writefreely-0.10.0>_
```

Fig 2: Execution of Sonar-scanner.bat



localhost:9000/dashboard?id=6140project

6140project master

Last analysis had 3 warnings April 8, 2021, 1:49 AM Version not provided

Passed

All conditions passed.

Overall Code

1 Bugs		Reliability C
0 Vulnerabilities		Security A
0 Security Hotspots	— Reviewed	Security Review A
2d 5h Debt	167 Code Smells	Maintainability A
0.0% Coverage on 5.7k Lines to cover	- Unit Tests	
2.7% Duplications on 13k Lines	21 Duplicated Blocks	

Fig 3: Output at localhost:9000

Code smell indicates the characteristic in source code of application that may cause deeper problem. It is not necessary that all code smells may cause vulnerability but those codes which have been used numerous times in application along with code smell creates hazardous situation in future for running application [3]. **SonarQube suspected 167 code smells.**

ii) VULNERABILITY ASSESSMENT:

VEGA Results:

Vega Scanner is using Kali Linux. Pre-requisite to using Vega is installing libwebkit-gtk1.0.

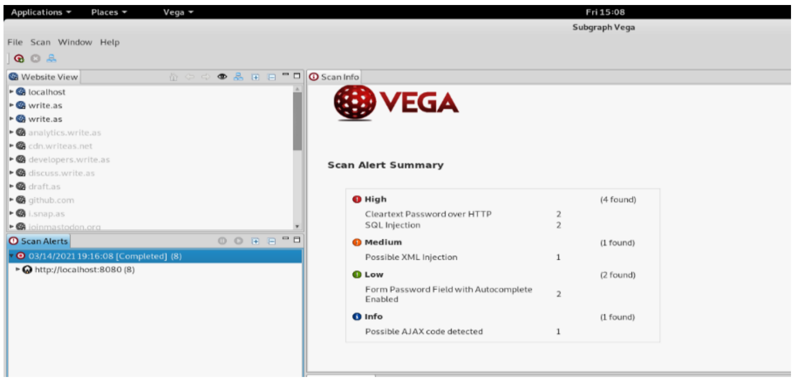
Step 1: Change Directory to Vega folder: cd Downloads -> cd Vega.

Step 2: Run Vega GUI with ./Vega.

Step 3: Click on Scan and Enter URL: http://localhost:8080/.

Step 4: Run Scan.

After Various Scans, Most frequently occurring Vulnerability was CWE 319 and under one single scan we got a summary listed below. Fig 6 describes that the autocomplete should be turned off whenever you are trying to provide sensitive information.



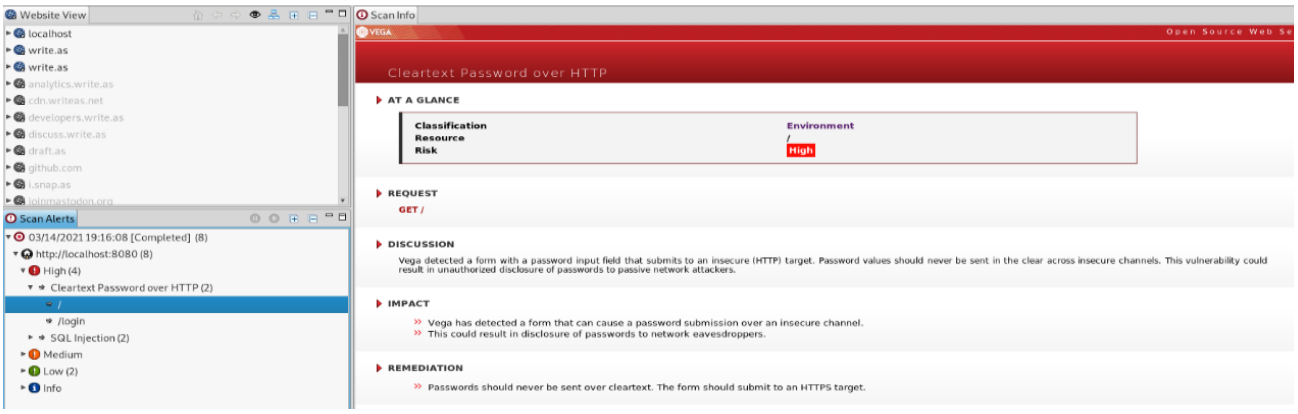
localhost
write.as
write.as
analytics.write.as
cdn.writeas.net
developers.write.as
discuss.write.as
draft.as
github.com
i.snap.as
joinmastodon.org

03/14/2021 19:16:08 [Completed] (8)
http://localhost:8080 (8)

Scan Alert Summary

High		(4 found)
Cleartext Password over HTTP	2	
SQL Injection	2	
Medium		(1 found)
Possible XML Injection	1	
Low		(2 found)
Form Password Field with Autocomplete Enabled	2	
Info		(1 found)
Possible AJAX code detected	1	

Fig 4: Vega Scan Alert Summary



localhost
write.as
write.as
analytics.write.as
cdn.writeas.net
developers.write.as
discuss.write.as
draft.as
github.com
i.snap.as
joinmastodon.org

03/14/2021 19:16:08 [Completed] (8)
http://localhost:8080 (8)
High (4)
Cleartext Password over HTTP (2)
/
/login
SQL Injection (2)
Medium
Low (2)
Info

Cleartext Password over HTTP

AT A GLANCE

Classification	Environment
Resource	/
Risk	High

REQUEST

GET /

DISCUSSION

Vega detected a form with a password input field that submits to an insecure (HTTP) target. Password values should never be sent in the clear across insecure channels. This vulnerability could result in unauthorized disclosure of passwords to passive network attackers.

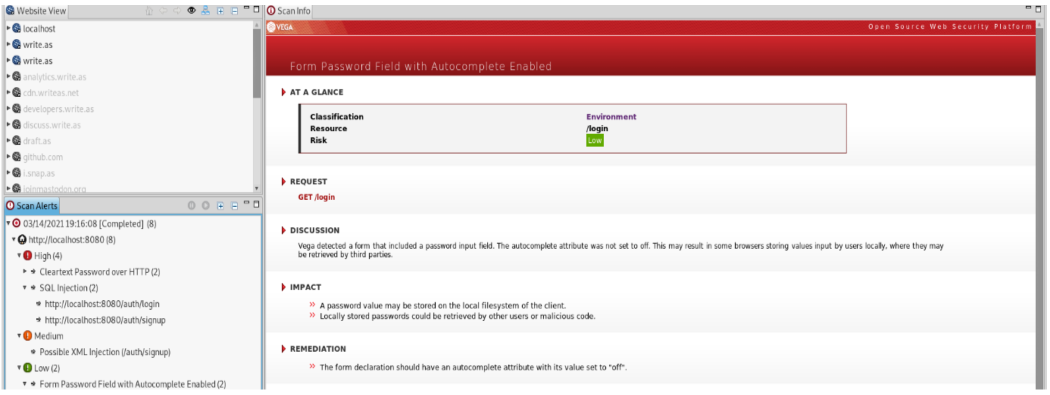
IMPACT

» Vega has detected a form that can cause a password submission over an insecure channel.
» This could result in disclosure of passwords to network eavesdroppers.

REMEDIATION

» Passwords should never be sent over cleartext. The form should submit to an HTTPS target.

