



# Super Secret Company Web Penetration Testing Report

Business Confidential

*Date: June 13<sup>th</sup>, 2022*  
*Project: 001-22*  
*Version 1.0*

---

# Table of Contents

Table of Contents .....	2
Confidentiality Statement.....	3
Disclaimer.....	3
Contact Information.....	3
Assessment Overview.....	4
Assessment Components.....	4
Web Application Penetration Test .....	4
Finding Severity Ratings .....	4
Risk Factors.....	5
Likelihood .....	5
Impact.....	6
Scope.....	6
Scope Exclusions .....	6
Executive Summary .....	7
Testing Summary .....	7
Security Strengths and Weaknesses.....	7
Security Recommendations .....	7
Vulnerabilities Summary.....	8
Web Application Penetration Test Findings .....	8
Technical Findings.....	9
Web Application Penetration Test Findings .....	9
Finding WPT-001: Local File Inclusion – About Page.....	9
Finding WPT-002: Local File Inclusion – Members Page .....	10
Finding WPT-003: Weak Login Credential .....	11
Finding WPT-004: SQL Injection.....	12
Finding WPT-005: Reflected XSS – Contact Page.....	13
Finding WPT-006: Reflected XSS – Login Page.....	15
Finding WPT-007: Unencrypted Communications .....	16
Finding WPT-008: Cleartext Submission of Password .....	17
Finding WPT-009: Directory Listing.....	18
Finding WPT-010: HTTP Header Information Disclosure .....	19
Additional Scans and Reports.....	20

## Confidentiality Statement

This document is the exclusive property of Super Secret Company as SSC and Hacker Otodidak as HO. This document contains proprietary and confidential information. Duplication, redistribution, or use, in whole or in part, in any form, requires the consent of both SSC and HO.

HO may share this document with auditors under non-disclosure agreements to demonstrate penetration test requirement compliance.

## Disclaimer

A penetration test is considered a snapshot in time. The findings and recommendations reflect the information gathered during the assessment and not any changes or modifications made outside of that period.

Time-limited engagements do not allow for a full evaluation of all security controls. HO prioritized the assessment to identify the weakest security controls an attacker would exploit. HO recommends conducting similar assessments on an annual basis by internal or third-party assessors to ensure the continued success of the controls.

## Contact Information

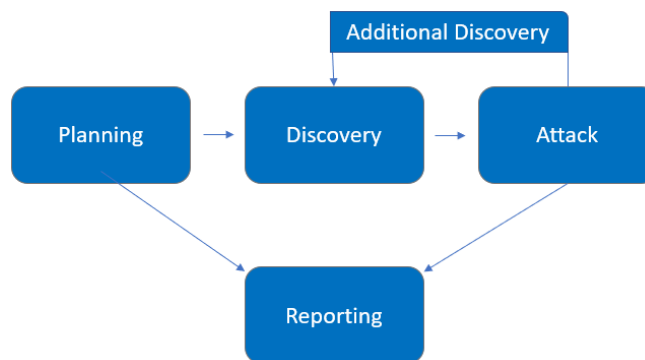
Name	Title	Contact Information
<b>Super Secret Company</b>		
John Smith	VP, Information Security (CISO)	Office: (555) 555-5555 Email: <a href="mailto:john.smith@supersecret.com">john.smith@supersecret.com</a>
Jim Smith	IT Manager	Office: (555) 555-5555 Email: <a href="mailto:jim.smith@supersecret.com">jim.smith@supersecret.com</a>
Joe Smith	Network Engineer	Office: (555) 555-5555 Email: <a href="mailto:joe.smith@supersecret.com">joe.smith@supersecret.com</a>
<b>Hacker Otodidak</b>		
Elliot Alderson	Lead Penetration Tester	Office: (555) 555-5555 Email: <a href="mailto:elliott@hackerotodidak.com">elliott@hackerotodidak.com</a>

## Assessment Overview

From June 13<sup>th</sup>, 2022 to July 13<sup>th</sup>, 2022, SSC engaged HO to evaluate the security posture of its web application compared to current industry best practices including a web application penetration test. All testing performed is based on the NIST SP 800-115 *Technical Guide to Information Security Testing and Assessment*, OWASP *Testing Guide (v4)*, and customized testing frameworks.

