Future Interns

Cyber Security

TASK – 01

Vulnerability Assessment Report for a Live Website

Intern Name – Arra Vaishnavi Reddy

Internship Details

Internship Domain: Cyber Security

Organization: Future Interns

Task Name: Vulnerability
Assessment
Report for a Live Website

Tool Used: OWASP ZAP

Target Website:
http://testphp.vulnweb.com

Assessment Type: Passive Scan

OBJECTIVE

The objective of this task is to perform a basic security assessment on a vulnerable web application using OWASP ZAP and identify common security misconfigurations and vulnerabilities .

About the Target Website

About the target website testphp.vulnweb.com is an intentionally vulnerable web application designed for security testing and learning purposes. It allows users to safely understand how vulnerabilities are detected without affecting real systems.

Tool Used: OWASP ZAP

OWASP ZAP (Zed Attack Proxy) is an open-source web application security testing tool used to find vulnerabilities in web applications during development and testing phases.

Scanning Methodology

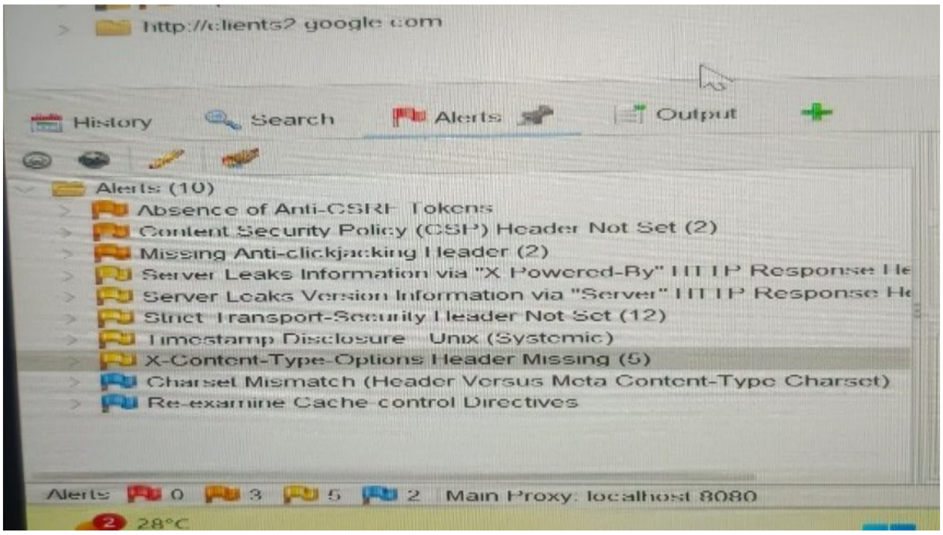
- OWASP ZAP was launched in Standard Mode
- The target URL was accessed through the ZAP-controlled browser
- A passive scan was performed
- Alerts generated by ZAP were analyzed and documented

Summary of Alerts Detected

- A total of 10 security alerts were detected during the assessment.
- These alerts were categorized based on their risk levels:

1. (Orange) Medium Risk Alerts: 3
2. (Yellow) Low Risk Alerts: 5
3. (Blue) Informational Alerts: 2

As you can see the Fig – 01 it clearly shows the 10 Alerts.



```
Alerts (10)
  Absence of Anti-CSRF Tokens
  Content Security Policy (CSP) Header Not Set (2)
  Missing Anti-clickjacking Header (2)
  Server Leaks Information via "X Powered-By" HTTP Response He
  Server Leaks Version Information via "Server" HTTP Response He
  Strict Transport-Security Header Not Set (12)
  Timestamp Disclosure Unix (Systemic)
  X-Content-Type-Options Header Missing (5)
  Charset Mismatch (Header Versus Meta Content-Type Charset)
  Re-examine Cache-control Directives

Alerts 0 3 5 2 Main Proxy: localhost 8080
```

Fig -01

Selected Alerts for Analysis

Alert 1: Absence of Anti-CSRF Tokens

Risk Level: Medium

Description:

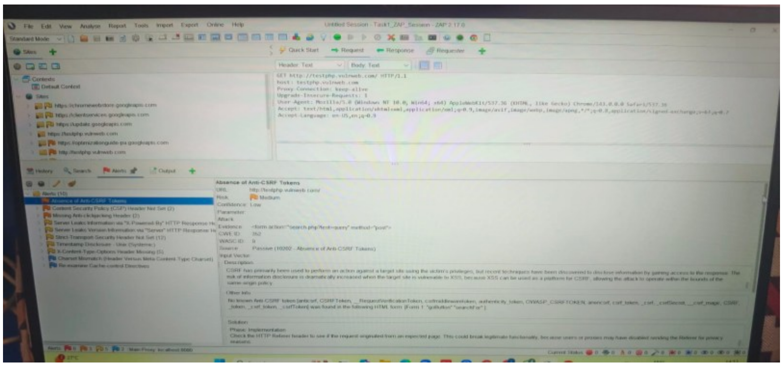
The application does not implement Anti-CSRF tokens, making it vulnerable to Cross-Site Request Forgery attacks.