Fig 5: Cleartext Password sent over HTTP by Vega scanner



localhost
write.as
write.as
analytics.write.as
cdn.writeas.net
developers.write.as
discuss.write.as
draft.as
github.com
i.snap.as
joinmastodon.org

03/14/2021 19:16:08 [Completed] (8)
http://localhost:8080 (8)
High (4)
Cleartext Password over HTTP (2)
SQL Injection (2)
http://localhost:8080/auth/login
http://localhost:8080/auth/signup
Medium
Possible XML Injection (/auth/signup)
Low (2)
Form Password Field with Autocomplete Enabled (2)

Form Password Field with Autocomplete Enabled

AT A GLANCE

Classification	Environment
Resource	/login
Risk	Low

REQUEST

GET /login

DISCUSSION

Vega detected a form that included a password input field. The autocomplete attribute was not set to off. This may result in some browsers storing values input by users locally, where they may be retrieved by third parties.

IMPACT

» A password value may be stored on the local filesystem of the client.
» Locally stored passwords could be retrieved by other users or malicious code.

REMEDIATION

» The form declaration should have an autocomplete attribute with its value set to "off".

Fig 6: Autocomplete enabled in Password field of login form.

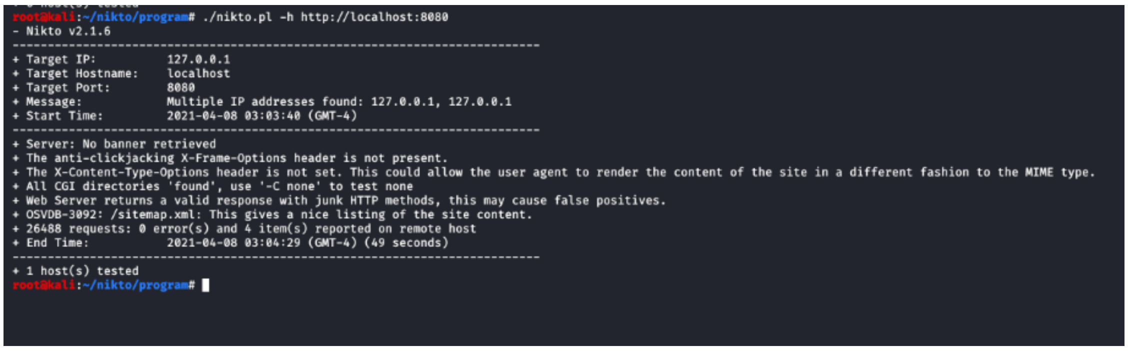
NIKTO:

Step 1: Git Clone Nikto

Step 2: Change Directory: cd nikto/program

Step 3: Command: ./nikto.pl -h http://localhost:8080/

Scanner provides information that the web application is prone to Clickjacking as X-frame-options header is not included in HTTP response. This Scan result [Fig 7] is supported by the output of OWASP ZAP [Fig 8].

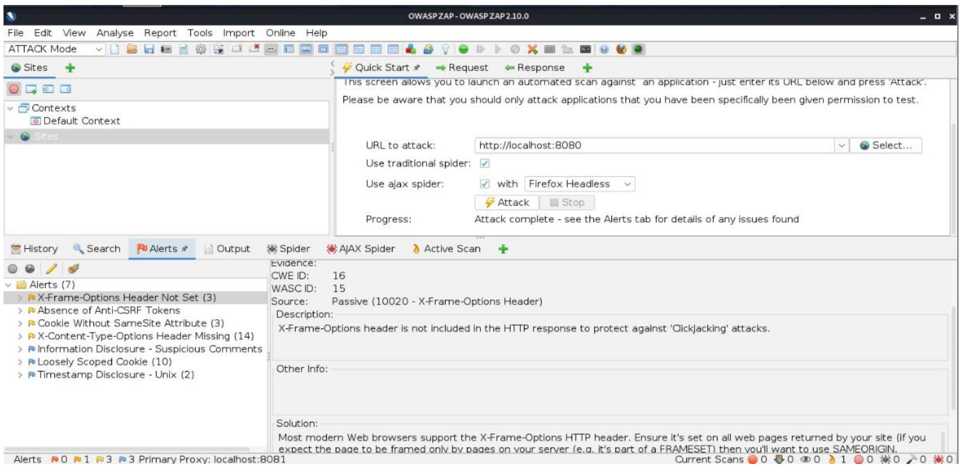


```
root@kali:~/nikto/program# ./nikto.pl -h http://localhost:8080
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          127.0.0.1
+ Target Hostname:    localhost
+ Target Port:        8080
+ Message:            Multiple IP addresses found: 127.0.0.1, 127.0.0.1
+ Start Time:         2021-04-08 03:03:40 (GMT-4)
---------------------------------------------------------------------------
+ Server: No banner retrieved
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type.
+ All CGI directories 'found', use '-C none' to test none
+ Web Server returns a valid response with junk HTTP methods, this may cause false positives.
+ OSVDB-3092: /sitemap.xml: This gives a nice listing of the site content.
+ 26488 requests: 0 error(s) and 4 item(s) reported on remote host
+ End Time:           2021-04-08 03:04:29 (GMT-4) (49 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
root@kali:~/nikto/program# 
```

Fig 7: Nikto Scan

OWASP ZAP:

OWASP ZAP has several options such as Active scan, Ajax spider and so on, also provides its output and shows alerts. An active scanning of our application gave us an alert with Alert ID: 10020-1 which means X-frame-options header is not included in HTTP response which can lead towards 'clickjacking' attack on application[2]. This scanning also provided other alerts with low risk.


OWASP ZAP - OWASP ZAP 2.10.0
File Edit View Analyse Report Tools Import Online Help
ATTACK Mode
Sites
Quick Start
Request
Response
Contexts
Default Context
History Search Alerts Output Spider AJAX Spider Active Scan
Attack
Stop
Select...

This screen allows you to launch an automated scan against an application - just enter its URL below and press 'Attack'.
Please be aware that you should only attack applications that you have been specifically been given permission to test.

URL to attack: http://localhost:8080
Use traditional spider: ✓
Use ajax spider: ✓ with Firefox Headless
Progress: Attack complete - see the Alerts tab for details of any issues found

Alerts (7)
- X-Frame-Options Header Not Set (3)
- Absence of Anti-CSRF Tokens
- Cookie Without SameSite Attribute (3)
- X-Content-Type-Options Header Missing (14)
- Information Disclosure - Suspicious Comments
- Loosely Scoped Cookie (10)
- Timestamp Disclosure - Unix (2)

Evidence:
CWE ID: 16
WASC ID: 15
Source: Passive (10020 - X-Frame-Options Header)
Description:
X-Frame-Options header is not included in the HTTP response to protect against 'ClickJacking' attacks.
Other Info:
Solution:
Most modern Web browsers support the X-Frame-Options HTTP header. Ensure it's set on all web pages returned by your site (if you expect the page to be framed only by pages on your server (e.g. it's part of a FRAMESET) then you'll want to use SAMEORIGIN.