Phases of penetration testing activities include the following:

- Planning – Customer goals are gathered and rules of engagement obtained.
- Discovery – Perform scanning and enumeration to identify potential vulnerabilities, weak areas, and exploits.
- Attack – Confirm potential vulnerabilities through exploitation and perform additional discoveries upon new access.
- Reporting – Document all found vulnerabilities and exploits, failed attempts, and company strengths and weaknesses.



## Assessment Components

### Web Application Penetration Test

A web application penetration test is an in-depth penetration test on both the unauthenticated and authenticated portions of your application. The engineer will test for OWASP Top-10 critical security flaws along with a variety of other potential vulnerabilities based on security best practices.

Activities include site mapping and enumeration, automated and manual injection testing, directory traversal testing, malicious file uploads, remote code execution, password attacks, authentication bypasses, session attacks, and other testing depending on specific site content and languages.

### Finding Severity Ratings

The following table defines levels of severity and the corresponding CVSS v3.1 score range that are used throughout the document to assess vulnerability and risk impact.

Severity	CVSS v3.1 Score Range	Definition
Critical	9.0-10.0	Exploitation is straightforward and usually results in system-level compromise. It is advised to form a plan of action and patch immediately.
High	7.0-8.9	Exploitation is more difficult but could cause elevated privileges and potentially a loss of data or downtime. It is advised to form a plan of action and patch as soon as possible.
Medium	4.0-6.9	Vulnerabilities exist but are not exploitable or require extra steps such as social engineering. It is advised to form a plan of action and patch after high-priority issues have been resolved.
Low	0.1-3.9	Vulnerabilities are non-exploitable but would reduce an organization's attack surface. It is advised to form a plan of action and patch during the next maintenance window.
Informational	N/A	No vulnerability exists. Additional information is provided regarding items noticed during testing, strong controls, and additional documentation.

## Risk Factors

Risk is measured by two factors: Likelihood and Impact:

### Likelihood

Likelihood measures the potential of a vulnerability being exploited. Ratings are given based on the difficulty of the attack, the available tools, attacker skill level, and client environment.

Likelihood	Definition
High	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.
Medium	Exploitation methods are well-known and may be performed using public tools, but require configuration. Understanding the underlying system is required for successful exploitation.
Low	Exploitation requires a deep understanding of the underlying systems or advanced technical skills. Precise conditions may be required for successful exploitation.

## Impact

Impact measures the potential vulnerability's effect on operations, including confidentiality, integrity, and availability of client systems and/or data, reputational harm, and financial loss.

Likelihood	Definition
High	Successful exploitation may result in large disruptions of critical business functions across the organization and significant financial damage.
Medium	Successful exploitation may cause significant disruptions to non-critical business functions.
Low	Successful exploitation may affect a few users, without causing much disruption to routine business functions.

## Scope

Assessment	Details
Web Application Penetration Test	<a href="http://supersecret.company">http://supersecret.company</a>

## Scope Exclusions

Per client request, HO did not perform any of the following attacks during testing:

- Targeting any host and network service aside from provided target host and service
- Fully automated tests

## Executive Summary

HO evaluated SSC's web application security posture through penetration testing from June 13<sup>th</sup>, 2022 to July 13<sup>th</sup>, 2022. The following sections provide a high-level overview of vulnerabilities discovered, successful and unsuccessful attempts, and strengths and weaknesses.