Impact:

Attackers may perform unauthorized actions on behalf of authenticated users.

Recommendation:

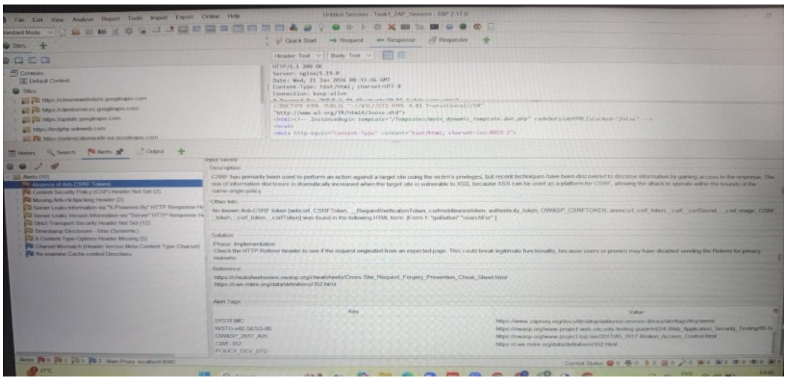
Implement Anti-CSRF tokens for all sensitive requests.



```
GET http://testphp.vulnweb.com/ HTTP/1.1
host: testphp.vulnweb.com
Proxy-Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Accept-Language: en-US,en;q=0.9
```

Absence of Anti-CSRF Tokens
URL: http://testphp.vulnweb.com/
Risk: Medium
Confidence: Low
Evidence: <form action="search.php?test=query" method="post">
CWE ID: 352
WASC ID: 9
Source: Passive (10202 - Absence of Anti-CSRF Tokens)

Fig - 02



```
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Wed, 21 Jan 2026 08:33:46 GMT
Content-Type: text/html; charset=UTF-8
Connection: keep-alive
```

Description
CSRF has primarily been used to perform an action against a target site using the victim's privileges, but recent techniques have been discovered to disclose information by gaining access to the response. The risk of information disclosure is dramatically increased when the target site is vulnerable to XSS, because XSS can be used as a platform for CSRF, allowing the attack to operate within the bounds of the same-origin policy.

Solution
Phase: Implementation
Check the HTTP Referer header to see if the request originated from an expected page. This could break legitimate functionality, because users or proxies may have disabled sending the Referer for privacy reasons.

Reference
https://cheatsheetseries.owasp.org/cheatsheets/Cross-Site_Request_Forgery_Prevention_Cheat_Sheet.html
https://cwe.mitre.org/data/definitions/352.html

Fig -03

Alert 2: Content Security Policy (CSP) Header Not Set

Risk Level: Medium

Description:

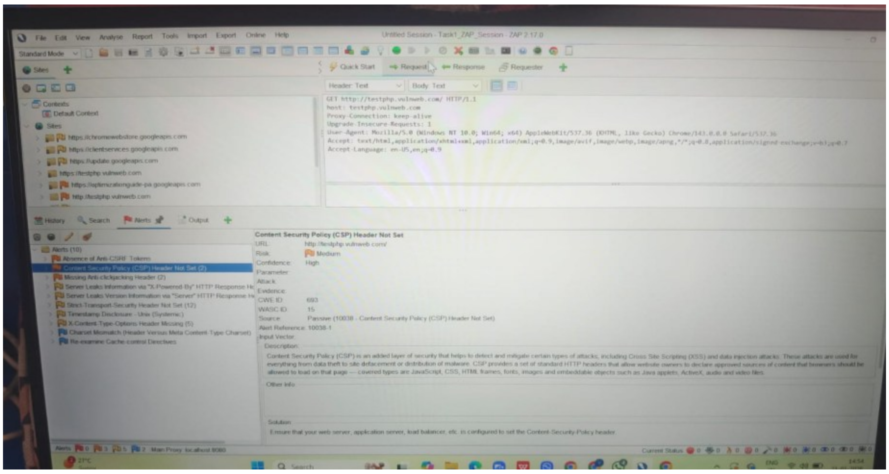
The Content Security Policy header is missing, increasing the risk of XSS attacks.

Impact:

Malicious scripts can be injected and executed in the user’s browser.

Recommendation:

Configure and enforce a strict Content Security Policy header .