Alerts 0 1 3 3 Primary Proxy: localhost:8081 Current Scans 0 0 0 1 0 0 0 0

Fig 8: OWASP ZAP Scan

B. VULNERABILITES SUCCESSFULLY EXPLOITED

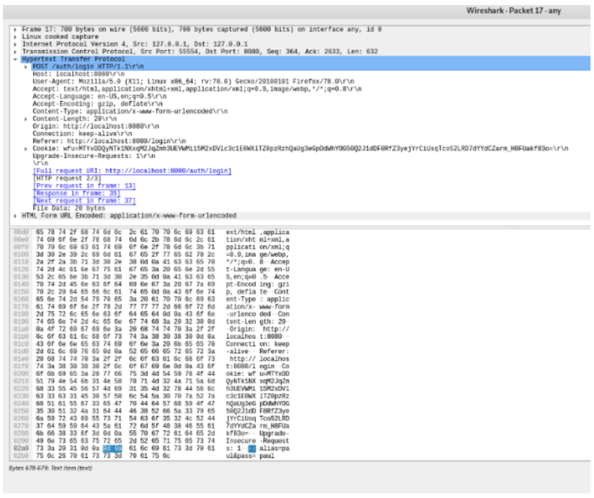
Vulnerability	**Severity**	**Tool**
CWE-319[6] (Sensitive Information Exposure)	**HIGH**	**Wireshark**
CWE-1021 (Clickjacking)[7]	**MEDIUM**	**Burp Suite**

Table 3: Vulnerabilities Exploited

i. Cleartext over http using Wireshark:

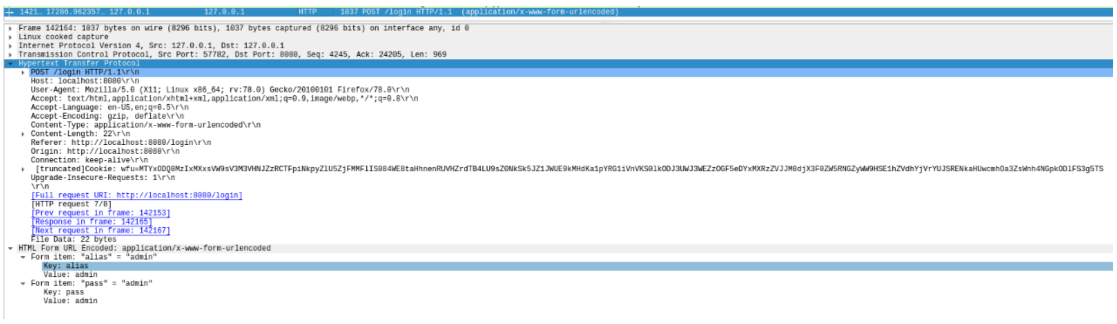
1. Open Wireshark in Kali Linux
2. Put Filter: http.request.method ==POST
3. Trace Network Packets leaving the Localhost IP address (127.0.0.1)

Fig 13 shows that User credentials are sent in the clear as alias & pass. Fig 14 shows what would happen if auto-complete enabled & Fig15 shows what happens when a new user login to account. In all the scenarios, Sensitive information was passed over HTTP channel.



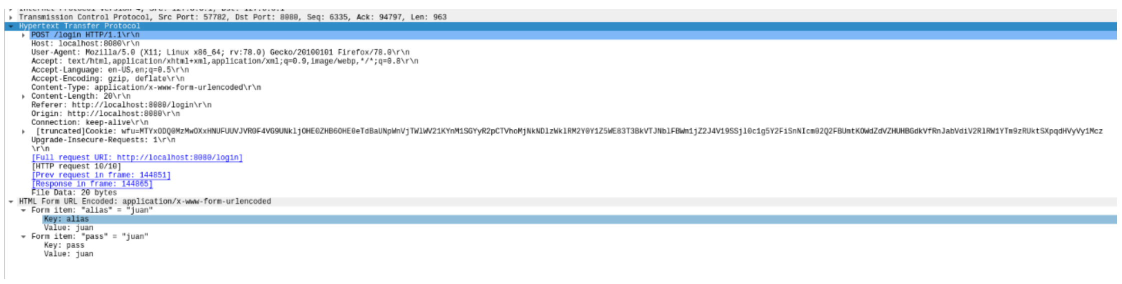
```
Wireshark · Packet 17 · any

▸ Frame 17: 700 bytes on wire (5600 bits), 700 bytes captured (5600 bits) on interface any, id 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
▸ Transmission Control Protocol, Src Port: 55554, Dst Port: 8080, Seq: 364, Ack: 2633, Len: 632
▾ Hypertext Transfer Protocol
  ▸ POST /auth/login HTTP/1.1\r\n
    Host: localhost:8080\r\n
    User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0\r\n
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8\r\n
    Accept-Language: en-US,en;q=0.5\r\n
    Accept-Encoding: gzip, deflate\r\n
    Content-Type: application/x-www-form-urlencoded\r\n
  ▸ Content-Length: 20\r\n
    Origin: http://localhost:8080\r\n
    Connection: keep-alive\r\n
    Referer: http://localhost:8080/login\r\n
  ▸ Cookie: wfu=MTYxODQyNTk1NXxqM2JqZmh3UEVWMi15M2xDVlc3c1E0WXlTZ0pzRzhQaUg3eGpDdWhYOG50Q2J1dDF8RfZ3yejYrCiUsqTco52LRD7dYYdCZarm_H8FUakf83o=\r\n
    Upgrade-Insecure-Requests: 1\r\n
    \r\n
    [Full request URI: http://localhost:8080/auth/login]
    [HTTP request 2/3]
    [Prev request in frame: 13]
    [Response in frame: 35]
    [Next request in frame: 37]
    File Data: 20 bytes
▸ HTML Form URL Encoded: application/x-www-form-urlencoded

00d0  65 78 74 2f 68 74 6d 6c  2c 61 70 70 6c 69 63 61   ext/html ,applica
00e0  74 69 6f 6e 2f 78 68 74  6d 6c 2b 78 6d 6c 2c 61   tion/xht ml+xml,a
00f0  70 70 6c 69 63 61 74 69  6f 6e 2f 78 6d 6c 3b 71   pplicati on/xml;q
0100  3d 30 2e 39 2c 69 6d 61  67 65 2f 77 65 62 70 2c   =0.9,ima ge/webp,
0110  2a 2f 2a 3b 71 3d 30 2e  38 0d 0a 41 63 63 65 70   */*;q=0. 8··Accep
0120  74 2d 4c 61 6e 67 75 61  67 65 3a 20 65 6e 2d 55   t-Langua ge: en-U
0130  53 2c 65 6e 3b 71 3d 30  2e 35 0d 0a 41 63 63 65   S,en;q=0 .5··Acce
0140  70 74 2d 45 6e 63 6f 64  69 6e 67 3a 20 67 7a 69   pt-Encod ing: gzi
0150  70 2c 20 64 65 66 6c 61  74 65 0d 0a 43 6f 6e 74   p, defla te··Cont
0160  65 6e 74 2d 54 79 70 65  3a 20 61 70 70 6c 69 63   ent-Type : applic
0170  61 74 69 6f 6e 2f 78 2d  77 77 77 2d 66 6f 72 6d   ation/x- www-form
0180  2d 75 72 6c 65 6e 63 6f  64 65 64 0d 0a 43 6f 6e   -urlenco ded··Con
0190  74 65 6e 74 2d 4c 65 6e  67 74 68 3a 20 32 30 0d   tent-Len gth: 20·
01a0  0a 4f 72 69 67 69 6e 3a  20 68 74 74 70 3a 2f 2f   ·Origin:  http://
01b0  6c 6f 63 61 6c 68 6f 73  74 3a 38 30 38 30 0d 0a   localhos t:8080··
01c0  43 6f 6e 6e 65 63 74 69  6f 6e 3a 20 6b 65 65 70   Connecti on: keep
01d0  2d 61 6c 69 76 65 0d 0a  52 65 66 65 72 65 72 3a   -alive·· Referer:
01e0  20 68 74 74 70 3a 2f 2f  6c 6f 63 61 6c 68 6f 73    http:// localhos
01f0  74 3a 38 30 38 30 2f 6c  6f 67 69 6e 0d 0a 43 6f   t:8080/l ogin··Co
0200  6f 6b 69 65 3a 20 77 66  75 3d 4d 54 59 78 4f 44   okie: wf u=MTYxOD
0210  51 79 4e 54 6b 31 4e 58  78 71 4d 32 4a 71 5a 6d   QyNTk1NX xqM2JqZm
0220  68 33 55 45 56 57 4d 69  31 35 4d 32 78 44 56 6c   h3UEVWMi 15M2xDVl
0230  63 33 63 31 45 30 57 58  6c 54 5a 30 70 7a 52 7a   c3c1E0WX lTZ0pzRz
0240  68 51 61 55 67 33 65 47  70 44 64 57 68 59 4f 47   hQaUg3eG pDdWhYOG
0250  35 30 51 32 4a 31 64 44  46 38 52 66 5a 33 79 65   50Q2J1dD F8RfZ3ye
0260  6a 59 72 43 69 55 73 71  54 63 6f 35 32 4c 52 44   jYrCiUsq Tco52LRD
0270  37 64 59 59 64 43 5a 61  72 6d 5f 48 38 46 55 61   7dYYdCZa rm_H8FUa
0280  6b 66 38 33 6f 3d 0d 0a  55 70 67 72 61 64 65 2d   kf83o=·· Upgrade-
0290  49 6e 73 65 63 75 72 65  2d 52 65 71 75 65 73 74   Insecure -Request
02a0  73 3a 20 31 0d 0a 0d 0a  61 6c 69 61 73 3d 70 61   s: 1···· alias=pa
