Security Assessment Report -2

Efficiency On the GO

<u>Submitted To:</u>

Sumith

<u>Presented By:</u>

Veeresh Akki

TABLE OF CONTENTS

EXECUTIVE SUMMARY

Performed a security assessment of a testphp.vulnweb.com website on 03/04/2025. I have identified a total of 13 vulnerabilities within the scope of the engagement which are broken down by severity in the table below.

HIGH	MEDIUM	LOW	Informational
4	2	2	2

The highest severity vulnerabilities give potential attackers the opportunity to exploit the database and modify the web application. In order to ensure data confidentiality, integrity, and availability, security remediations should be implemented as described in the security assessment findings.

SCOPE

All testing was based on the scope as defined in the Request For Proposal (RFP) and official written communications. The items in scope are listed below.

Networks

Network	Note
44.228.249.3	Testphp.vulnweb.com

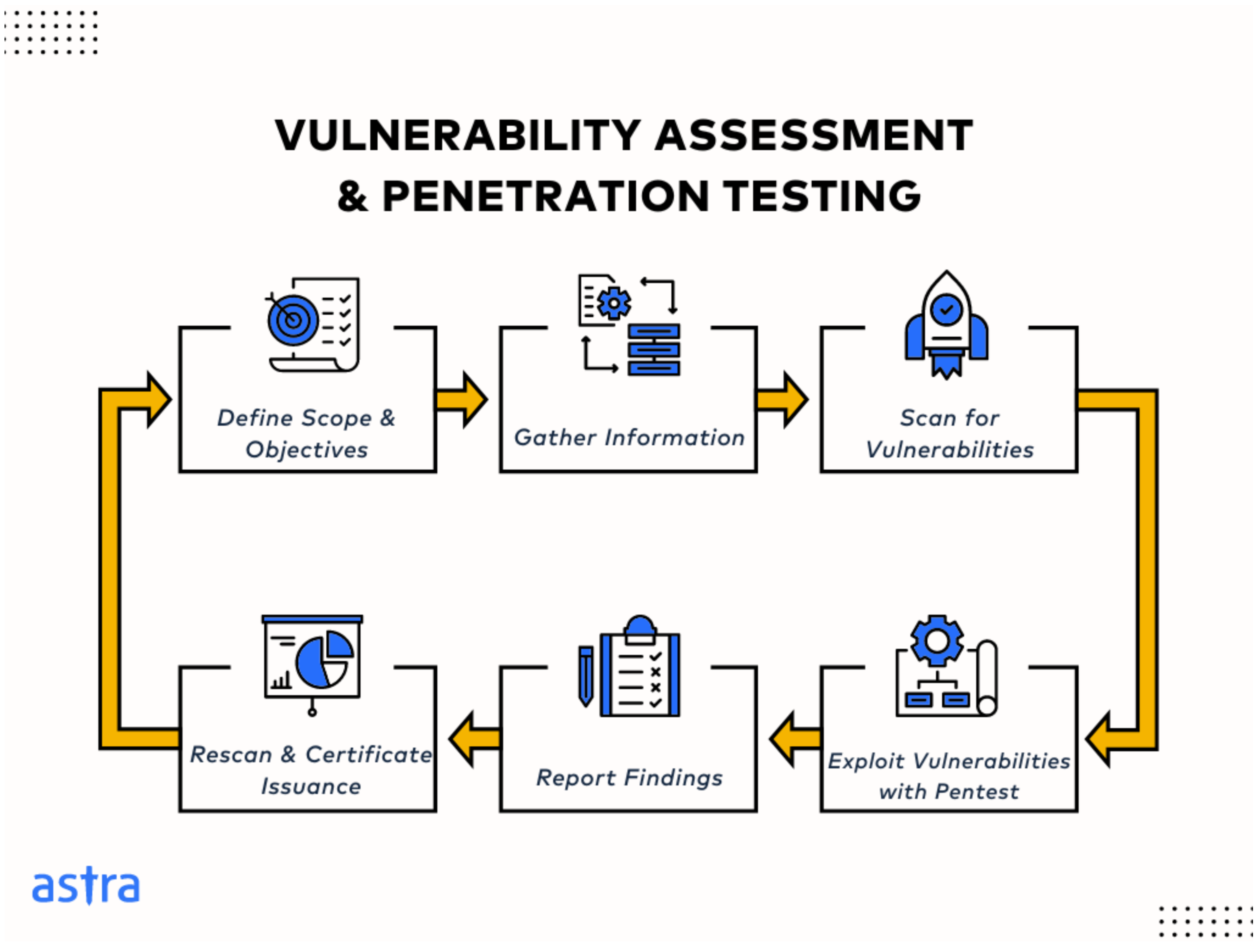
TESTING METHODOLOGY

The Vulnerability Assessment and Penetration Testing (VAPT) process begins with information gathering, where we identify technologies, exposed directories, and potential vulnerabilities using tools like Nmap, WhatWeb, and Gobuster. Next, we perform scanning and enumeration using OWASP ZAP, Burp Suite, and Nikto to detect security flaws.

In the vulnerability assessment phase, we manually and automatically test for SQL Injection (SQLi), Cross-Site Scripting (XSS), Cross-Site Request Forgery (CSRF), Broken Authentication, and Security Misconfigurations. Identified vulnerabilities are then exploited with PoC (Proof of Concept) attacks, such as SQL injection queries, XSS payloads, and CSRF attacks, to demonstrate their impact.

Following exploitation, we analyze post-exploitation risks, including session hijacking and privilege escalation. Finally, a detailed VAPT report is prepared, documenting findings with evidence, screenshots, and mitigation recommendations to enhance security. This structured methodology ensures a comprehensive assessment of vulnerabilities and their potential impact.

The following image is a graphical representation of this methodo


VULNERABILITY ASSESSMENT & PENETRATION TESTING
Define Scope & Objectives
Gather Information
Scan for Vulnerabilities
Exploit Vulnerabilities with Pentest
Report Findings
Rescan & Certificate Issuance
astra

CLASSIFICATION DEFINITIONS

Risk Classifications

Level	Score	Description
High	**10-8**	The vulnerability poses an urgent threat to the organization, and remediation should be prioritized.
Medium	**4-7**	Successful exploitation is possible and may result in notable disruption of business functionality. This vulnerability should be remediated when feasible.
Low	**1-3**	The vulnerability poses a negligible/minimal threat to the organization. The presence of this vulnerability should be noted and remediated if possible.
Informational	**0**	These findings have no clear threat to the organization, but may cause business processes to function differently than desired or reveal sensitive information about the company.

Exploitation Likelihood Classifications

Likelihood	Description
Likely	Exploitation methods are well-known and can be performed using publicly available tools. Low-skilled attackers and automated tools could successfully exploit the vulnerability with minimal difficulty.
Possible	Exploitation methods are well-known, may be performed using public tools, but require configuration. Understanding of the underlying system is required for successful exploitation.
Unlikely	Exploitation requires deep understanding of the underlying systems or advanced technical skills. Precise conditions may be required for successful exploitation.

Business Impact Classifications

Impact	Description
Major	Successful exploitation may result in large disruptions of critical business functions across the organization and significant financial damage.
Moderate	Successful exploitation may cause significant disruptions to non-critical business functions.
Minor	Successful exploitation may affect few users, without causing much disruption to routine business functions.

Remediation Difficulty Classifications

Difficulty	Description
Hard	Remediation may require extensive reconfiguration of underlying systems that is time consuming. Remediation may require disruption of normal business functions.
Moderate	Remediation may require minor reconfigurations or additions that may be time-intensive or expensive.
Easy	Remediation can be accomplished in a short amount of time, with little difficulty.