## Testing Summary

HO evaluated SSC's web application from June 13<sup>th</sup>, 2022 to July 13<sup>th</sup>, 2022. The assessment included manual and semi-automatic testing on the unauthenticated web application (HTTP service). The methodology followed the OWASP testing guidelines and framework, which allowed for the thorough discovery of vulnerabilities. During the assessment, HO discovered several High findings including local file inclusion and weak login credential issue on login functionality. HO then found medium findings including SQL injection and reflected XSS. HO also discovered several low and informational issues related to web application security best practices: unencrypted communications, cleartext submission of passwords, directory listing, and HTTP header information disclosure. Overall, HO found the application was not well developed and exposed to several high-severity attacks.

## Security Strengths and Weaknesses

HO identified the following strengths which greatly increase SSC's security:

- HTTPOnly cookie flag already used to ensure that cookies cannot be accessed by client-side scripts to reduce the impact of XSS vulnerabilities

HO identified the following weaknesses which greatly decrease SSC's security:

- Password policy found to be insufficient
- Insufficient input validation mechanism

## Security Recommendations

In addition to the recommendations specified in each of the technical findings, HO recommends SSC take the following actions as soon as possible to minimize business risk:

- Review password policy
- Review input validation mechanism in all SSC's web application functionalities

## Vulnerabilities Summary

The following tables illustrate the vulnerabilities found by impact and recommended remediations as well as a report card grade compared to companies of similar size and infrastructure:

### Web Application Penetration Test Findings

0	3	3	2	2
Critical	High	Moderate	Low	Informational

Finding	Severity	Recommendation
WAPT-001: Local File Inclusion - About Page	High	Performing whitelisting on the parameter value, by matching it against a list of permitted files
WAPT-002: Local File Inclusion - Members Page	High	Performing whitelisting on the parameter value, by matching it against a list of permitted files
WAPT-003: Weak Login Credential	High	Use a strong password policy and unguessable username on the login
WAPT-004: SQL Injection	Medium	Use parameterized queries (prepared statements) when dealing with SQL queries that contain user input
WAPT-005: Reflected XSS - Contact Page	Medium	Sanitizing inputs, encode and escape any characters, implement Content Security Policy (CSP) and X-XSS-Protection header
WAPT-006: Reflected XSS - Login Page	Medium	Sanitizing inputs, encode and escape any characters, implement Content Security Policy (CSP) and X-XSS-Protection header
WAPT-007: Unencrypted Communications	Low	Use transport-level encryption (SSL/TLS) and Strict-Transport-Security HTTP header
WAPT-008: Cleartext Submission of Password	Low	Use encrypted connection (HTTPS) to transfer user credentials
WAPT-009: Directory Listing	Informational	Restrict directory listings from the web server configuration
WAPT-010: HTTP Header Information Disclosure	Informational	Modify the HTTP headers of the webserver to not disclose detailed information about the underlying web server



# Technical Findings

## Web Application Penetration Test Findings

### Finding WPT-001: Local File Inclusion – About Page

<b>Description:</b>	HO was able to trick the value of the “?file=” parameter on the about page into exposing sensitive files on the web server like /etc/passwd file. This occurs because web application uses the path to a file as input and treats this input as trusted.
<b>Risk:</b>	<b>High</b> - Likelihood: Medium, Impact: High
<b>Location:</b>	http://supersecret.company/about.php?file=
<b>Impact:</b>	An attacker can retrieve /etc/passwd file and source code of all web files including config file credentials (config.php)
<b>References:</b>	<a href="#">CWE-22: Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')</a> <a href="#">OWASP - PHP File Inclusion</a>