02b0  75 6c 26 70 61 73 73 3d  70 61 75 6c               ul&pass= paul

Bytes 678-679: Text item (text)
```

Fig 9: Alias & Pass



```
1421… 17286.962357… 127.0.0.1        127.0.0.1        HTTP     1037 POST /login HTTP/1.1  (application/x-www-form-urlencoded)

▸ Frame 142164: 1037 bytes on wire (8296 bits), 1037 bytes captured (8296 bits) on interface any, id 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
▸ Transmission Control Protocol, Src Port: 57782, Dst Port: 8080, Seq: 4245, Ack: 24205, Len: 969
▾ Hypertext Transfer Protocol
  ▸ POST /login HTTP/1.1\r\n
    Host: localhost:8080\r\n
    User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0\r\n
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8\r\n
    Accept-Language: en-US,en;q=0.5\r\n
    Accept-Encoding: gzip, deflate\r\n
    Content-Type: application/x-www-form-urlencoded\r\n
  ▸ Content-Length: 22\r\n
    Referer: http://localhost:8080/login\r\n
    Origin: http://localhost:8080\r\n
    Connection: keep-alive\r\n
  ▸ [truncated]Cookie: wfu=MTYxODQ0MzIxMXxsVW9sV3M3VHNJZzRCTFpiNkpyZlU5ZjFMMFlIS084WE8taHhnenRUVHZrdTB4LU9sZ0NkSk5JZ1JWUE9kMHdKa1pYRG1iVnVKS0lkODJ3UWJ3WEZzOGF5eDYxMXRzZVJJM0djX3F0ZW5RNGZyWW9HSE1hZVdhYjVrYUJSRENkaHUwcmh0a3ZsWnh4NGpkODlFS3g5TS
    Upgrade-Insecure-Requests: 1\r\n
    \r\n
    [Full request URI: http://localhost:8080/login]
    [HTTP request 7/8]
    [Prev request in frame: 142153]
    [Response in frame: 142165]
    [Next request in frame: 142167]
    File Data: 22 bytes
▾ HTML Form URL Encoded: application/x-www-form-urlencoded
  ▾ Form item: "alias" = "admin"
      Key: alias
      Value: admin
  ▾ Form item: "pass" = "admin"
      Key: pass
      Value: admin
```

Fig 10: Auto-complete Enabled Information

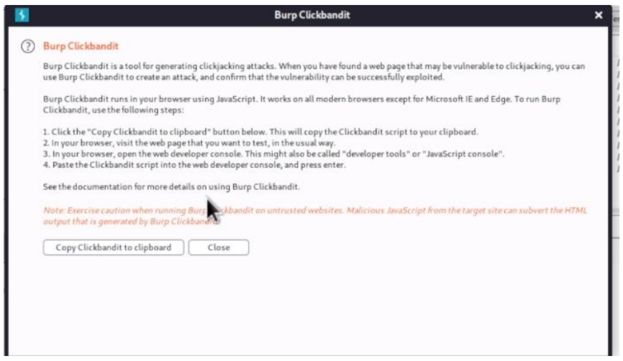


```
▸ Transmission Control Protocol, Src Port: 57782, Dst Port: 8080, Seq: 6335, Ack: 94797, Len: 963
▾ Hypertext Transfer Protocol
  ▸ POST /login HTTP/1.1\r\n
    Host: localhost:8080\r\n
    User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0\r\n
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8\r\n
    Accept-Language: en-US,en;q=0.5\r\n
    Accept-Encoding: gzip, deflate\r\n
    Content-Type: application/x-www-form-urlencoded\r\n
  ▸ Content-Length: 20\r\n
    Referer: http://localhost:8080/login\r\n
    Origin: http://localhost:8080\r\n
    Connection: keep-alive\r\n
  ▸ [truncated]Cookie: wfu=MTYxODQ0MzMwOXxHNUFUUVJVR0F4VG9UNkljOHE0ZHB6OHE0eTdBaUNpWnVjTWlWV21KYnM1SGYyR2pCTVhoMjNkNDlzWklRM2Y0Y1Z5WE83T3BkVTJNblFBWm1jZ2J4V19SSjl0c1g5Y2FiSnNIcm02Q2FBUmtKOWdZdVZHUHBGdkVfRnJabVdiV2RlRW1YTm9zRUktSXpqdHVyVy1Mcz
    Upgrade-Insecure-Requests: 1\r\n
    \r\n
    [Full request URI: http://localhost:8080/login]
    [HTTP request 10/10]
    [Prev request in frame: 144851]
    [Response in frame: 144865]
    File Data: 20 bytes
▾ HTML Form URL Encoded: application/x-www-form-urlencoded
  ▾ Form item: "alias" = "juan"
      Key: alias
      Value: juan
  ▾ Form item: "pass" = "juan"
      Key: pass
      Value: juan
```

Fig 11: New User Login

ii. Clickjacking using BURPSUITE:

- Open Burpsuite & Fire up Burp Clickbandit
- Copy the Script and paste it in the webpage by clicking View Inspection Element
- Go to console and type allow pasting
- Paste the Script



Burp Clickbandit

Burp Clickbandit

Burp Clickbandit is a tool for generating clickjacking attacks. When you have found a web page that may be vulnerable to clickjacking, you can use Burp Clickbandit to create an attack, and confirm that the vulnerability can be successfully exploited.

Burp Clickbandit runs in your browser using JavaScript. It works on all modern browsers except for Microsoft IE and Edge. To run Burp Clickbandit, use the following steps:

1. Click the "Copy Clickbandit to clipboard" button below. This will copy the Clickbandit script to your clipboard.
2. In your browser, visit the web page that you want to test, in the usual way.
3. In your browser, open the web developer console. This might also be called "developer tools" or "JavaScript console".
4. Paste the Clickbandit script into the web developer console, and press enter.

See the documentation for more details on using Burp Clickbandit.

Note: Exercise caution when running Burp Clickbandit on untrusted websites. Malicious JavaScript from the target site can subvert the HTML output that is generated by Burp Clickbandit.