ASSESSMENT FINDINGS

Number	Finding	Risk Score	Risk
1	SQL Injection	9	**High**
2	Broken Access Control	9	**High**
3	CSRF Tokens	8	**High**
4	Cross Site Scripting	8	**High**
5	Content Security Policy (CSP) Header Not Set	5	**Medium**
6	Missing Anti-clickjacking Header	4	**Low**
7	Server Leaks Version Information via "Server" HTTP Response Header Field	2	**Low**
8	X-Content-Type-Options Header Missing	2	**Low**
9	Authentication Request Identified	0	**Informational**
10	Charset Mismatch (Header Versus Meta Content-Type Charset)	0	**Informational**

TEMPLATE NOTE: (Sorting by descending risk score)

1 - SQL Injection Vulnerability Finding

HIGH RISK (9/10)	
Exploitation Likelihood	**Possible**
Business Impact	**Severe**
Remediation Difficulty	**Easy**

Security Description:

SQL Injection (SQLi) is a web security vulnerability that occurs when an attacker is able to inject malicious SQL queries into an application's database query via input fields. This happens when user inputs are not properly sanitized or validated, allowing direct execution of SQL commands. Attackers can manipulate the database, extract sensitive information, delete records, modify user privileges, or even execute administrative database commands.

Impact:

- Data Breach: Unauthorized access to stored credentials, personal information, and financial records.
- Authentication Bypass: Attackers can bypass login pages and gain unauthorized admin access.
- Data Manipulation: Attackers can insert, update, or delete important records.
- Server Compromise: In some cases, SQLi can lead to remote code execution, compromising the entire server.

Severity: Critical

Steps to Test:

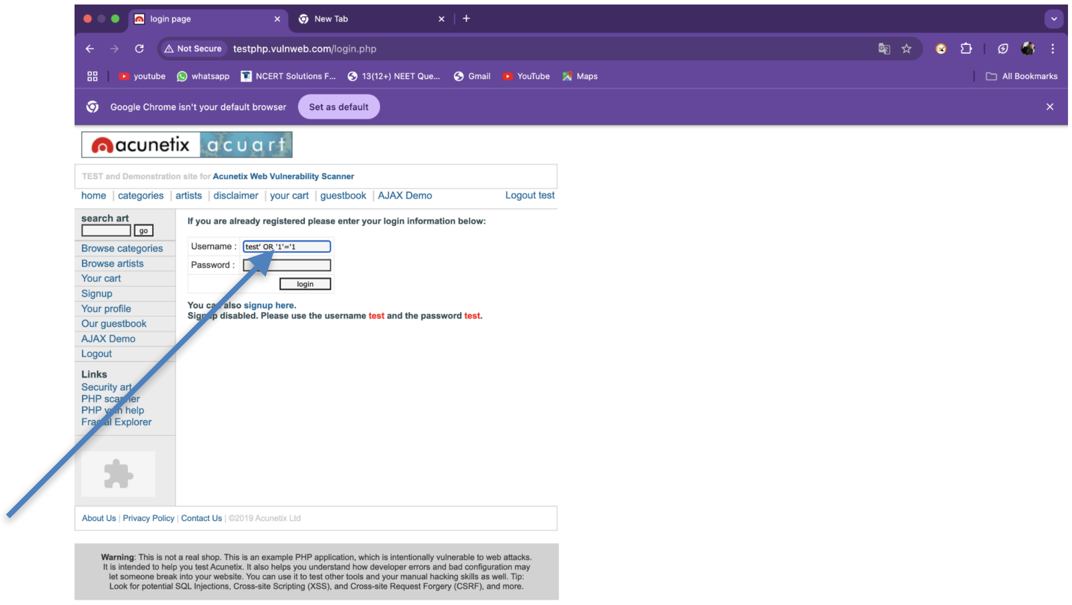
1. Open the target website that is http://testphp.vulnweb.com/
2. Open the login page and inject the sql query.
3. Try to login without password.
4. As this website is vulnerable to sql injection, it logged in successfully without password.

PoC - Payloads: test’OR ‘1’=‘1

Remediation

- Use **prepared statements** and **parameterized queries** to avoid direct SQL execution.
- Implement **strict input validation** and **allowlist filtering**.
- Restrict **database permissions** for the web application user.
- Enable **Web Application Firewall (WAF)** to detect and block SQLi attempts.

Snapshots:



TEST and Demonstration site for Acunetix Web Vulnerability Scanner

home | categories | artists | disclaimer | your cart | guestbook | AJAX Demo Logout test

search art go

Browse categories
Browse artists
Your cart
Signup
Your profile
Our guestbook
AJAX Demo
Logout

Links
Security art
PHP scanner
PHP vuln help
Fractal Explorer

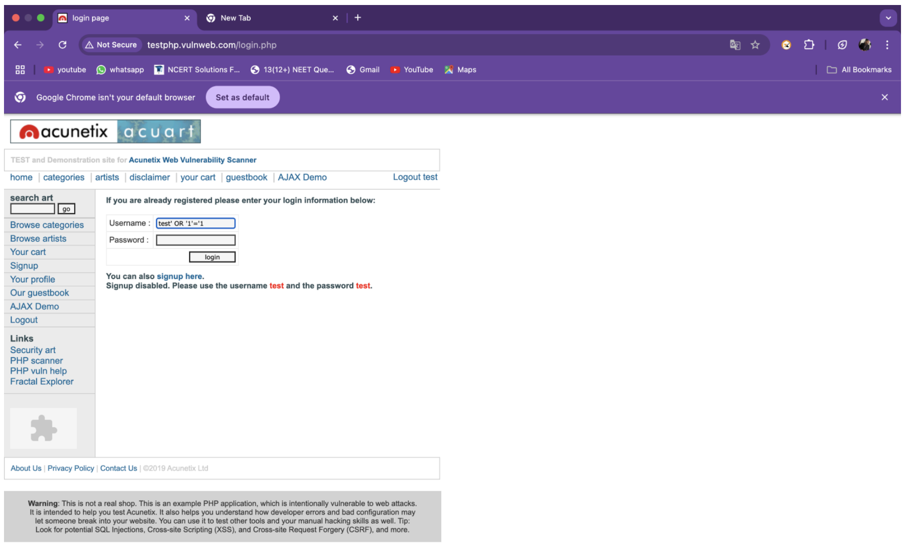
If you are already registered please enter your login information below:

Username :	test' OR '1'='1
Password :	
	login

You can also signup here.
Signup disabled. Please use the username test and the password test.

About Us | Privacy Policy | Contact Us | ©2019 Acunetix Ltd

Warning: This is not a real shop. This is an example PHP application, which is intentionally vulnerable to web attacks. It is intended to help you test Acunetix. It also helps you understand how developer errors and bad configuration may let someone break into your website. You can use it to test other tools and your manual hacking skills as well. Tip: Look for potential SQL Injections, Cross-site Scripting (XSS), and Cross-site Request Forgery (CSRF), and more.



TEST and Demonstration site for Acunetix Web Vulnerability Scanner

home | categories | artists | disclaimer | your cart | guestbook | AJAX Demo Logout test

search art go

Browse categories
Browse artists
Your cart
Signup
Your profile
Our guestbook
AJAX Demo
Logout

Links
Security art
PHP scanner
PHP vuln help
Fractal Explorer

If you are already registered please enter your login information below:

Username :	test' OR '1'='1
Password :	
	login

You can also signup here.
Signup disabled. Please use the username test and the password test.

About Us | Privacy Policy | Contact Us | ©2019 Acunetix Ltd

Warning: This is not a real shop. This is an example PHP application, which is intentionally vulnerable to web attacks. It is intended to help you test Acunetix. It also helps you understand how developer errors and bad configuration may let someone break into your website. You can use it to test other tools and your manual hacking skills as well. Tip: Look for potential SQL Injections, Cross-site Scripting (XSS), and Cross-site Request Forgery (CSRF), and more.

APPENDIX A - TOOLS USED

TOOL	DESCRIPTION
BurpSuite Community Edition	Used for testing of web applications.
SQLMAP	Used for exploitation of vulnerable services and vulnerability scanning.
Zap Fuzzer	To exploit sql payload

Table A.1: *Tools used during assessment*

2 - Broken Access Control Vulnerability Finding

HIGH RISK (9/10)	
Exploitation Likelihood	**Possible**
Business Impact	**Severe**
Remediation Difficulty	**Easy**

Security Description:

Broken Access Control is a critical security vulnerability that occurs when users can access resources or functionalities that should be restricted. This happens due to improper authorization checks, missing access controls, or predictable URLs that expose sensitive information. Attackers can exploit this to view unauthorized data, modify user accounts, or escalate privileges.