### Proof of Concept

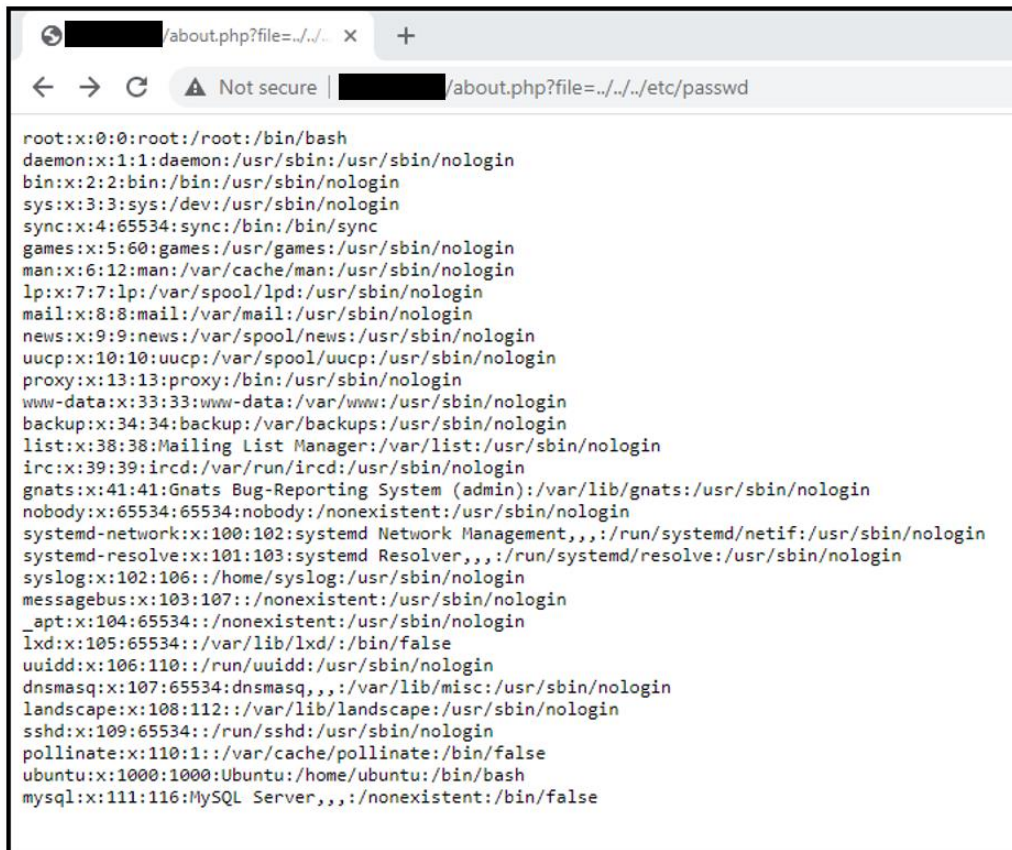


Figure 1: Captured disclosed /etc/passwd file on about page

## Remediation

Never used untrusted data to form a file location to be included. Validate the data to ensure that the supplied value for a file is permitted by performing whitelisting on the parameter value, by matching it against a list of permitted files. If the supplied value does not match any value in the whitelist, then the server should redirect to a standard error page.

### Finding WPT-002: Local File Inclusion – Members Page

<b>Description:</b>	HO was able to trick the value of the “?view=” parameter on the members page into exposing sensitive files on the web server like /etc/passwd file. This occurs because web application uses the path to a file as input and treats this input as trusted.
<b>Risk:</b>	<b>High</b> - Likelihood: Medium, Impact: High
<b>Location:</b>	http://supersecret.company/members.php?view=
<b>Impact:</b>	An attacker who can guess the weak credentials can access the admin access on the web
<b>References:</b>	<a href="#">CWE-22: Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')</a> <a href="#">OWASP - PHP File Inclusion</a>

## Proof of Concept



Figure 2: Captured disclosed /etc/passwd file on the member's page

## Remediation

Never used untrusted data to form a file location to be included. Validate the data to ensure that the supplied value for a file is permitted by performing whitelisting on the parameter value, by matching it against a list of permitted files. If the supplied value does not match any value in the whitelist, then the server should redirect to a standard error page.