Fig 12: Click Bandit Script

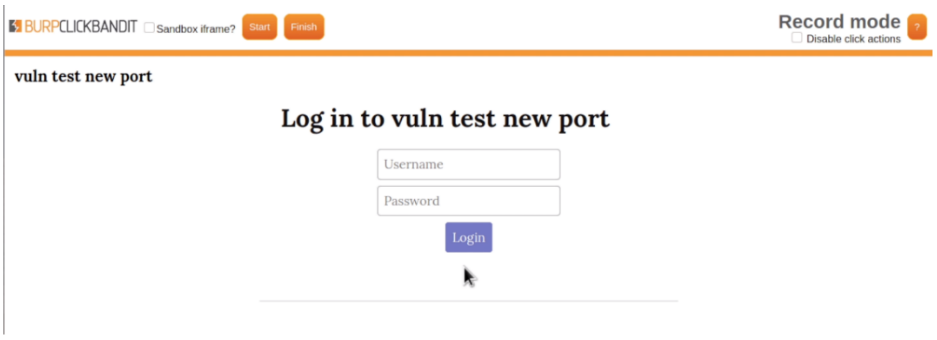


Fig 13: Normal Login Page

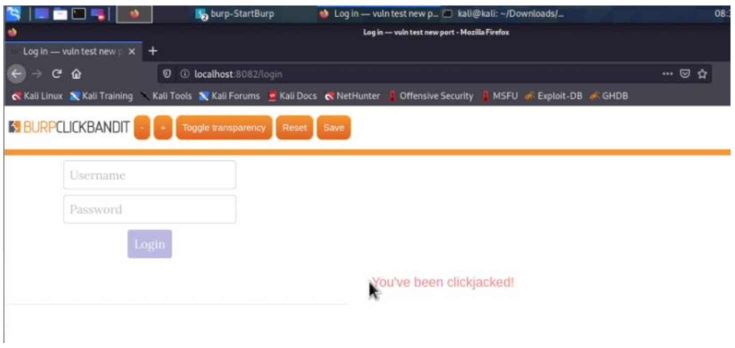
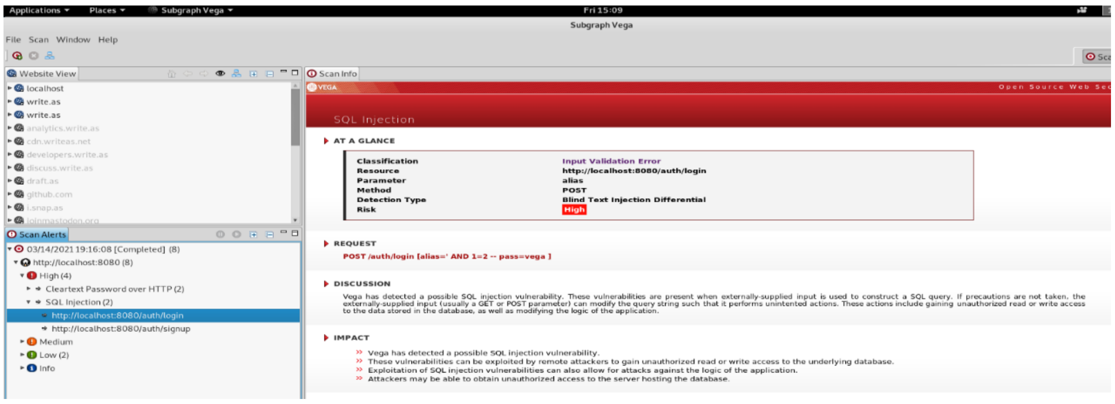


Fig 14: Clickjacking Exploitation

C. FALSE POSITIVES:

i. SQL injection(vega):

Under one scan , we found out that SQL injection can take place which is deemed higher risk.


Subgraph Vega
File Scan Window Help
Website View
localhost
write.as
Scan Alerts
03/14/2021 19:16:08 [Completed] (8)
http://localhost:8080 (8)
High (4)
Cleartext Password over HTTP (2)
SQL Injection (2)
http://localhost:8080/auth/login
http://localhost:8080/auth/signup
Medium
Low (2)
Info
Scan Info
VEGA

SQL Injection

AT A GLANCE

Classification	Input Validation Error
Resource	http://localhost:8080/auth/login
Parameter	alias
Method	POST
Detection Type	Blind Text Injection Differential
Risk	High

REQUEST

```
POST /auth/login [alias=' AND 1=2 -- pass=vega ]
```

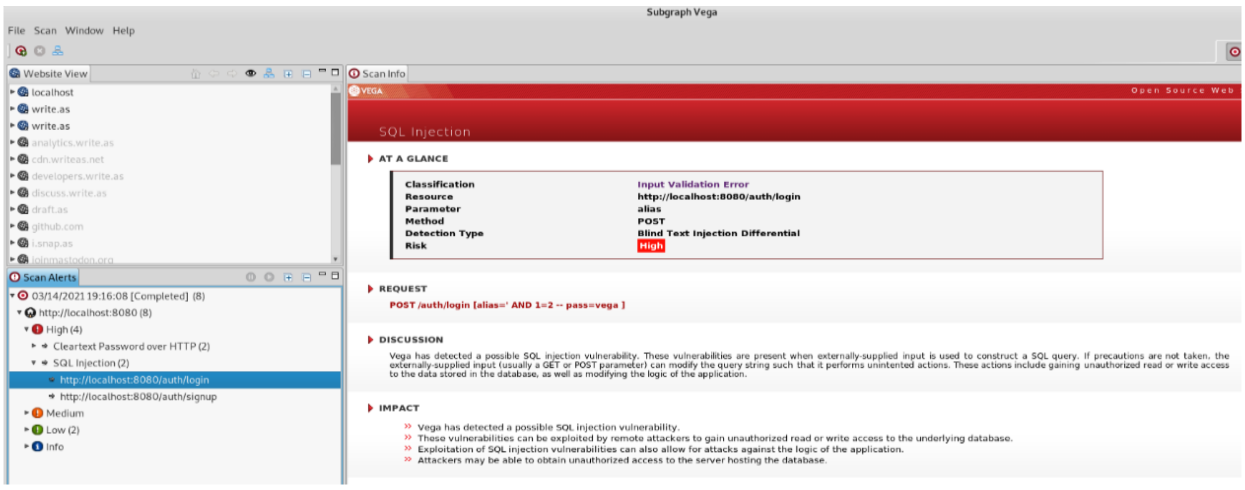
DISCUSSION

Vega has detected a possible SQL injection vulnerability. These vulnerabilities are present when externally-supplied input is used to construct a SQL query. If precautions are not taken, the externally-supplied input (usually a GET or POST parameter) can modify the query string such that it performs unintented actions. These actions include gaining unauthorized read or write access to the data stored in the database, as well as modifying the logic of the application.

IMPACT

» Vega has detected a possible SQL injection vulnerability.
» These vulnerabilities can be exploited by remote attackers to gain unauthorized read or write access to the underlying database.
» Exploitation of SQL injection vulnerabilities can also allow for attacks against the logic of the application.
» Attackers may be able to obtain unauthorized access to the server hosting the database.