Impact

- **Unauthorized Data Access**: Attackers can access sensitive information of other users.
- **Privilege Escalation**: Users can gain admin-level privileges by modifying requests.
- **Account Takeover**: Exploiting weak access control can lead to session hijacking and unauthorized modifications.

Severity: Critical

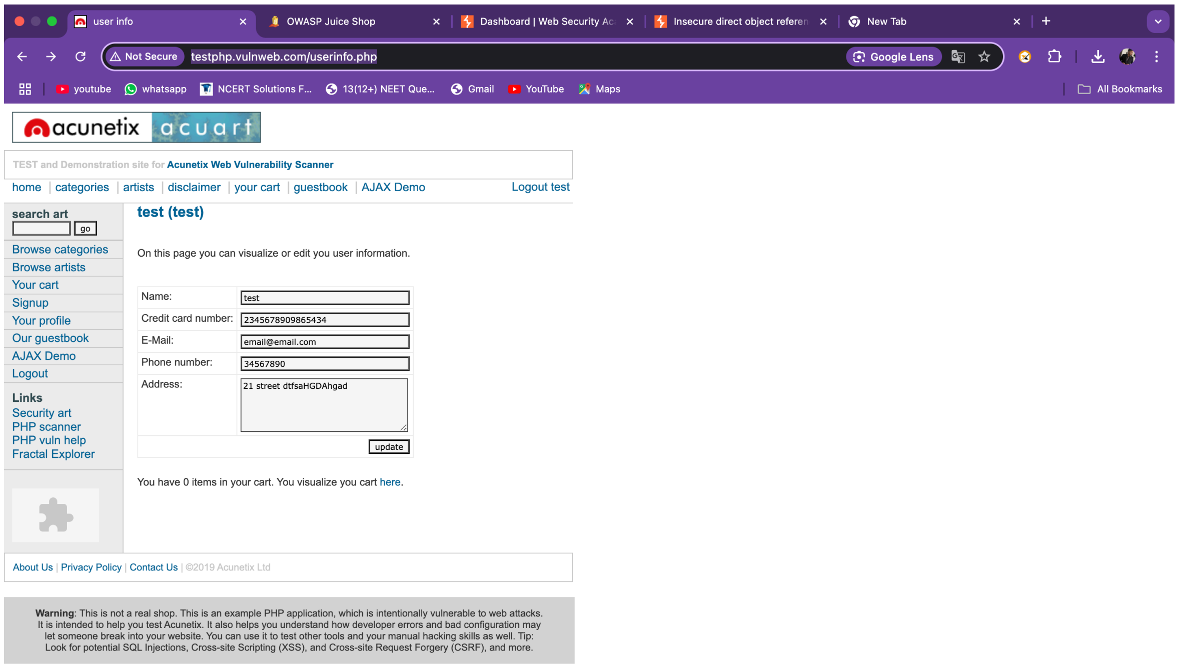
Steps to Test:

1. **Login as a normal user** on `http://testphp.vulnweb.com/`.
2. Navigate to a **profile or order details** page:
3. Then in the url change the path to **(http://testphp.vulnweb.com/admin/).**
4. If we access the admin page then it is vulnerable to broken access control vulnerability.

Severity: Critical

PoC - Payloads: http://testphp.vulnweb.com/admin/

Snapshots:


user info | OWASP Juice Shop | Dashboard | Web Security Ac | Insecure direct object referen | New Tab

Not Secure testphp.vulnweb.com/userinfo.php

acunetix acuart

TEST and Demonstration site for Acunetix Web Vulnerability Scanner

home | categories | artists | disclaimer | your cart | guestbook | AJAX Demo   Logout test

search art [ ] go

Browse categories
Browse artists
Your cart
Signup
Your profile
Our guestbook
AJAX Demo
Logout

Links
Security art
PHP scanner
PHP vuln help
Fractal Explorer

test (test)

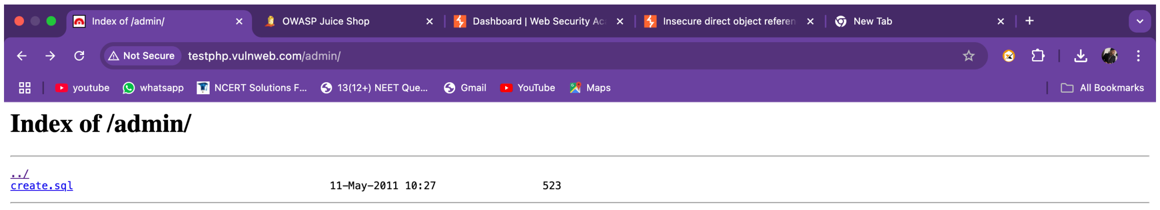
On this page you can visualize or edit you user information.

Name:	test
Credit card number:	2345678909865434
E-Mail:	email@email.com
Phone number:	34567890
Address:	21 street dtfsaHGDAhgad
	update

You have 0 items in your cart. You visualize you cart here.

About Us | Privacy Policy | Contact Us | ©2019 Acunetix Ltd

Warning: This is not a real shop. This is an example PHP application, which is intentionally vulnerable to web attacks. It is intended to help you test Acunetix. It also helps you understand how developer errors and bad configuration may let someone break into your website. You can use it to test other tools and your manual hacking skills as well. Tip: Look for potential SQL Injections, Cross-site Scripting (XSS), and Cross-site Request Forgery (CSRF), and more.


Index of /admin/ | OWASP Juice Shop | Dashboard | Web Security Ac | Insecure direct object referen | New Tab

Not Secure testphp.vulnweb.com/admin/

Index of /admin/

```
../
create.sql                                         11-May-2011 10:27                 523
```

APPENDIX D- TOOLS USED

TOOL	***DESCRIPTION***
Zap Fuzzer	*To exploit Directory Fuzzer*

Table D.4: *Tools used during assessment*

3 - CSRF Tokens Vulnerability Finding

HIGH RISK (9/10)	
Exploitation Likelihood	**Possible**
Business Impact	**Severe**
Remediation Difficulty	**Easy**

Security Description:

CSRF is an attack where a user **unknowingly executes an unwanted action** in a web application where they are authenticated. This is done by **tricking the user into clicking a malicious link or visiting a crafted website**, which sends requests on their behalf.

Impact

- **Forced password changes** without user consent.
- **Unauthorized transactions or fund transfers**.
- **Forced account deletions or privilege escalations**.

Severity: High

Steps to Test

i) open the target website and login.

ii) Then copy the form code from the victims page.

iii) Host one new website where we insert the malicious code of form, which we have copied from victims page.

iv) Then send the link of the hosted website to victim and made him to click on it.

v) Once the victim clicks on it, automatically the username get changed.

vi) Without victim consent the username got changed.

Severity: Critical

PoC - Payloads:

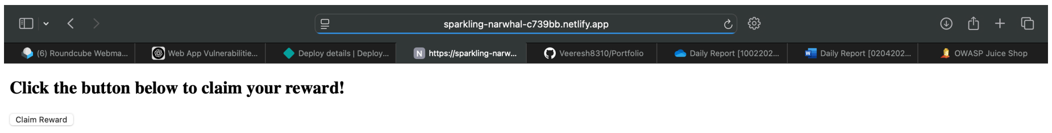
```
 <form action="http://testphp.vulnweb.com/sendcommand.php" method="POST">

    <input type="hidden" name="cart_id" value="741b12df2badb8c8f179775a14715964">

    <input type="submit" value="Click here to claim your free gift!">

</form>
```

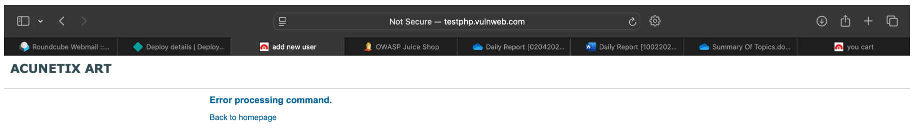
Snapshots:



sparkling-narwhal-c739bb.netlify.app
(6) Roundcube Webma...
Web App Vulnerabilitie...
Deploy details | Deploy...
https://sparkling-narw...
Veeresh8310/Portfolio
Daily Report [1002202...
Daily Report [0204202...
OWASP Juice Shop

Click the button below to claim your reward!