## Finding WPT-003: Weak Login Credential

<b>Description:</b>	Login mechanism using a weak password and guessable username. HO was able to guess the credentials required to access the member's page. A weak password is short, common, a system default, or something that could be rapidly guessed by executing a brute force attack using a subset of all possible passwords, such as words in the dictionary, proper names, words based on the user name or common variations on these themes.
<b>Risk:</b>	<b>High</b> - Likelihood: High, Impact: High
<b>Location:</b>	http://supersecret.company/index.php
<b>Impact:</b>	An attacker who can guess the weak credentials can access the member's page and see restricted information from the authorized pages.
<b>References:</b>	<a href="#">CWE-521: Weak Password Requirements</a> <a href="#">OWASP A2:2017-Broken Authentication</a>

### Proof of Concept

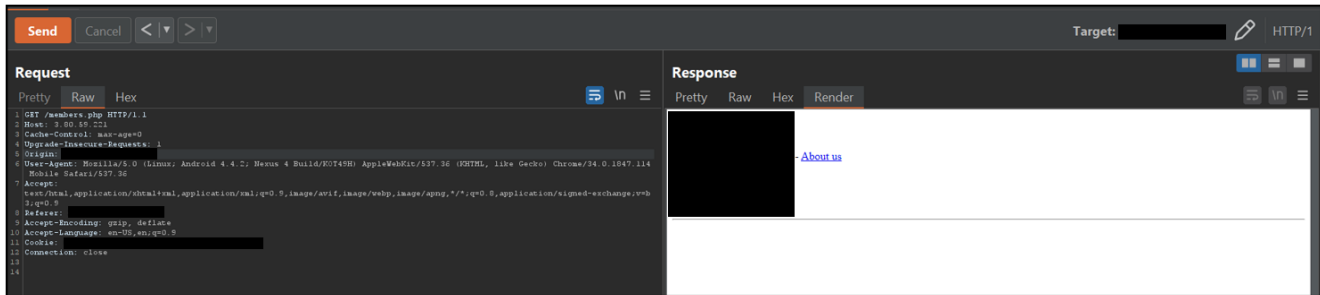


Figure 3: Captured successful login with admin:admin credential

An attacker can log in with these known credentials:

- Username: **admin**
- Password: **admin**

### Remediation

Enforce a strong password policy. Don't permit guessable usernames or weak passwords based on dictionary words.

Finding WPT-004: SQL Injection

<b>Description:</b>	HO was able to inject malicious SQL statements that control a web application's database server. By using SQLMap, HO successfully gained all usernames and passwords of the web application's users.
<b>Risk:</b>	<b>Medium</b> - Likelihood: Medium, Impact: High
<b>Location:</b>	http://supersecret.company/members.php?view=article.php&id=
<b>Impact:</b>	The attacker can retrieve all data from the web application's database server and gain access to all the user accounts of the application
<b>References:</b>	<a href="#">CWE-89: Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')</a> <a href="#">A03 Injection - OWASP Top 10:2021</a>