File Edit View Analyse Report Tools Import Export Online Help
Untitled Session - Task1_ZAP_Session - ZAP 2.17.0
Standard Mode
Sites
Quick Start
Request
Response
Requester
Header: Text
Body: Text
Contexts
Default Context
Sites
https://chromewebstore.googleapis.com
https://clientservices.googleapis.com
https://update.googleapis.com
https://testphp.vulnweb.com
https://optimizationguide-pa.googleapis.com
http://testphp.vulnweb.com
History
Search
Alerts
Output
Alerts (10)
Absence of Anti-CSRF Tokens
Content Security Policy (CSP) Header Not Set (2)
Missing Anti-clickjacking Header (2)
Server Leaks Information via "X-Powered-By" HTTP Response He
Server Leaks Version Information via "Server" HTTP Response He
Strict-Transport-Security Header Not Set (12)
Timestamp Disclosure - Unix (Systemic)
X-Content-Type-Options Header Missing (5)
Charset Mismatch (Header Versus Meta Content-Type Charset)
Re-examine Cache-control Directives
Alerts 0 3 5 2 Main Proxy: localhost:8080
Current Status 0 0 0 0 0 0 0 0 0 0
27°C
Search
ENG
14:54

```
GET http://testphp.vulnweb.com/ HTTP/1.1
host: testphp.vulnweb.com
Proxy-Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Accept-Language: en-US,en;q=0.9
```

Content Security Policy (CSP) Header Not Set

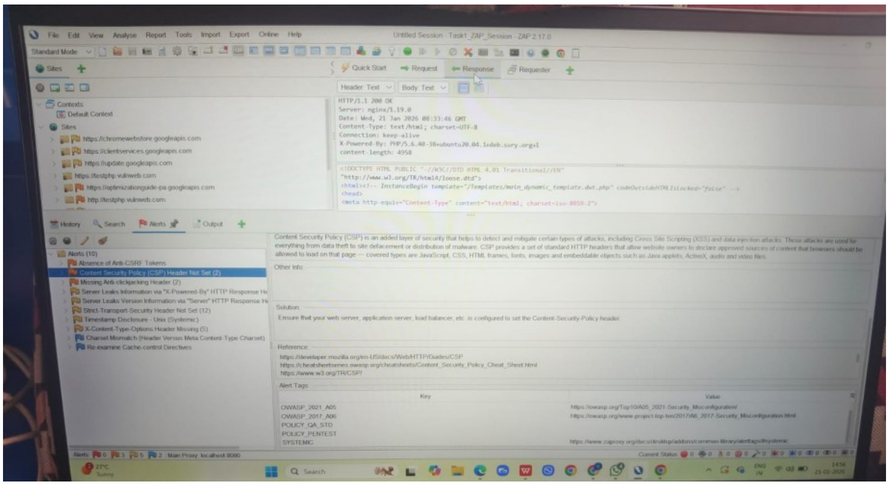
URL: http://testphp.vulnweb.com/
Risk: Medium
Confidence: High
Parameter:
Attack:
Evidence:
CWE ID: 693
WASC ID: 15
Source: Passive (10038 - Content Security Policy (CSP) Header Not Set)
Alert Reference: 10038-1
Input Vector:

Description:
Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks, including Cross Site Scripting (XSS) and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources of content that browsers should be allowed to load on that page — covered types are JavaScript, CSS, HTML frames, fonts, images and embeddable objects such as Java applets, ActiveX, audio and video files.

Other Info:

Solution:
Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.

Fig-04


File Edit View Analyse Report Tools Import Export Online Help
Untitled Session - Task1_ZAP_Session - ZAP 2.17.0
Standard Mode
Sites
Quick Start
Request
Response
Requester
Header: Text
Body: Text
Contexts
Default Context
Sites
https://chromewebstore.googleapis.com
https://clientservices.googleapis.com
https://update.googleapis.com
https://testphp.vulnweb.com
https://optimizationguide-pa.googleapis.com
http://testphp.vulnweb.com
History
Search
Alerts
Output
Alerts (10)
Absence of Anti-CSRF Tokens
Content Security Policy (CSP) Header Not Set (2)
Missing Anti-clickjacking Header (2)
Server Leaks Information via "X-Powered-By" HTTP Response He
Server Leaks Version Information via "Server" HTTP Response He
Strict-Transport-Security Header Not Set (12)
Timestamp Disclosure - Unix (Systemic)
X-Content-Type-Options Header Missing (5)
Charset Mismatch (Header Versus Meta Content-Type Charset)
Re-examine Cache-control Directives
Alerts 0 3 5 2 Main Proxy: localhost:8080
Current Status 0 0 0 0 0 0 0 0 0 0
27°C Sunny
Search
ENG IN
14:56
21-01-2026

```
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Wed, 21 Jan 2026 08:33:46 GMT
Content-Type: text/html; charset=UTF-8
Connection: keep-alive
X-Powered-By: PHP/5.6.40-38+ubuntu20.04.1+deb.sury.org+1
content-length: 4958
```