Claim Reward



Error processing command.

Back to homepage

APPENDIX C - TOOLS USED

TOOL	***DESCRIPTION***
BurpSuite Community Edition	*Used for testing of web applications.*
Zap Fuzzer	*To exploit XSS payload*

Table C.3: *Tools used during assessment*

4 - XSS Vulnerability Finding

HIGH RISK (9/10)	
Exploitation Likelihood	**Possible**
Business Impact	**Severe**
Remediation Difficulty	**Easy**

Security Description:

Cross-Site Scripting (XSS) is a client-side attack where an attacker injects **malicious JavaScript code** into a web application, which is then executed in the victim's browser. This can result in **session hijacking, defacement, credential theft, and phishing attacks**. XSS vulnerabilities occur when user inputs are **not sanitized properly** before being displayed on web pages.

Impact

- **Session Hijacking**: Attackers can steal session tokens and gain unauthorized access.
- **Phishing Attacks**: Redirect users to malicious websites to collect credentials.
- **Defacement**: Modify the UI to display misleading information.

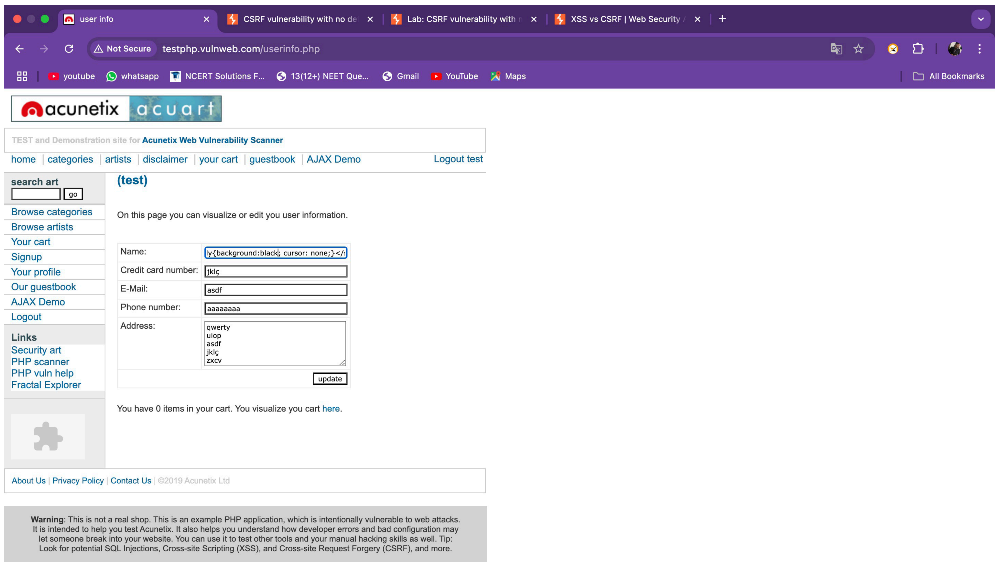
Severity: High

Steps to Test:

1. Open the target site.
2. Enter the Malicious code into the targeted field, here the Name text field is target site.
3. Then hit update button.
4. After that automatically the background colour changes to black, which is caused due to the Malicious code which i have injected.

Severity: Critical

Snapshots:



testphp.vulnweb.com/userinfo.php

TEST and Demonstration site for **Acunetix Web Vulnerability Scanner**

home | categories | artists | disclaimer | your cart | guestbook | AJAX Demo | Logout test

(test)

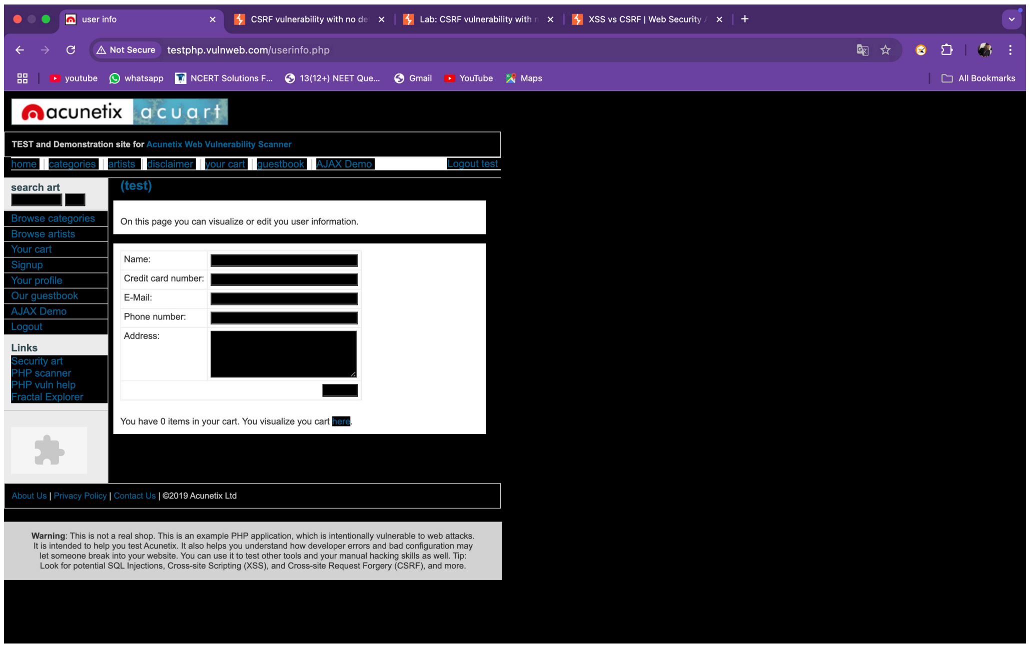
On this page you can visualize or edit you user information.

Name:	y{background:black; cursor: none;}</:
Credit card number:	jklç
E-Mail:	asdf
Phone number:	aaaaaaaa
Address:	qwerty uiop asdf jklç zxcv

You have 0 items in your cart. You visualize you cart here.

About Us | Privacy Policy | Contact Us | ©2019 Acunetix Ltd

Warning: This is not a real shop. This is an example PHP application, which is intentionally vulnerable to web attacks. It is intended to help you test Acunetix. It also helps you understand how developer errors and bad configuration may let someone break into your website. You can use it to test other tools and your manual hacking skills as well. Tip: Look for potential SQL Injections, Cross-site Scripting (XSS), and Cross-site Request Forgery (CSRF), and more.



testphp.vulnweb.com/userinfo.php

TEST and Demonstration site for Acunetix Web Vulnerability Scanner

home | categories | artists | disclaimer | your cart | guestbook | AJAX Demo | Logout test

(test)

On this page you can visualize or edit you user information.

Name:
Credit card number:
E-Mail:
Phone number:
Address:

You have 0 items in your cart. You visualize you cart here.

About Us | Privacy Policy | Contact Us | ©2019 Acunetix Ltd

Warning: This is not a real shop. This is an example PHP application, which is intentionally vulnerable to web attacks. It is intended to help you test Acunetix. It also helps you understand how developer errors and bad configuration may let someone break into your website. You can use it to test other tools and your manual hacking skills as well. Tip: Look for potential SQL Injections, Cross-site Scripting (XSS), and Cross-site Request Forgery (CSRF), and more.

APPENDIX B - TOOLS USED

TOOL	DESCRIPTION
BurpSuite Community Edition	Used for testing of web applications.
Visual Studio	To start live server for the code.