Proof of Concept

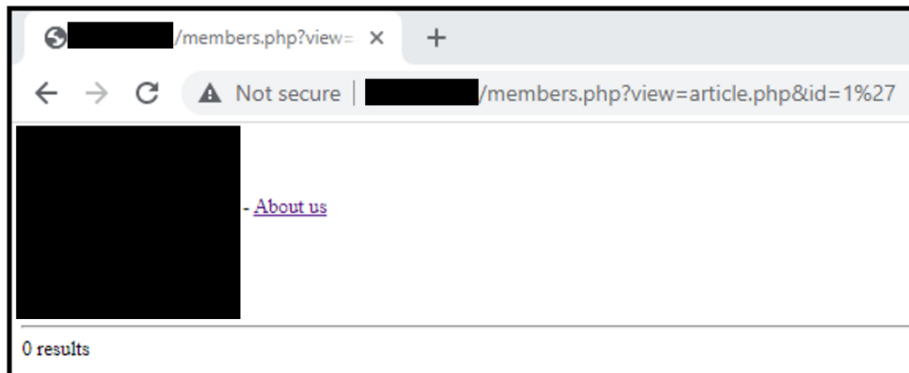


Figure 4: Captured different responses when inserted quote (') on id parameter

```

(root@kali)~# sqlmap -r sql1.txt -t users -D [redacted] --dump
[04:44:51] [INFO] parsing HTTP request from 'sql1.txt'
[04:44:51] [INFO] resuming back-end DBMS 'mysql'
[04:44:51] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
-----
Parameter: id (GET)
  Type: boolean-based blind
  Title: AND boolean-based blind - WHERE or HAVING clause
  Payload: view-article.php?id=1 AND 4942=4942

  Type: time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
  Payload: view-article.php?id=1 AND (SELECT 8011 FROM (SELECT(SLEEP(5)))vPMG)

  Type: UNION query
  Title: Generic UNION query (NULL) - 3 columns
  Payload: view-article.php?id=1 UNION ALL SELECT NULL,NULL,CONCAT(0x71787a7871,0x49787177471694b504e4559754a44d5a68774b48784442547864506f4e566c786f4d6549715179,0x71627a7171)--

[04:44:52] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu 18.04 (bionic)
web application technology: Apache 2.4.29
back-end DBMS: MySQL >= 5.0.12
[04:44:52] [INFO] fetching columns for table [redacted] in database [redacted]
[04:44:52] [INFO] fetching entries for table [redacted] in database [redacted]
Database: [redacted]
Table: [redacted]
[2 entries]
-----
| id | email | phone | password | username | nomor_ktp |
-----
| 1 | victim@ | 8723465270 | admin | admin | 9990878765 |
| 2 | helpa | 1231312 | 123123 | helpdesk | 123123 |
-----

[04:44:52] [INFO] table [redacted] dumped to CSV file '/root/.local/share/sqlmap/output/3.80.59.221/dump/[redacted].csv'
[04:44:52] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/3.80.59.221'
[*] ending @ 04:44:52 /2022-06-13/
  
```

Figure 5: Captured the successfully SQL Injection exploitation with the SQLMap tool

## Remediation

Use parameterized queries (prepared statements) when dealing with SQL queries that contain user input. Parameterized queries allow the database to understand which parts of the SQL query should be considered as user input, therefore solving SQL injection.

## Finding WPT-005: Reflected XSS – Contact Page

<b>Description:</b>	HO was able to inject malicious javascript codes into a request of the contact form on the “form” parameter and have the server return the script to the client in the response without any filtered or escaped. This occurs because the application is taking untrusted data and reusing it without performing any validation or sanitization.
<b>Risk:</b>	<b>Medium</b> - Likelihood: Medium, Impact: Low
<b>Location:</b>	http://supersecret.company/contact.php
<b>Impact:</b>	An attacker can use this vulnerability to redirect a user to the attacker’s malicious site and trick the user to steal the user’s credential
<b>References:</b>	<a href="#">CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')</a> <a href="#">OWASP Top Ten 2017   A7:2017-Cross-Site Scripting (XSS)</a>