Fig 15: Possible SQL Injection



SQL Injection

AT A GLANCE

Classification	Input Validation Error
Resource	http://localhost:8080/auth/login
Parameter	alias
Method	POST
Detection Type	Blind Text Injection Differential
Risk	High

REQUEST

POST /auth/login [alias=' AND 1=2 -- pass=vega]

DISCUSSION

Vega has detected a possible SQL injection vulnerability. These vulnerabilities are present when externally-supplied input is used to construct a SQL query. If precautions are not taken, the externally-supplied input (usually a GET or POST parameter) can modify the query string such that it performs unintented actions. These actions include gaining unauthorized read or write access to the data stored in the database, as well as modifying the logic of the application.

IMPACT

» Vega has detected a possible SQL injection vulnerability.
» These vulnerabilities can be exploited by remote attackers to gain unauthorized read or write access to the underlying database.
» Exploitation of SQL injection vulnerabilities can also allow for attacks against the logic of the application.
» Attackers may be able to obtain unauthorized access to the server hosting the database.

Fig 16: Possible SQL Injection while sign-up

ii. SQLMap Results:

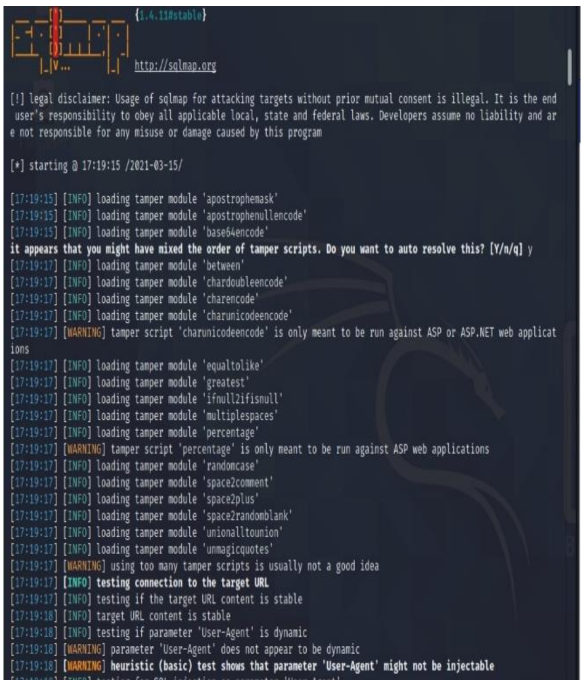

```
                        {1.4.11#stable}

                        http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end
 user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and ar
e not responsible for any misuse or damage caused by this program

[*] starting @ 17:19:15 /2021-03-15/

[17:19:15] [INFO] loading tamper module 'apostrophemask'
[17:19:15] [INFO] loading tamper module 'apostrophenullencode'
[17:19:15] [INFO] loading tamper module 'base64encode'
it appears that you might have mixed the order of tamper scripts. Do you want to auto resolve this? [Y/n/q] y
[17:19:17] [INFO] loading tamper module 'between'
[17:19:17] [INFO] loading tamper module 'chardoubleencode'
[17:19:17] [INFO] loading tamper module 'charencode'
[17:19:17] [INFO] loading tamper module 'charunicodeencode'
[17:19:17] [WARNING] tamper script 'charunicodeencode' is only meant to be run against ASP or ASP.NET web applicat
ions
[17:19:17] [INFO] loading tamper module 'equaltolike'
[17:19:17] [INFO] loading tamper module 'greatest'
[17:19:17] [INFO] loading tamper module 'ifnull2ifisnull'
[17:19:17] [INFO] loading tamper module 'multiplespaces'
[17:19:17] [INFO] loading tamper module 'percentage'
[17:19:17] [WARNING] tamper script 'percentage' is only meant to be run against ASP web applications
[17:19:17] [INFO] loading tamper module 'randomcase'
[17:19:17] [INFO] loading tamper module 'space2comment'
[17:19:17] [INFO] loading tamper module 'space2plus'
[17:19:17] [INFO] loading tamper module 'space2randomblank'
[17:19:17] [INFO] loading tamper module 'unionalltounion'
[17:19:17] [INFO] loading tamper module 'unmagicquotes'
[17:19:17] [WARNING] using too many tamper scripts is usually not a good idea
[17:19:17] [INFO] testing connection to the target URL
[17:19:17] [INFO] testing if the target URL content is stable
[17:19:18] [INFO] target URL content is stable
[17:19:18] [INFO] testing if parameter 'User-Agent' is dynamic
[17:19:18] [WARNING] parameter 'User-Agent' does not appear to be dynamic
[17:19:18] [WARNING] heuristic (basic) test shows that parameter 'User-Agent' might not be injectable
```

Fig 17: SQLMAP Initialization

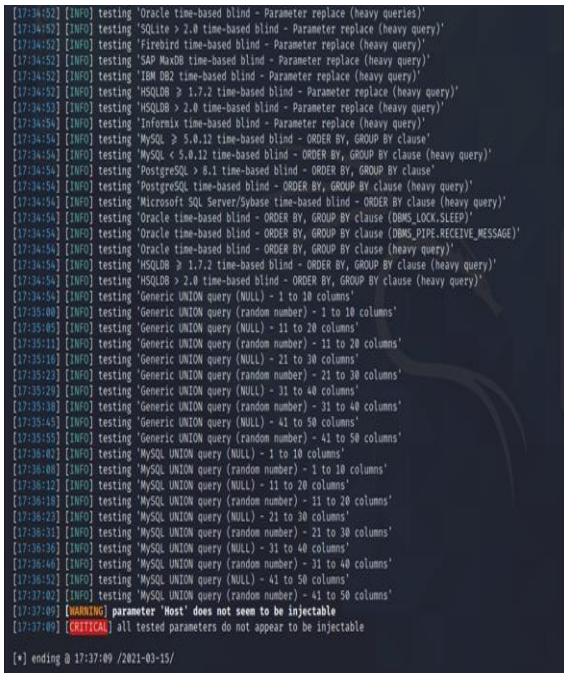

```
[17:34:52] [INFO] testing 'Oracle time-based blind - Parameter replace (heavy queries)'
[17:34:52] [INFO] testing 'SQLite > 2.0 time-based blind - Parameter replace (heavy query)'
[17:34:52] [INFO] testing 'Firebird time-based blind - Parameter replace (heavy query)'
[17:34:52] [INFO] testing 'SAP MaxDB time-based blind - Parameter replace (heavy query)'
[17:34:52] [INFO] testing 'IBM DB2 time-based blind - Parameter replace (heavy query)'
[17:34:52] [INFO] testing 'HSQLDB ≥ 1.7.2 time-based blind - Parameter replace (heavy query)'
[17:34:53] [INFO] testing 'HSQLDB > 2.0 time-based blind - Parameter replace (heavy query)'
[17:34:54] [INFO] testing 'Informix time-based blind - Parameter replace (heavy query)'
[17:34:54] [INFO] testing 'MySQL ≥ 5.0.12 time-based blind - ORDER BY, GROUP BY clause'
[17:34:54] [INFO] testing 'MySQL < 5.0.12 time-based blind - ORDER BY, GROUP BY clause (heavy query)'
[17:34:54] [INFO] testing 'PostgreSQL > 8.1 time-based blind - ORDER BY, GROUP BY clause'
[17:34:54] [INFO] testing 'PostgreSQL time-based blind - ORDER BY, GROUP BY clause (heavy query)'
[17:34:54] [INFO] testing 'Microsoft SQL Server/Sybase time-based blind - ORDER BY clause (heavy query)'
[17:34:54] [INFO] testing 'Oracle time-based blind - ORDER BY, GROUP BY clause (DBMS_LOCK.SLEEP)'
[17:34:54] [INFO] testing 'Oracle time-based blind - ORDER BY, GROUP BY clause (DBMS_PIPE.RECEIVE_MESSAGE)'
[17:34:54] [INFO] testing 'Oracle time-based blind - ORDER BY, GROUP BY clause (heavy query)'
[17:34:54] [INFO] testing 'HSQLDB ≥ 1.7.2 time-based blind - ORDER BY, GROUP BY clause (heavy query)'