Table B.2: *Tools used during assessment*

5 – Content Security Policy (CSP) Header Not Set Vulnerability Finding

Medium RISK (6/10)	
Exploitation Likelihood	**Medium**
Business Impact	**Moderate**
Remediation Difficulty	**Easy**

Description:

The Content Security Policy (CSP) header is an important security measure that helps prevent attacks such as Cross-Site Scripting (XSS), data injection, and malicious content loading (e.g., from unauthorized or malicious sources). By not setting a CSP header, the application is exposed to these types of attacks. Without CSP, attackers can inject malicious scripts or resources into a web page, leading to data theft, session hijacking, malware distribution, or even site defacement.

Impact:

1. Increased Risk of Cross-Site Scripting (XSS)
2. Clickjacking Attacks
3. Data Injection Attacks
4. Defacement and Browser Exploits

Analysis

CSP Header Missing

The application doesn't include a Content Security Policy (CSP) header, which leaves it vulnerable to a variety of attacks. CSP is a security feature that allows web developers to control the resources that can be loaded by a webpage, such as JavaScript, CSS, images, and other media. Without CSP, attackers can inject malicious scripts or load content from untrusted sources.

URLs Affected:

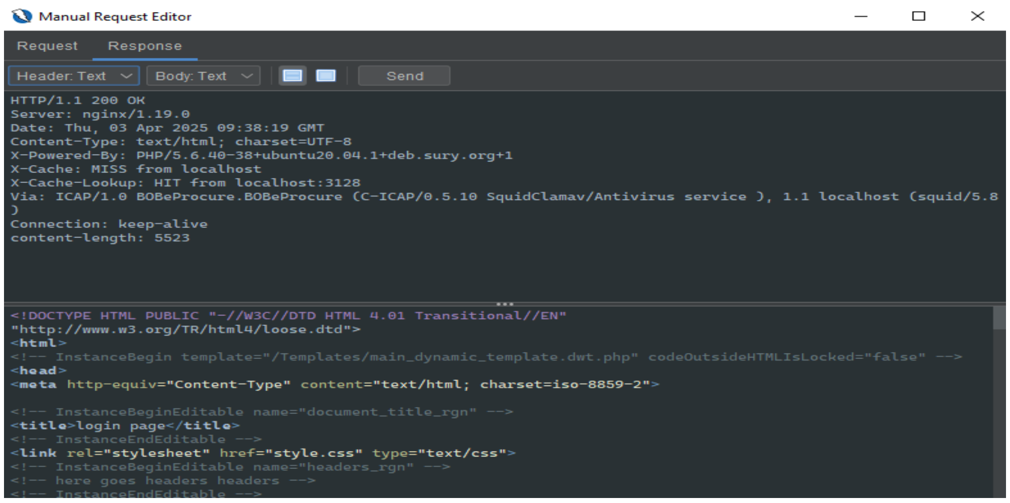
1. http://testphp.vulnweb.com/AJAX/index.php
2. http://testphp.vulnweb.com/artists.php
3. http://testphp.vulnweb.com/artists.php?artist=1
4. http://testphp.vulnweb.com/artists.php?artist=2

5. http://testphp.vulnweb.com/artists.php?artist=3
6. http://testphp.vulnweb.com/cart.php
7. http://testphp.vulnweb.com/categories.php
8. http://testphp.vulnweb.com/disclaimer.php
9. http://testphp.vulnweb.com/guestbook.php
10. http://testphp.vulnweb.com/high
11. http://testphp.vulnweb.com/hpp/
12. http://testphp.vulnweb.com/hpp/?pp=12
13. http://testphp.vulnweb.com/hpp/params.php?p=valid&pp=12
14. http://testphp.vulnweb.com/index.php
15. http://testphp.vulnweb.com/listproducts.php?artist=1
16. http://testphp.vulnweb.com/listproducts.php?artist=2
17. http://testphp.vulnweb.com/listproducts.php?artist=3
18. Still 18 more are affected

All of the above URLs do not include a CSP header, leaving them vulnerable to content injection and data theft attacks.

Severity: Medium to High

Snapshots:



```
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Thu, 03 Apr 2025 09:38:19 GMT
Content-Type: text/html; charset=UTF-8
X-Powered-By: PHP/5.6.40-38+ubuntu20.04.1+deb.sury.org+1
X-Cache: MISS from localhost
X-Cache-Lookup: HIT from localhost:3128
Via: ICAP/1.0 BOBeProcure.BOBeProcure (C-ICAP/0.5.10 SquidClamav/Antivirus service ), 1.1 localhost (squid/5.8
)
Connection: keep-alive
content-length: 5523
```

```
<!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">

<!-- InstanceBeginEditable name="document_title_rgn" -->
<title>login page</title>
<!-- InstanceEndEditable -->
<link rel="stylesheet" href="style.css" type="text/css">
<!-- InstanceBeginEditable name="headers_rgn" -->
<!-- here goes headers headers -->
<!-- InstanceEndEditable -->
```

Remediation Recommendations

1. **Implement Content Security Policy (CSP) Header:**

   ```
   Content-Security-Policy: default-src 'self'; script-src 'self'; style-src 'self'; img-src 'self';
   ```
2. **Review and Customize CSP Rules:**
3. **Test CSP Policies:**
4. **Use HTTPS for All Resources:**
5. **Audit and Update Regularly:**
6.

APPENDIX E- TOOLS USED

TOOL	***DESCRIPTION***
Zap	*To see the request and response*

Table E.5: *Tools used during assessment*

6 – Missing Anti-clickjacking Header Vulnerability Finding

Medium RISK (6/10)	
Exploitation Likelihood	**Medium**
Business Impact	**Moderate**
Remediation Difficulty	**Easy**

Security Implications

Clickjacking is a type of attack where a malicious site can trick a user into interacting with a page that is hidden in a frame. This can lead to unintended actions being performed by the user, such as changing account settings or making a purchase, without their knowledge. The X-Frame-Options header or Content-Security-Policy (CSP) with the frame-ancestors directive can prevent this vulnerability by ensuring that the site cannot be embedded in a frame on another domain.

Analysis

Missing Anti-clickjacking Header

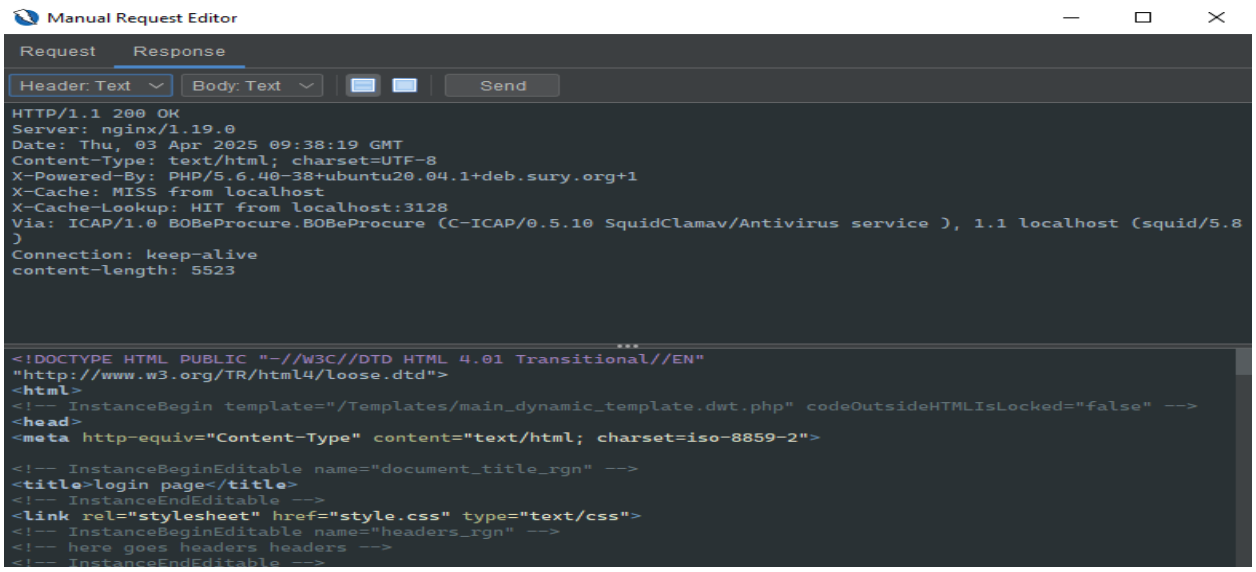
The application does not include the **X-Frame-Options** header or a **Content-Security-Policy (CSP)** with the frame-ancestors directive to protect against **Clickjacking** attacks. Without this protection, malicious websites could embed the application's pages in a frame, potentially tricking users into performing actions on the application without their knowledge.