## Proof of Concept

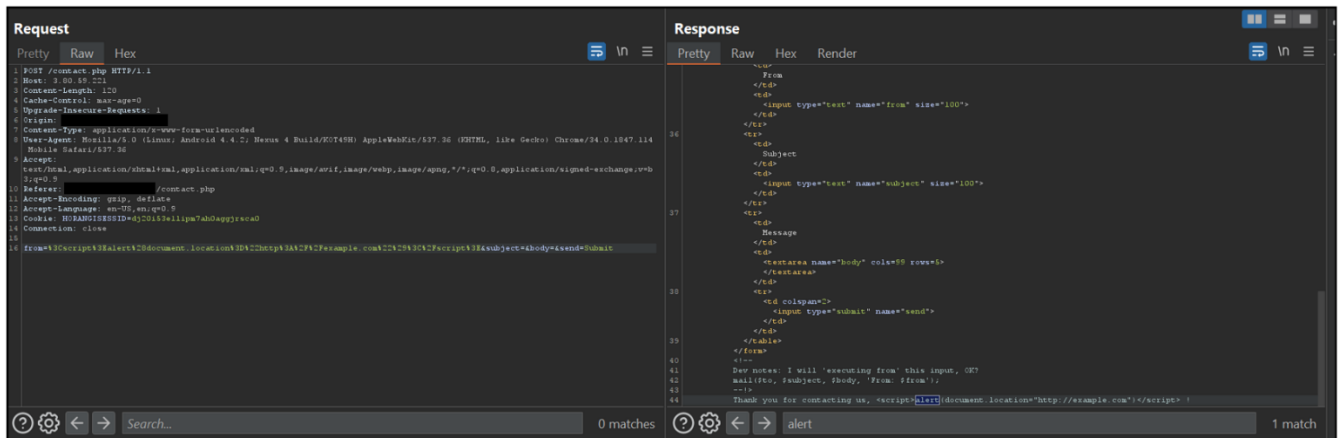


Figure 6: Captured the request and response of XSS payload execution

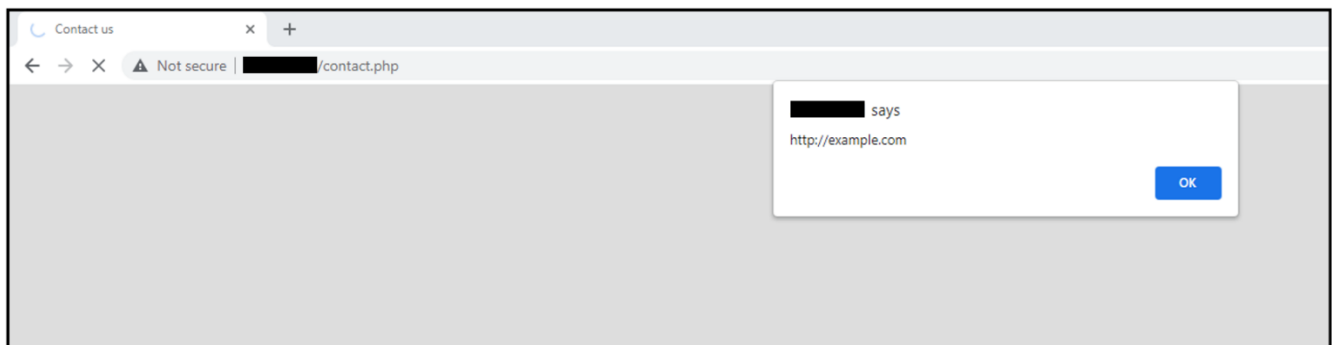


Figure 7: Captured the pop-up alert is successfully executed after sending XSS payload on the form

## Remediation

Sanitizing inputs with whitelist what is allowed, encode and escape any character that can affect the execution context, implement Content Security Policy (CSP) and X-XSS-Protection header.

**Finding WPT-006: Reflected XSS – Login Page**

<b>Description:</b>	HO was able to inject malicious javascript codes into a request of the contact form on the “error” parameter and have the server return the script to the client in the response without any filtered or escaped. This occurs because the application is taking untrusted data and reusing it without performing any validation or sanitization.
<b>Risk:</b>	<b>Medium</b> - Likelihood: Medium, Impact: Low
<b>Location:</b>	http://supersecret.company/index.php?error=
<b>Impact:</b>	An attacker can use this vulnerability to redirect a user to the attacker’s malicious site and trick the user to steal the user’s credential
<b>References:</b>	<a href="#">CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')</a> <a href="#">OWASP Top Ten 2017   A7:2017-Cross-Site Scripting (XSS)</a>