[17:34:54] [INFO] testing 'HSQLDB > 2.0 time-based blind - ORDER BY, GROUP BY clause (heavy query)'
[17:34:54] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[17:35:00] [INFO] testing 'Generic UNION query (random number) - 1 to 10 columns'
[17:35:05] [INFO] testing 'Generic UNION query (NULL) - 11 to 20 columns'
[17:35:11] [INFO] testing 'Generic UNION query (random number) - 11 to 20 columns'
[17:35:16] [INFO] testing 'Generic UNION query (NULL) - 21 to 30 columns'
[17:35:23] [INFO] testing 'Generic UNION query (random number) - 21 to 30 columns'
[17:35:29] [INFO] testing 'Generic UNION query (NULL) - 31 to 40 columns'
[17:35:38] [INFO] testing 'Generic UNION query (random number) - 31 to 40 columns'
[17:35:45] [INFO] testing 'Generic UNION query (NULL) - 41 to 50 columns'
[17:35:55] [INFO] testing 'Generic UNION query (random number) - 41 to 50 columns'
[17:36:02] [INFO] testing 'MySQL UNION query (NULL) - 1 to 10 columns'
[17:36:08] [INFO] testing 'MySQL UNION query (random number) - 1 to 10 columns'
[17:36:12] [INFO] testing 'MySQL UNION query (NULL) - 11 to 20 columns'
[17:36:18] [INFO] testing 'MySQL UNION query (random number) - 11 to 20 columns'
[17:36:23] [INFO] testing 'MySQL UNION query (NULL) - 21 to 30 columns'
[17:36:31] [INFO] testing 'MySQL UNION query (random number) - 21 to 30 columns'
[17:36:36] [INFO] testing 'MySQL UNION query (NULL) - 31 to 40 columns'
[17:36:46] [INFO] testing 'MySQL UNION query (random number) - 31 to 40 columns'
[17:36:52] [INFO] testing 'MySQL UNION query (NULL) - 41 to 50 columns'
[17:37:02] [INFO] testing 'MySQL UNION query (random number) - 41 to 50 columns'
[17:37:09] [WARNING] parameter 'Host' does not seem to be injectable
[17:37:09] [CRITICAL] all tested parameters do not appear to be injectable

[*] ending @ 17:37:09 /2021-03-15/
```

Fig 18: SQLMAP output

Our goal was to use the automated tool named SQLMAP and use all the possible Attack vectors to exploit this lead. After using various tries and configurations, it was clear that, this was a false positive. We Also Tested for Cross Site Scripting which also turned out unsuccessful.

We tried many combinations of SQLMap configurations in order to tried to exploit, specific database, different query and parameter options in order to find a breach or at least extract some information regarding the registers in the database. Many clauses where tested such as UNION, SELECT, GROUP BY, HAVING, etc. But there was no favorable result.

V. LAWS, REGULATORY AND SECURITY POLICY

Attack	**Effect**	**Laws Applicable[9]**
CWE-319	Wilful Interception of Credentials	Section 184- Criminal Code of Canada Section 380(1) of the code- Hacking Section 402.2 of the code- Identity Fraud
CWE-1021	Data Exploitation (Entering Information to fields laid out by Hacker in UI) Possible Malware Infection	Section 430 of the code – Mischief Section 402.2 of the code- Identity Fraud

Table 4: Laws Applicable for Attempting these attacks.

Writefreely does not have any security policy of its own and asks the contributors to provide the policy[11] but it is licensed under GNU Affero General public license v3.0 [12]. But for iOS interacts with third-party services ("WriteFreely instances") run by a variety of providers, who can each set their own privacy policy. Users should review the privacy policy of any instance they choose to interact with via WriteFreely for iOS[13].

VI. COUNTERMEASURES/MITIGATION

A. *CWE-319: According to OWASP there are multiple possible ways of avoiding Sensitive data exposure[5] but main ways of doing it is as follows:*

- *Use of HTTPS instead of HTTP Channel to transmit data over network.*
- *Encrypt Data while in Transit, at rest.*
- *Disable Auto-complete & Caching over Sensitive Information*

For the localhost, one other way of securing data is by using digitally signed certificate and which can be generated using mkcert[10] , a github repository but it is recommended to use a long term certificate known to CA (Certificate Authority).

B. *CWE-1021: According to netsparker[4], websites need to have*

- *X-Frame-Options set as SAMEORIGIN or*
- *X-Frame-Options set as DENY*

Another way of doing it on local network is by use of css file where you can increase the length of Iframe, so that you know whether there is some another layer embedded or not.

VII. CONCLUSION:

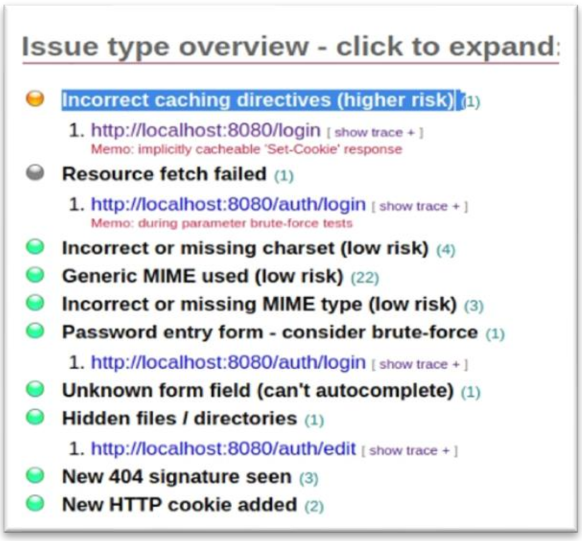
'WriteFreely' an independent open-source blogging platform know to not collect much information has provided a new way of writing posts. Although, the web application has tried to negate the possibility of high-risk attacks such as XSS, SQL Injection but there was a major issue of sensitive data exposure which can be exploited by elite hackers to perform various activities, In addition, there were same medium to low risk vulnerabilities as well. Hence, we suggest that use of HTTPS and other mitigation techniques is must to avoid Vulnerability attacks. In regard to Code smells, A generic conception is that programmer should try to minimize complex coding and remove unnecessary variables which have not been used while always trying to provide comments and updating the code.