URLs Affected:

1. http://testphp.vulnweb.com/AJAX/index.php
2. http://testphp.vulnweb.com/artists.php
3. http://testphp.vulnweb.com/artists.php?artist=1
4. http://testphp.vulnweb.com/artists.php?artist=2
5. http://testphp.vulnweb.com/artists.php?artist=3
6. http://testphp.vulnweb.com/cart.php
7. http://testphp.vulnweb.com/categories.php
8. http://testphp.vulnweb.com/disclaimer.php

9. http://testphp.vulnweb.com/guestbook.php
10. http://testphp.vulnweb.com/hpp/
11. http://testphp.vulnweb.com/hpp/?pp=12
12. http://testphp.vulnweb.com/hpp/params.php?p=valid&pp=12
13. http://testphp.vulnweb.com/index.php
14. http://testphp.vulnweb.com/listproducts.php?artist=1
15. http://testphp.vulnweb.com/listproducts.php?artist=2
16. http://testphp.vulnweb.com/listproducts.php?artist=3
17. http://testphp.vulnweb.com/listproducts.php?cat=1

All of these URLs are vulnerable because they do not include the **X-Frame-Options** header or **CSP** header with frame-ancestors directive to prevent clickjacking attacks.



```
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Thu, 03 Apr 2025 09:38:19 GMT
Content-Type: text/html; charset=UTF-8
X-Powered-By: PHP/5.6.40-38+ubuntu20.04.1+deb.sury.org+1
X-Cache: MISS from localhost
X-Cache-Lookup: HIT from localhost:3128
Via: ICAP/1.0 BOBeProcure.BOBeProcure (C-ICAP/0.5.10 SquidClamav/Antivirus service ), 1.1 localhost (squid/5.8
)
Connection: keep-alive
content-length: 5523
```

```
<!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">

<!-- InstanceBeginEditable name="document_title_rgn" -->
<title>login page</title>
<!-- InstanceEndEditable -->
<link rel="stylesheet" href="style.css" type="text/css">
<!-- InstanceBeginEditable name="headers_rgn" -->
<!-- here goes headers headers -->
<!-- InstanceEndEditable -->
```

Remediation Recommendations

1. **Implement the X-Frame-Options Header:**

   X-Frame-Options: SAMEORIGIN

2. **Implement Content-Security-Policy (CSP) with frame-ancestors:**

   Content-Security-Policy: frame-ancestors 'self';

3. **Test the Protection Mechanisms:**

APPENDIX F- TOOLS USED

TOOL	***DESCRIPTION***
Zap	*To see the request and response*

Table F.6: *Tools used during assessment*

7 – Server Leaks Version Information via "Server" HTTP Response Header Vulnerability Finding

Low RISK (3/10)	
Exploitation Likelihood	**Medium**
Business Impact	**Moderate**
Remediation Difficulty	**Easy**

Security Implications

*The **Server** HTTP header is disclosing information about the underlying server software (in this case, **nginx/1.19.0**). While this is not a critical vulnerability, it provides attackers with knowledge about the specific server software and version, which can potentially assist in identifying vulnerabilities associated with that version. The disclosure of such information should be avoided as it can be leveraged for targeted attacks or to determine specific server configurations that may be insecure.*

Analysis

The **Server** header is found in the HTTP responses from several pages on the website, revealing the version of the web server:

arduino

Copy

Server: nginx/1.19.0

This header is present on the following URLs:

- http://testphp.vulnweb.com/AJAX/index.php
- http://testphp.vulnweb.com/artists.php
- http://testphp.vulnweb.com/cart.php
- http://testphp.vulnweb.com/categories.php
- http://testphp.vulnweb.com/guestbook.php
- (and many others).

URLs Affected:

- http://testphp.vulnweb.com/AJAX/index.php
- http://testphp.vulnweb.com/artists.php
- http://testphp.vulnweb.com/cart.php
- http://testphp.vulnweb.com/categories.php
- http://testphp.vulnweb.com/disclaimer.php
- http://testphp.vulnweb.com/guestbook.php
- http://testphp.vulnweb.com/hpp/
- http://testphp.vulnweb.com/index.php
- http://testphp.vulnweb.com/listproducts.php?artist=1
- ... (additional affected URLs as per the evidence).

Impact on Security:

While this vulnerability is **Low** in severity, it could still be exploited by attackers to identify specific vulnerabilities in the **nginx/1.19.0** server version. Potential risks include:

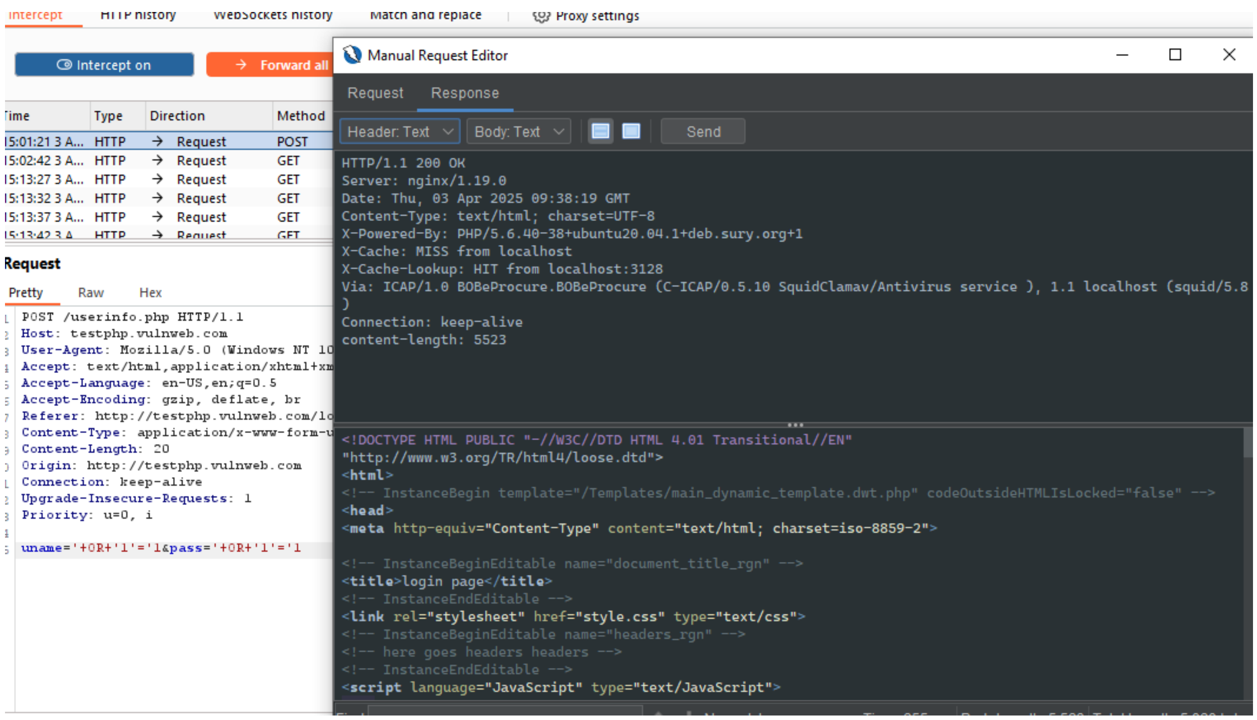
- **Targeted Attacks**: Attackers can search for exploits targeting that specific version of **nginx**.
- **Reconnaissance**: Attackers may use the exposed server version information to refine their attacks.