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html><!-- InstanceBegin template="/Templates/main_dynamic_template.dwt.php" codeOutsideHTMLIsLocked="false" -->
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-2">
```

Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks, including Cross Site Scripting (XSS) and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources of content that browsers should be allowed to load on that page — covered types are JavaScript, CSS, HTML frames, fonts, images and embeddable objects such as Java applets, ActiveX, audio and video files.

Other Info:

Solution:
Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.

Reference:
https://developer.mozilla.org/en-US/docs/Web/HTTP/Guides/CSP
https://cheatsheetseries.owasp.org/cheatsheets/Content_Security_Policy_Cheat_Sheet.html
https://www.w3.org/TR/CSP/

Alert Tags:

Key	Value
OWASP_2021_A05	https://owasp.org/Top10/A05_2021-Security_Misconfiguration/
OWASP_2017_A06	https://owasp.org/www-project-top-ten/2017/A6_2017-Security_Misconfiguration.html
POLICY_QA_STD	
POLICY_PENTEST	
SYSTEMIC	https://www.zaproxy.org/docs/desktop/addons/common-library/alerttags/#systemic

Fig-05

Alert 3: Missing Anti-Clickjacking Header

Risk Level: Medium

Description:

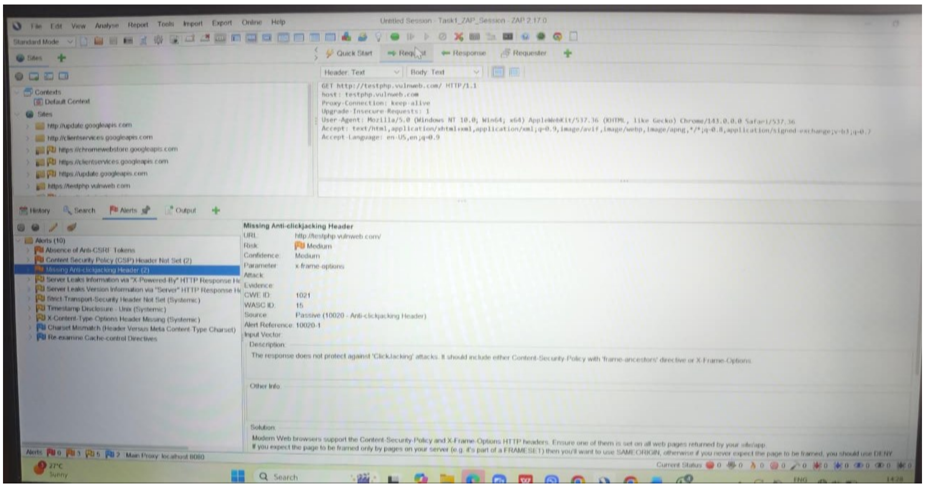
The application does not set the X-Frame-Options header.

Impact:

The website may be embedded into malicious iframes leading to clickjacking attacks.

Recommendation:

Set X-Frame-Options or use CSP frame-ancestors directive.


File Edit View Analyse Report Tools Import Export Online Help
Untitled Session - Task1_ZAP_Session - ZAP 2.17.0
Standard Mode
Sites
Quick Start
Request
Response
Requester
Header: Text
Body: Text
History
Search
Alerts
Output
Search

```
GET http://testphp.vulnweb.com/ HTTP/1.1
host: testphp.vulnweb.com
Proxy-Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Accept-Language: en-US,en;q=0.9
```

Contexts
Default Context
Sites
http://update.googleapis.com
http://clientservices.googleapis.com
https://chromewebstore.googleapis.com
https://clientservices.googleapis.com
https://update.googleapis.com
https://testphp.vulnweb.com

Alerts (10)
- Absence of Anti-CSRF Tokens
- Content Security Policy (CSP) Header Not Set (2)
- Missing Anti-clickjacking Header (2)
- Server Leaks Information via "X-Powered-By" HTTP Response He
- Server Leaks Version Information via "Server" HTTP Response He
- Strict-Transport-Security Header Not Set (Systemic)
- Timestamp Disclosure - Unix (Systemic)
- X-Content-Type-Options Header Missing (Systemic)
- Charset Mismatch (Header Versus Meta Content-Type Charset)
- Re-examine Cache-control Directives

Missing Anti-clickjacking Header
URL: http://testphp.vulnweb.com/
Risk: Medium
Confidence: Medium
Parameter: x-frame-options
Attack:
Evidence:
CWE ID: 1021
WASC ID: 15
Source: Passive (10020 - Anti-clickjacking Header)
Alert Reference: 10020-1
Input Vector:

Description

The response does not protect against 'ClickJacking' attacks. It should include either Content-Security-Policy with 'frame-ancestors' directive or X-Frame-Options.

Other Info

Solution

Modern Web browsers support the Content-Security-Policy and X-Frame-Options HTTP headers. Ensure one of them is set on all web pages returned by your site/app.
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, otherwise if you never expect the page to be framed, you should use DENY

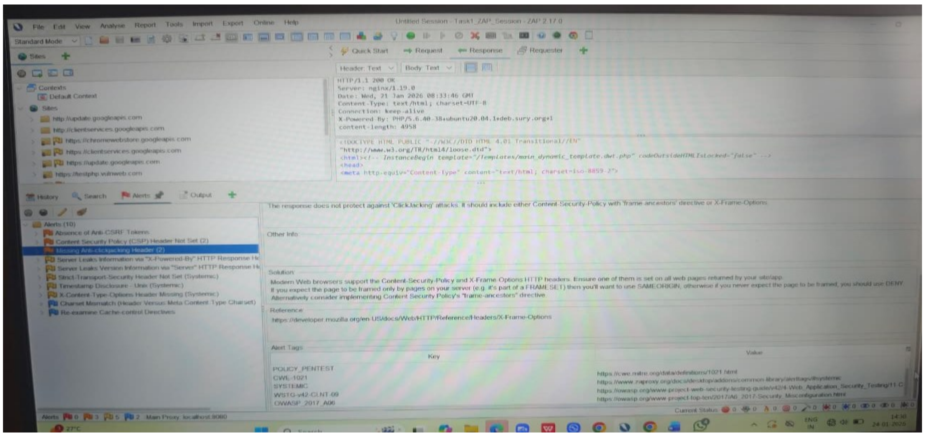
Alerts 0 3 5 2 Main Proxy: localhost:8080

Current Status 0 0 0 0 0 0 0 0 0 0

27°C
Sunny

14:28

Fig -06


File Edit View Analyse Report Tools Import Export Online Help
Untitled Session - Task1_ZAP_Session - ZAP 2.17.0
Standard Mode
Sites
Quick Start
Request
Response
Requester
Header: Text
Body: Text
History
Search
Alerts
Output
Search