VIII. APPENDIX

Other High and Médium risk vulnerabilities found but not exploited are listed here:

Vulnerability	Risk	Found By
CWE-524 [8] -Incorrect Caching Directive	High	SkipFish
XML Injection	Medium	Vega Scanner

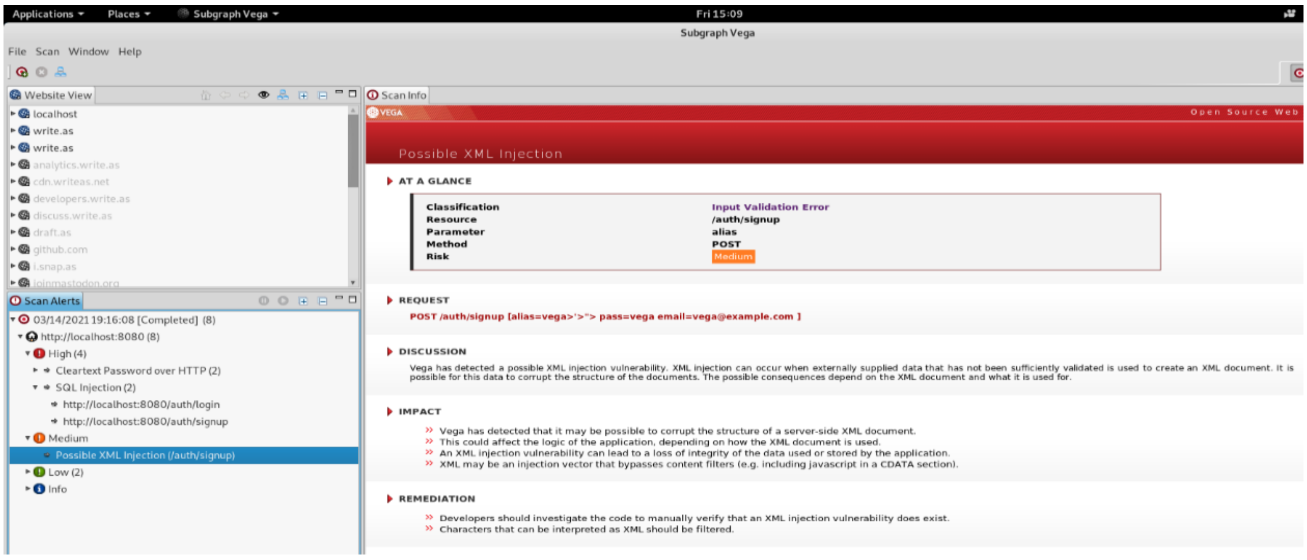
Table 5: Other Vulnerabilities Reported



Issue type overview - click to expand:

- Incorrect caching directives (higher risk) (1)
  1. http://localhost:8080/login [show trace +]
     Memo: implicitly cacheable 'Set-Cookie' response
- Resource fetch failed (1)
  1. http://localhost:8080/auth/login [show trace +]
     Memo: during parameter brute-force tests
- Incorrect or missing charset (low risk) (4)
- Generic MIME used (low risk) (22)
- Incorrect or missing MIME type (low risk) (3)
- Password entry form - consider brute-force (1)
  1. http://localhost:8080/auth/login [show trace +]
- Unknown form field (can't autocomplete) (1)
- Hidden files / directories (1)
  1. http://localhost:8080/auth/edit [show trace +]
- New 404 signature seen (3)
- New HTTP cookie added (2)

Fig. 19 Skipfish report



Website View:
- localhost
- write.as
- write.as
- analytics.write.as
- cdn.writeas.net
- developers.write.as
- discuss.write.as
- draft.as
- github.com
- i.snap.as
- joinmastodon.org

Scan Alerts:
- 03/14/2021 19:16:08 [Completed] (8)
  - http://localhost:8080 (8)
    - High (4)
      - Cleartext Password over HTTP (2)
      - SQL Injection (2)
        - http://localhost:8080/auth/login
        - http://localhost:8080/auth/signup
    - Medium
      - Possible XML Injection (/auth/signup)
    - Low (2)
    - Info

Possible XML Injection

AT A GLANCE

Classification	Input Validation Error
Resource	/auth/signup
Parameter	alias
Method	POST
Risk	Medium

REQUEST

POST /auth/signup [alias=vega>'>"> pass=vega email=vega@example.com]

DISCUSSION

Vega has detected a possible XML injection vulnerability. XML injection can occur when externally supplied data that has not been sufficiently validated is used to create an XML document. It is possible for this data to corrupt the structure of the documents. The possible consequences depend on the XML document and what it is used for.

IMPACT

» Vega has detected that it may be possible to corrupt the structure of a server-side XML document.
» This could affect the logic of the application, depending on how the XML document is used.
» An XML injection vulnerability can lead to a loss of integrity of the data used or stored by the application.
» XML may be an injection vector that bypasses content filters (e.g. including javascript in a CDATA section).

REMEDIATION

» Developers should investigate the code to manually verify that an XML injection vulnerability does exist.
» Characters that can be interpreted as XML should be filtered.

Fig 20: XML Injection

IX. REFERENCES

[1] "OWASP Top Ten Web Application Security Risks | OWASP". Owasp.Org, 2021, https://owasp.org/www-project-top-ten/.

[2] "OWASP ZAP". Zaproxy.Org, 2021, https://www.zaproxy.org/docs/alerts/10020-1/.

[3] Kazi Zakia Sultana, Zadia Codabux, and Byron Williams, "Examining the Relationship of Code and Architectural Smells with Software Vulnerabilities", arXiv:2010.15978 [cs.SE]

[4]"Missing X-Frame-Options Header | Netsparker". Netsparker.Com, 2021, https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/missing-x-frame-options-header/.

[5] Owasp.Org, 2021, https://owasp.org/www-pdf-archive/OWASP_Top_10_-_2013.pdf.

[6] "CWE - CWE-319: Cleartext Transmission Of Sensitive Information (4.4)". Cwe.Mitre.Org, 2021, https://cwe.mitre.org/data

[7] "CWE - CWE-1021: Improper Restriction Of Rendered UI Layers Or Frames (4.4)". Cwe.Mitre.Org, 2021, https://cwe.mitre.org/data/definitions/1021.html

[8] "CWE - CWE-524: Use Of Cache Containing Sensitive Information (4.4)". Cwe.Mitre.Org, 2021, https://cwe.mitre.org/data/definitions/524.html

[9] Group, Global. "Cybersecurity 2021 | Laws And Regulations | Canada | ICLG". International Comparative Legal Guides International Business Reports, 2021, https://iclg.com/practice-areas/cybersecurity-laws-and-regulations/canada

[10] "Filosottile/Mkcert". *Github*, 2021, https://github.com/FiloSottile/mkcert.

[11] "Build Software Better, Together". *Github*, 2021, https://github.com/writefreely/writefreely-swift/security.

[12]"Writefreely/Writefreely". *Github*,2021,https://github.com/writefreely/writefreely/blob/develop/LICENSE.

[13]"Privacy Policy — Writefreely For Ios". *Writefreely.Org*, 2021, https://writefreely.org/apps/ios/privacy.

[14]Center, Support. "Burp Suite Documentation - Contents". Portswigger.Net, 2021, https://portswigger.net/burp/documentation/contents. Accessed 24 Apr 2021.

[15]"Documentation". Vega, 2021, https://vega.github.io/vega/docs/. Accessed 24 Apr 2021.

[16] “Skipfish – Documentation”. Tools.Kali.Org, 2021, https://tools.kali.org/web-applications/skipfish. Accessed 24 Apr 2021

[17] "Wireshark · Documentation". Wireshark.Org, 2021, https://www.wireshark.org/docs/. Accessed 24 Apr 2021

[18] "Oracle VM Virtualbox". Virtualbox.Org, 2021, https://www.virtualbox.org/manual/UserManual.html. Accessed 24 Apr 2021.

[19]Tools.Kali.Org, 2021, https://tools.kali.org/information-gathering/nikto. Accessed 24 Apr 2021.

[20]Sonarqube Documentation | Sonarqube Docs". Docs.Sonarqube.Org, 2021, https://docs.sonarqube.org/latest/. Accessed 24 Apr 2021.

[21]OWASP ZAP – Documentation". Zaproxy.Org, 2021, https://www.zaproxy.org/docs/. Accessed 24 Apr 2021.

[22]Sqlmap: Automatic SQL Injection And Database Takeover Tool". Sqlmap.Org, 2021, https://sqlmap.org/. Accessed 24 Apr 2021.