Remediation Recommendations:

To mitigate this issue, the **Server** header should be removed from all HTTP responses. You can do this by modifying server configurations as follows:

1. **In nginx**:
   Add the following directive to your **nginx.conf** to disable the **Server** header:
   ```
   server_tokens off;
   ```
2. **In Apache**:
   To remove the server version from **Apache**, you can add the following directive to the **httpd.conf** file:
   ```
   ServerTokens Prod

   ServerSignature Off
   ```



Time	Type	Direction	Method
15:01:21 3 A...	HTTP	→ Request	POST
15:02:42 3 A...	HTTP	→ Request	GET
15:13:27 3 A...	HTTP	→ Request	GET
15:13:32 3 A...	HTTP	→ Request	GET
15:13:37 3 A...	HTTP	→ Request	GET
15:13:42 3 A	HTTP	→ Request	GET

```
POST /userinfo.php HTTP/1.1
Host: testphp.vulnweb.com
User-Agent: Mozilla/5.0 (Windows NT 10
Accept: text/html,application/xhtml+xm
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: http://testphp.vulnweb.com/lo
Content-Type: application/x-www-form-u
Content-Length: 20
Origin: http://testphp.vulnweb.com
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Priority: u=0, i

uname='+OR+'1'='1&pass='+OR+'1'='1
```

```
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Thu, 03 Apr 2025 09:38:19 GMT
Content-Type: text/html; charset=UTF-8
X-Powered-By: PHP/5.6.40-38+ubuntu20.04.1+deb.sury.org+1
X-Cache: MISS from localhost
X-Cache-Lookup: HIT from localhost:3128
Via: ICAP/1.0 BOBeProcure.BOBeProcure (C-ICAP/0.5.10 SquidClamav/Antivirus service ), 1.1 localhost (squid/5.8
)
Connection: keep-alive
content-length: 5523
```

```
<!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">

<!-- InstanceBeginEditable name="document_title_rgn" -->
<title>login page</title>
<!-- InstanceEndEditable -->
<link rel="stylesheet" href="style.css" type="text/css">
<!-- InstanceBeginEditable name="headers_rgn" -->
<!-- here goes headers headers -->
<!-- InstanceEndEditable -->
<script language="JavaScript" type="text/JavaScript">
```

Remediation Recommendations

- **Implement Content Security Policy (CSP) Header:**
- Content-Security-Policy: default-src 'self'; script-src 'self'; style-src 'self'; img-src 'self';
- **Review and Customize CSP Rules:**
- **Test CSP Policies:**
- **Use HTTPS for All Resources:**
- **Audit and Update Regularly:**

APPENDIX G- TOOLS USED

TOOL	***DESCRIPTION***
Zap	*To see the request and response*

Table G.7: *Tools used during assessment*

8 – X-Content-Type-Options Header Missing Vulnerability Finding

Low RISK (2/10)	
Exploitation Likelihood	**Medium**
Business Impact	**Easy**
Remediation Difficulty	**Moderate**

Security Implications

*The **X-Content-Type-Options** header is not set to **'nosniff'**. This header is designed to prevent browsers (especially older versions of Internet Explorer and Chrome) from performing MIME sniffing on the response body. Without this header, browsers may incorrectly interpret the content type and display content in a way not intended by the server, leading to potential security risks such as executing malicious content. Modern browsers and Firefox use the declared content type and do not perform MIME sniffing, but legacy browsers may still be vulnerable.*

Analysis

The **X-Content-Type-Options** header is missing in the HTTP responses for several URLs:

Example response header:

pgsql

Copy

X-Content-Type-Options: (not present)

The following URLs were found to be affected by this issue:

- http://testphp.vulnweb.com/AJAX/index.php
- http://testphp.vulnweb.com/artists.php
- http://testphp.vulnweb.com/cart.php
- http://testphp.vulnweb.com/categories.php
- http://testphp.vulnweb.com/disclaimer.php
- http://testphp.vulnweb.com/guestbook.php
- http://testphp.vulnweb.com/images/logo.gif
- http://testphp.vulnweb.com/index.php

- http://testphp.vulnweb.com/listproducts.php?artist=1
- (and many others).

URLs Affected:

- http://testphp.vulnweb.com/AJAX/index.php
- http://testphp.vulnweb.com/artists.php
- http://testphp.vulnweb.com/cart.php
- http://testphp.vulnweb.com/categories.php
- http://testphp.vulnweb.com/guestbook.php
- http://testphp.vulnweb.com/images/logo.gif
- http://testphp.vulnweb.com/index.php
- http://testphp.vulnweb.com/listproducts.php?artist=1
- (Additional affected URLs as per evidence).

Impact:

- MIME Sniffing: Without this header, browsers may incorrectly interpret or display the content type, potentially leading to security risks or rendering issues.
- Cross-Site Scripting (XSS): This vulnerability could be leveraged in certain attacks where content is interpreted differently than expected.

Remediation Recommendations:

Remediation Recommendations:

To mitigate this vulnerability, ensure the web application/web server sets the **X-Content-Type-Options** header to **'nosniff'** for all HTTP responses. Here's how to do this:

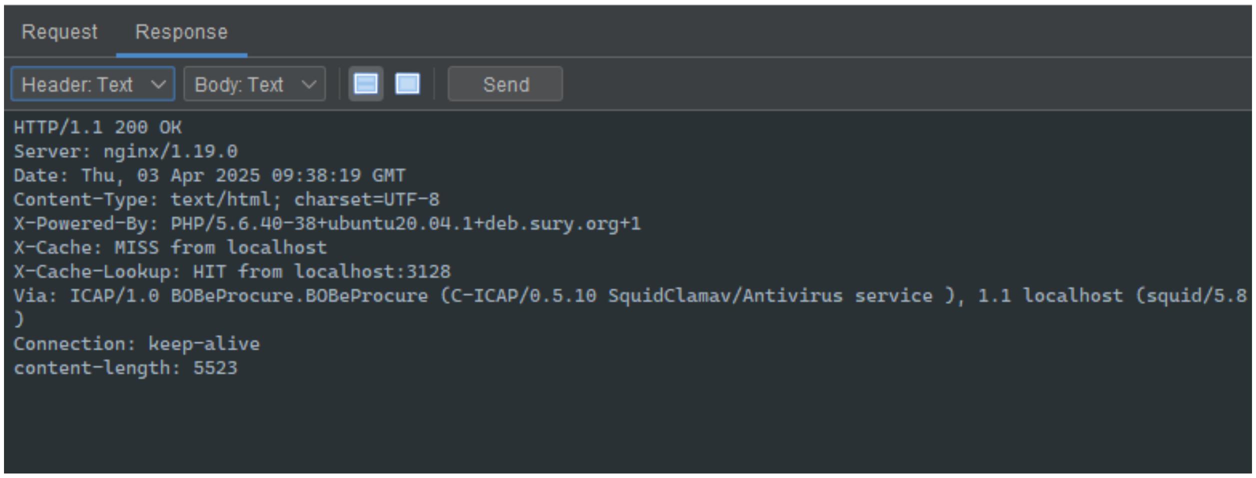
1. **In nginx**: Add the following directive to your **nginx.conf**:

   ```
   X-Content-Type-Options "nosniff";
   ```

2. **In Apache**: Add the following to your **.htaccess** or **httpd.conf** file:

   ```
   Header set X-Content-Type-Options "nosniff"
   ```

3. **In other server configurations**, ensure that the **X-Content-Type-Options** header is set to **'nosniff'** for all responses.



```
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Thu, 03 Apr 2025 09:38:19 GMT
Content-Type: text/html; charset=UTF-8
X-Powered-By: PHP/5.6.40-38+ubuntu20.04.1+deb.sury.org+1
X-Cache: MISS from localhost
X-Cache-Lookup: HIT from localhost:3128
Via: ICAP/1.0 BOBeProcure.BOBeProcure (C-ICAP/0.5.10 SquidClamav/Antivirus service ), 1.1 localhost (squid/5.8
)
Connection: keep-alive
content-length: 5523
```

APPENDIX E- TOOLS USED

TOOL	***DESCRIPTION***
Zap	*To see the request and response*

Table E.5: *Tools used during assessment*

9 – Authentication Request Identified Informational Finding

Informational RISK (0/10)	
Exploitation Likelihood	**Informational**
Business Impact	**Easy**
Remediation Difficulty	**Informational**

Security Implications

The given request has been identified as an authentication request. This alert indicates that a request is being recognized as one associated with authentication, and that the system may have recognized the user and password parameters used in the request. Specifically, the request contains a set of key=value lines that help identify relevant fields such as the user and password.