```
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Wed, 21 Jan 2026 08:33:46 GMT
Content-Type: text/html; charset=UTF-8
Connection: keep-alive
X-Powered-By: PHP/5.6.40-38+ubuntu20.04.1+deb.sury.org+1
content-length: 4958
```

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html><!-- InstanceBegin template="/Templates/main_dynamic_template.dwt.php" codeOutsideHTMLIsLocked="false" -->
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-2">
```

Contexts
Default Context
Sites
http://update.googleapis.com
http://clientservices.googleapis.com
https://chromewebstore.googleapis.com
https://clientservices.googleapis.com
https://update.googleapis.com
https://testphp.vulnweb.com

Alerts (10)
- Absence of Anti-CSRF Tokens
- Content Security Policy (CSP) Header Not Set (2)
- Missing Anti-clickjacking Header (2)
- Server Leaks Information via "X-Powered-By" HTTP Response He
- Server Leaks Version Information via "Server" HTTP Response He
- Strict-Transport-Security Header Not Set (Systemic)
- Timestamp Disclosure - Unix (Systemic)
- X-Content-Type-Options Header Missing (Systemic)
- Charset Mismatch (Header Versus Meta Content-Type Charset)
- Re-examine Cache-control Directives

The response does not protect against 'ClickJacking' attacks. It should include either Content-Security-Policy with 'frame-ancestors' directive or X-Frame-Options.

Other Info

Solution

Modern Web browsers support the Content-Security-Policy and X-Frame-Options HTTP headers. Ensure one of them is set on all web pages returned by your site/app.
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, otherwise if you never expect the page to be framed, you should use DENY.
Alternatively consider implementing Content Security Policy's "frame-ancestors" directive.

Reference

https://developer.mozilla.org/en-US/docs/Web/HTTP/Reference/Headers/X-Frame-Options

Alert Tags

Key	Value
POLICY_PENTEST	
CWE-1021	https://cwe.mitre.org/data/definitions/1021.html
SYSTEMIC	https://www.zaproxy.org/docs/desktop/addons/common-library/alerttags/#systemic
WSTG-v42-CLNT-09	https://owasp.org/www-project-web-security-testing-guide/v42/4-Web_Application_Security_Testing/11-C
OWASP_2017_A06	https://owasp.org/www-project-top-ten/2017/A6_2017-Security_Misconfiguration.html

Alerts 0 3 5 2 Main Proxy: localhost:8080

Current Status 0 0 0 0 0 0 0 0 0 0

27°C

ENG IN 14:30 24-01-2026

Fig-07

Alert 4: X-Content-Type-Options Header Missing

Risk Level: Low

Description:

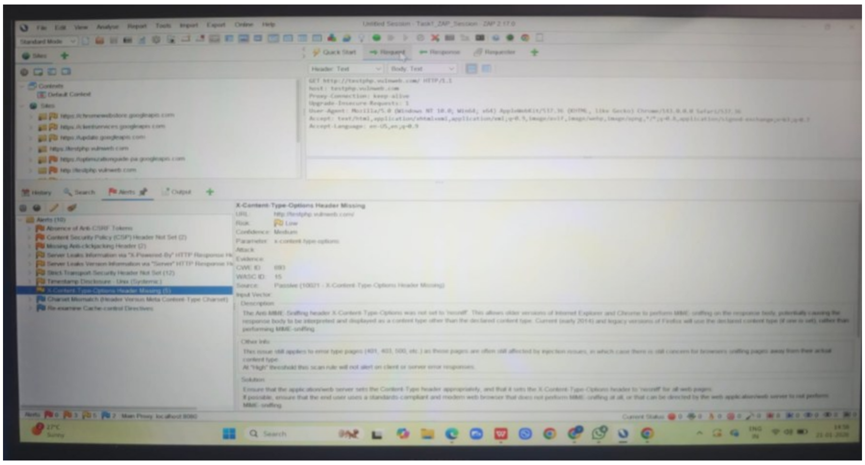
The X-Content-Type-Options header is not set to “nosniff”.

Impact:

Browsers may incorrectly interpret content types, increasing XSS risks.

Recommendation:

Set X-Content-Type-Options header to “nosniff”.



```
GET http://testphp.vulnweb.com/ HTTP/1.1
Host: testphp.vulnweb.com
Proxy-Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Accept-Language: en-US,en;q=0.9
```

Contexts
Default Context
Sites
https://chromewebstore.googleapis.com
https://clientservices.googleapis.com
https://update.googleapis.com
https://testphp.vulnweb.com
https://optimizationguide-pa.googleapis.com
http://testphp.vulnweb.com

Alerts (10)
- Absence of Anti-CSRF Tokens
- Content Security Policy (CSP) Header Not Set (2)
- Missing Anti-clickjacking Header (2)
- Server Leaks Information via "X-Powered-By" HTTP Response He
- Server Leaks Version Information via "Server" HTTP Response He
- Strict-Transport-Security Header Not Set (12)
- Timestamp Disclosure - Unix (Systemic)
- X-Content-Type-Options Header Missing (5)
- Charset Mismatch (Header Versus Meta Content-Type Charset)
- Re-examine Cache-control Directives

X-Content-Type-Options Header Missing
URL: http://testphp.vulnweb.com/
Risk: Low
Confidence: Medium
Parameter: x-content-type-options
Attack:
Evidence:
CWE ID: 693
WASC ID: 15
Source: Passive (10021 - X-Content-Type-Options Header Missing)
Input Vector:

Description
The Anti-MIME-Sniffing header X-Content-Type-Options was not set to 'nosniff'. This allows older versions of Internet Explorer and Chrome to perform MIME-sniffing on the response body, potentially causing the response body to be interpreted and displayed as a content type other than the declared content type. Current (early 2014) and legacy versions of Firefox will use the declared content type (if one is set), rather than performing MIME-sniffing.

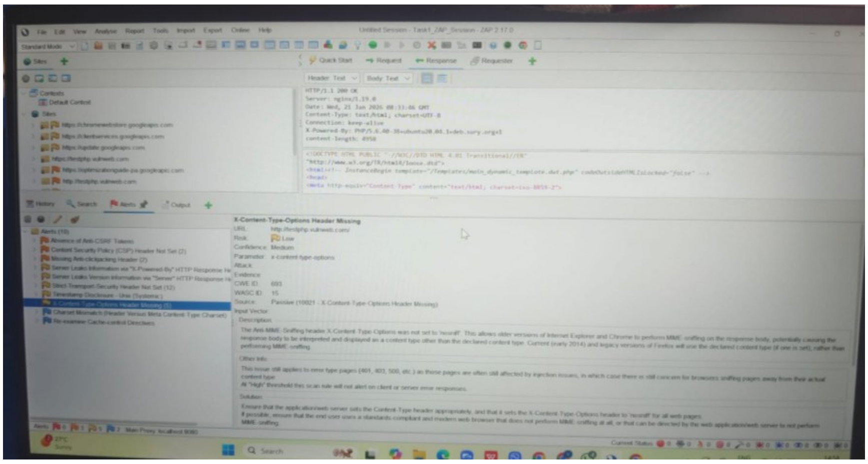
Other Info
This issue still applies to error type pages (401, 403, 500, etc.) as those pages are often still affected by injection issues, in which case there is still concern for browsers sniffing pages away from their actual content type.
At "High" threshold this scan rule will not alert on client or server error responses.

Solution
Ensure that the application/web server sets the Content-Type header appropriately, and that it sets the X-Content-Type-Options header to 'nosniff' for all web pages.
If possible, ensure that the end user uses a standards-compliant and modern web browser that does not perform MIME-sniffing at all, or that can be directed by the web application/web server to not perform MIME-sniffing.