If the context has the Authentication Method set to "Auto-Detect", the rule will automatically adapt to the identified authentication mechanism based on the parameters in the request.

Analysis

The following request was identified as an authentication request:

- **URL**: http://testphp.vulnweb.com/secured/newuser.php
- **Method**: POST
- **Parameters**:
  - uemail
  - upass
- **Other Info**:
  - userParam=uemail userValue=UxxgWSSs passwordParam=upass referer=http://testphp.vulnweb.com/signup.php
  - userParam=uemail userValue=ZAP passwordParam=upass referer=http://testphp.vulnweb.com/signup.php

Instances Identified: 2

URLs Affected:

- http://testphp.vulnweb.com/secured/newuser.php

Impact:

This is an **informational** alert and does not present a vulnerability. It simply indicates that the authentication mechanism has been identified based on the request parameters. No security issues are caused by this observation.

APPENDIX E- TOOLS USED

TOOL	***DESCRIPTION***
Zap	*Scan tool*

Table E.5: *Tools used during assessment*

APPENDIX A - TOOLS USED

TOOL	DESCRIPTION
BurpSuite Community Edition	Used for testing of web applications.
Metasploit	Used for exploitation of vulnerable services and vulnerability scanning.
Nmap	Used for scanning ports on hosts.
OpenVAS	Used to scan the networks for vulnerabilities.
PostgreSQL Client Tools	Used to connect to the PostgreSQL server.

Table A.1: Tools used during assessment

10 – Charset Mismatch (Header Versus Meta Content-Type Charset) Informational Finding

Informational RISK (0/10)	
Exploitation Likelihood	**Informational**
Business Impact	**Easy**
Remediation Difficulty	**Informational**

Security Implications

This check identifies responses where the **HTTP Content-Type header** declares a **charset** that is different from the charset defined in the **META content-type** tag of the HTML or XML document. When there is a mismatch between the charset in the HTTP header and the charset declared in the META tag, web browsers may be forced into undesirable content-sniffing behavior in an attempt to detect the correct character set for the content.

An attacker could exploit this mismatch by injecting content in a specific encoding (e.g., UTF-7), potentially manipulating some browsers into interpreting the content differently, which could lead to a variety of issues such as script injection or misinterpretation of the page's content.

Analysis

The following requests were identified with a charset mismatch between the HTTP Header and the META content-type encoding declarations:

- **URL**: http://testphp.vulnweb.com/
  - **HTTP Header Charset**: UTF-8
  - **META Charset**: iso-8859-2
- **URL**: http://testphp.vulnweb.com/AJAX/index.php
  - **HTTP Header Charset**: UTF-8
  - **META Charset**: iso-8859-1
- **URL**: http://testphp.vulnweb.com/artists.php

    - **HTTP Header Charset**: UTF-8
    - **META Charset**: iso-8859-2
- **URL**: http://testphp.vulnweb.com/artists.php?artist=1
    - **HTTP Header Charset**: UTF-8
    - **META Charset**: iso-8859-2
- **URL**: http://testphp.vulnweb.com/artists.php?artist=2
    - **HTTP Header Charset**: UTF-8
    - **META Charset**: iso-8859-2
- **URL**: http://testphp.vulnweb.com/artists.php?artist=3
    - **HTTP Header Charset**: UTF-8
    - **META Charset**: iso-8859-2
- **URL**: http://testphp.vulnweb.com/cart.php
    - **HTTP Header Charset**: UTF-8
    - **META Charset**: iso-8859-2
- **URL**: http://testphp.vulnweb.com/categories.php
    - **HTTP Header Charset**: UTF-8
    - **META Charset**: iso-8859-2
- **URL**: http://testphp.vulnweb.com/disclaimer.php
    - **HTTP Header Charset**: UTF-8
    - **META Charset**: iso-8859-2
- **URL**: http://testphp.vulnweb.com/guestbook.php
    - **HTTP Header Charset**: UTF-8
    - **META Charset**: iso-8859-2
- **URL**: http://testphp.vulnweb.com/index.php
    - **HTTP Header Charset**: UTF-8
    - **META Charset**: iso-8859-2

- **URL**: http://testphp.vulnweb.com/listproducts.php?artist=1
    - **HTTP Header Charset**: UTF-8
    - **META Charset**: iso-8859-2
- **URL**: http://testphp.vulnweb.com/listproducts.php?artist=2
    - **HTTP Header Charset**: UTF-8
    - **META Charset**: iso-8859-2
- **URL**: http://testphp.vulnweb.com/listproducts.php?artist=3
    - **HTTP Header Charset**: UTF-8
    - **META Charset**: iso-8859-2
- **URL**: http://testphp.vulnweb.com/listproducts.php?cat=1
    - **HTTP Header Charset**: UTF-8
    - **META Charset**: iso-8859-2
- **URL**: http://testphp.vulnweb.com/listproducts.php?cat=2
    - **HTTP Header Charset**: UTF-8
    - **META Charset**: iso-8859-2
- **URL**: http://testphp.vulnweb.com/listproducts.php?cat=3
    - **HTTP Header Charset**: UTF-8
    - **META Charset**: iso-8859-2
- **URL**: http://testphp.vulnweb.com/listproducts.php?cat=4
    - **HTTP Header Charset**: UTF-8
    - **META Charset**: iso-8859-2
- **URL**: http://testphp.vulnweb.com/login.php
    - **HTTP Header Charset**: UTF-8
    - **META Charset**: iso-8859-2
- **URL**: http://testphp.vulnweb.com/product.php?pic=1
    - **HTTP Header Charset**: UTF-8

- **META Charset**: iso-8859-2

URLs Affected:

- http://testphp.vulnweb.com/
- http://testphp.vulnweb.com/AJAX/index.php
- http://testphp.vulnweb.com/artists.php
- http://testphp.vulnweb.com/artists.php?artist=1
- http://testphp.vulnweb.com/artists.php?artist=2
- http://testphp.vulnweb.com/artists.php?artist=3
- http://testphp.vulnweb.com/cart.php
- http://testphp.vulnweb.com/categories.php
- http://testphp.vulnweb.com/disclaimer.php
- http://testphp.vulnweb.com/guestbook.php
- http://testphp.vulnweb.com/index.php
- http://testphp.vulnweb.com/listproducts.php?artist=1
- http://testphp.vulnweb.com/listproducts.php?artist=2
- http://testphp.vulnweb.com/listproducts.php?artist=3
- http://testphp.vulnweb.com/listproducts.php?cat=1
- http://testphp.vulnweb.com/listproducts.php?cat=2
- http://testphp.vulnweb.com/listproducts.php?cat=3
- http://testphp.vulnweb.com/listproducts.php?cat=4
- http://testphp.vulnweb.com/login.php
- http://testphp.vulnweb.com/product.php?pic=1

Impact:

While this is an informational alert and does not directly indicate a vulnerability, it can lead to security concerns if an attacker is able to inject malicious content using an encoding of their choice. For example, if an attacker controls the content at the start of the page, they could use a misinterpreted charset to inject malicious scripts or payloads.

APPENDIX E- TOOLS USED

TOOL	***DESCRIPTION***
Zap	*Scan tool*

Table E.5: *Tools used during assessment*

APPENDIX B - ENGAGEMENT INFORMATION

Client Information

Client	Effigo

Contact Information

Name	Veeresh Akki
Phone	8197471529
Email	veeresh.akki@effigo.tech