Alerts 0 3 5 2 Main Proxy: localhost:8080

Current Status 0 0 0 0 0 0 0 0 0 0

Fig -08



```
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Wed, 21 Jan 2026 08:31:06 GMT
Content-Type: text/html; charset=UTF-8
Connection: keep-alive
X-Powered-By: PHP/5.6.40-38+ubuntu20.04.1+deb.sury.org+1
content-length: 4958
```

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html><!-- InstanceBegin template="/Templates/main_dynamic_template.dwt.php" codeOutsideHTMLIsLocked="false" -->
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-2">
```

X-Content-Type-Options Header Missing
URL: http://testphp.vulnweb.com/
Risk: Low
Confidence: Medium
Parameter: x-content-type-options
CWE ID: 693
WASC ID: 15
Source: Passive (10021 - X-Content-Type-Options Header Missing)

Fig -09

Alert 5: Strict-Transport-Security Header Not Set

Risk Level: Low

Description:

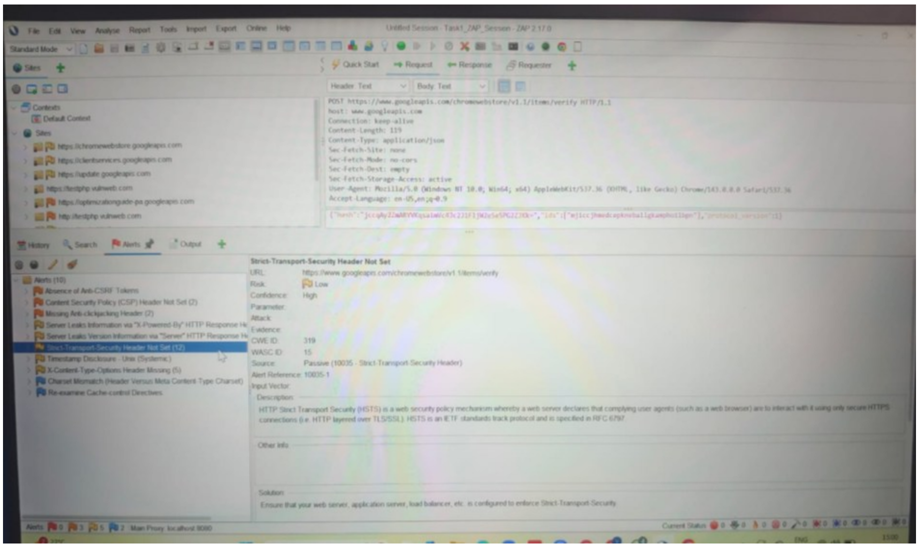
The application does not enforce HTTPS using HSTS.

Impact:

Users may be exposed to man-in-the-middle attacks.

Recommendation:

Enable Strict-Transport-Security header to enforce HTTPS connections.



- Contexts
  - Default Context
- Sites
  - https://chromewebstore.googleapis.com
  - https://clientservices.googleapis.com
  - https://update.googleapis.com
  - https://testphp.vulnweb.com
  - https://optimizationguide-pa.googleapis.com
  - http://testphp.vulnweb.com

```
POST https://www.googleapis.com/chromewebstore/v1.1/items/verify HTTP/1.1
host: www.googleapis.com
Connection: keep-alive
Content-Length: 119
Content-Type: application/json
Sec-Fetch-Site: none
Sec-Fetch-Mode: no-cors
Sec-Fetch-Dest: empty
Sec-Fetch-Storage-Access: active
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36
Accept-Language: en-US,en;q=0.9

{"hash":"jccqAy2ZwAKYVKqsaIaWc43c221F1jW2v5e5PG2Z2Kk=","ids":["mjiccjhmedcepknebaligkamphoilbpn"],"protocol_version":1}
```

- Alerts (10)
  - Absence of Anti-CSRF Tokens
  - Content Security Policy (CSP) Header Not Set (2)
  - Missing Anti-clickjacking Header (2)
  - Server Leaks Information via "X-Powered-By" HTTP Response He
  - Server Leaks Version Information via "Server" HTTP Response He
  - Strict-Transport-Security Header Not Set (12)
  - Timestamp Disclosure - Unix (Systemic)
  - X-Content-Type-Options Header Missing (5)
  - Charset Mismatch (Header Versus Meta Content-Type Charset)
  - Re-examine Cache-control Directives

Strict-Transport-Security Header Not Set

URL: https://www.googleapis.com/chromewebstore/v1.1/items/verify
Risk: Low
Confidence: High
Parameter:
Attack:
Evidence:
CWE ID: 319
WASC ID: 15
Source: Passive (10035 - Strict-Transport-Security Header)
Alert Reference: 10035-1
Input Vector:

Description

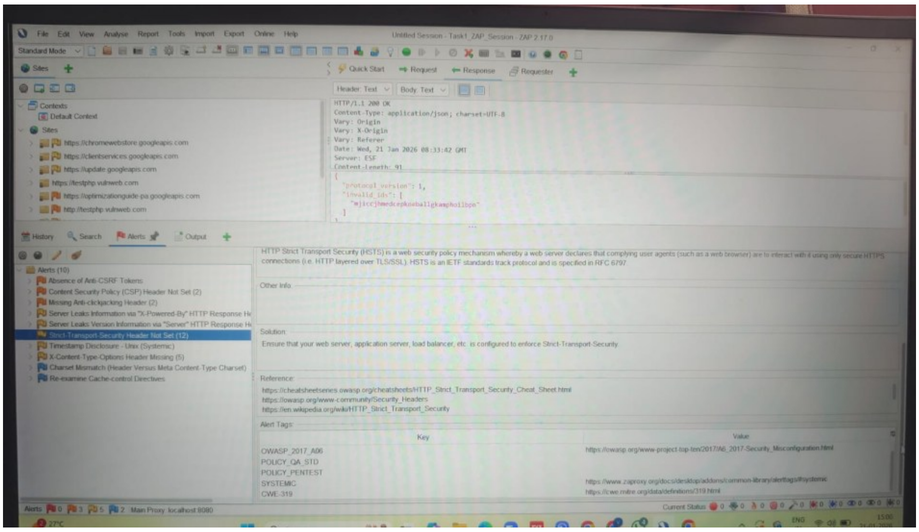
HTTP Strict Transport Security (HSTS) is a web security policy mechanism whereby a web server declares that complying user agents (such as a web browser) are to interact with it using only secure HTTPS connections (i.e. HTTP layered over TLS/SSL). HSTS is an IETF standards track protocol and is specified in RFC 6797

Other Info

Solution

Ensure that your web server, application server, load balancer, etc. is configured to enforce Strict-Transport-Security

Fig-10



- Contexts
  - Default Context
- Sites
  - https://chromewebstore.googleapis.com
  - https://clientservices.googleapis.com
  - https://update.googleapis.com
  - https://testphp.vulnweb.com
  - https://optimizationguide-pa.googleapis.com
  - http://testphp.vulnweb.com

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
Vary: Origin
Vary: X-Origin
Vary: Referer
Date: Wed, 21 Jan 2026 08:33:42 GMT
Server: ESF
Content-Length: 91

{
  "protocol_version": 1,
  "invalid_ids": [
    "mjiccjhmedcepknebaligkamphoilbpn"
  ]
}
```

- Alerts (10)
  - Absence of Anti-CSRF Tokens
  - Content Security Policy (CSP) Header Not Set (2)
  - Missing Anti-clickjacking Header (2)
  - Server Leaks Information via "X-Powered-By" HTTP Response He
  - Server Leaks Version Information via "Server" HTTP Response He
  - Strict-Transport-Security Header Not Set (12)
  - Timestamp Disclosure - Unix (Systemic)
  - X-Content-Type-Options Header Missing (5)
  - Charset Mismatch (Header Versus Meta Content-Type Charset)
  - Re-examine Cache-control Directives

HTTP Strict Transport Security (HSTS) is a web security policy mechanism whereby a web server declares that complying user agents (such as a web browser) are to interact with it using only secure HTTPS connections (i.e. HTTP layered over TLS/SSL). HSTS is an IETF standards track protocol and is specified in RFC 6797

Other Info

Solution

Ensure that your web server, application server, load balancer, etc. is configured to enforce Strict-Transport-Security

Reference

https://cheatsheetseries.owasp.org/cheatsheets/HTTP_Strict_Transport_Security_Cheat_Sheet.html
https://owasp.org/www-community/Security_Headers
https://en.wikipedia.org/wiki/HTTP_Strict_Transport_Security

Alert Tags

Key	Value
OWASP_2017_A06	https://owasp.org/www-project-top-ten/2017/A6_2017-Security_Misconfiguration.html
POLICY_QA_STD	
POLICY_PENTEST	
SYSTEMIC	https://www.zaproxy.org/docs/desktop/addons/common-library/alertflags/#systemic
CWE-319	https://cwe.mitre.org/data/definitions/319.html

Fig-11

Conclusion

This vulnerability assessment was conducted using OWASP ZAP on an intentionally vulnerable web application through a passive scanning approach. The assessment successfully identified multiple security misconfigurations, including missing security headers and the absence of Anti-CSRF protection.

Although the identified vulnerabilities were mostly of medium and low risk, they highlight common security weaknesses that can be exploited if left unaddressed. Proper implementation of recommended security controls such as security headers and Anti-CSRF tokens can significantly reduce the attack surface of a web application.

This task provided practical exposure to real-world vulnerability assessment techniques and emphasized the importance of proactive security testing during the web application development lifecycle.

Learning Outcomes

Through this task, the following key learnings were achieved:

Gained hands-on experience with OWASP ZAP for web application security testing

Understood the concept of passive vulnerability scanning

Learned how to identify common web vulnerabilities and security misconfigurations

Developed skills in risk classification (Low, Medium, High)

Improved ability to explain technical security issues in simple, business-friendly language

Learned to document vulnerabilities with clear impact and remediation steps

Enhanced understanding of web security best practices and defensive measures