**Proof of Concept**

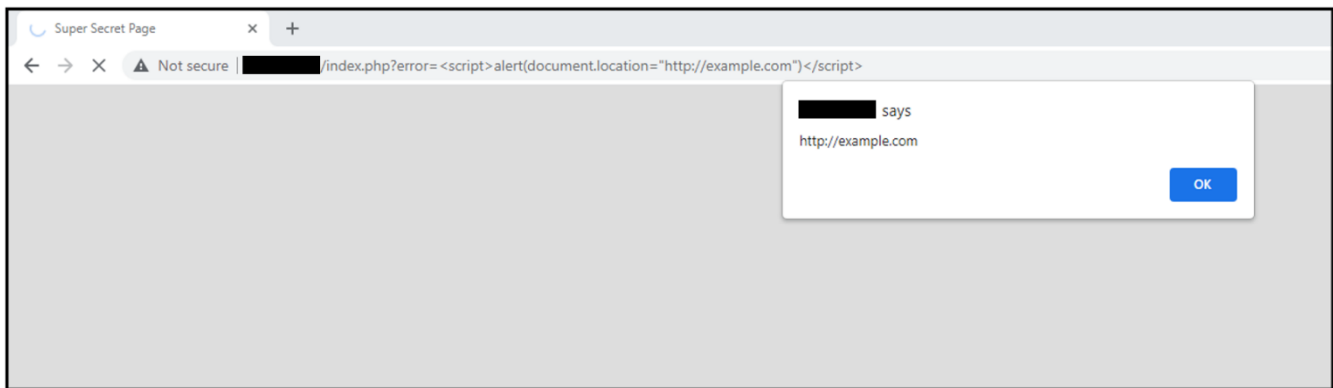


Figure 8: Captured the successfully pop-up alert execution of reflected XSS on login page error parameter

**Remediation**

Sanitizing inputs with whitelist what is allowed, encode and escape any character that can affect the execution context, implement Content Security Policy (CSP) and X-XSS-Protection header.

## Finding WPT-007: Unencrypted Communications

<b>Description:</b>	The application allows users to connect to it over unencrypted connections. An attacker suitably positioned to view a legitimate user's network traffic could record and monitor their interactions with the application and obtain any information the user supplies.
<b>Risk:</b>	<b>Low</b> - Likelihood: Low, Impact: Low
<b>Location:</b>	http://supersecret.company/*
<b>Impact:</b>	The attacker can eavesdrop on all communication and see any information that is being transmitted such as the login credentials. This issue may also trigger browser warnings about the insecurity of the connection
<b>References:</b>	<a href="#">CWE-326: Inadequate Encryption Strength</a> <a href="#">Security/Server Side TLS - MozillaWiki</a> <a href="#">Strict-Transport-Security - HTTP   MDN</a>

### Proof of Concept

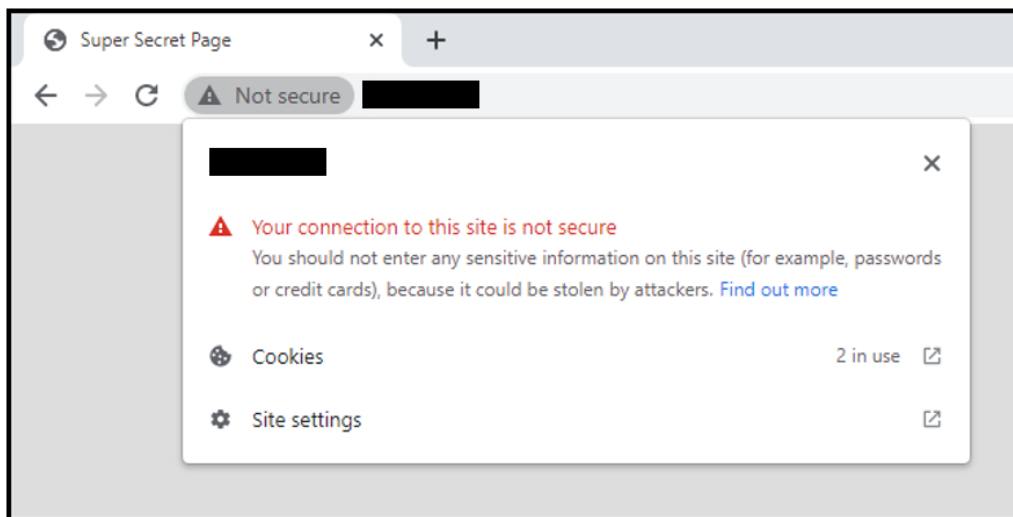


Figure 9: Captured the triggered browser warning about the insecurity of the connection

### Remediation

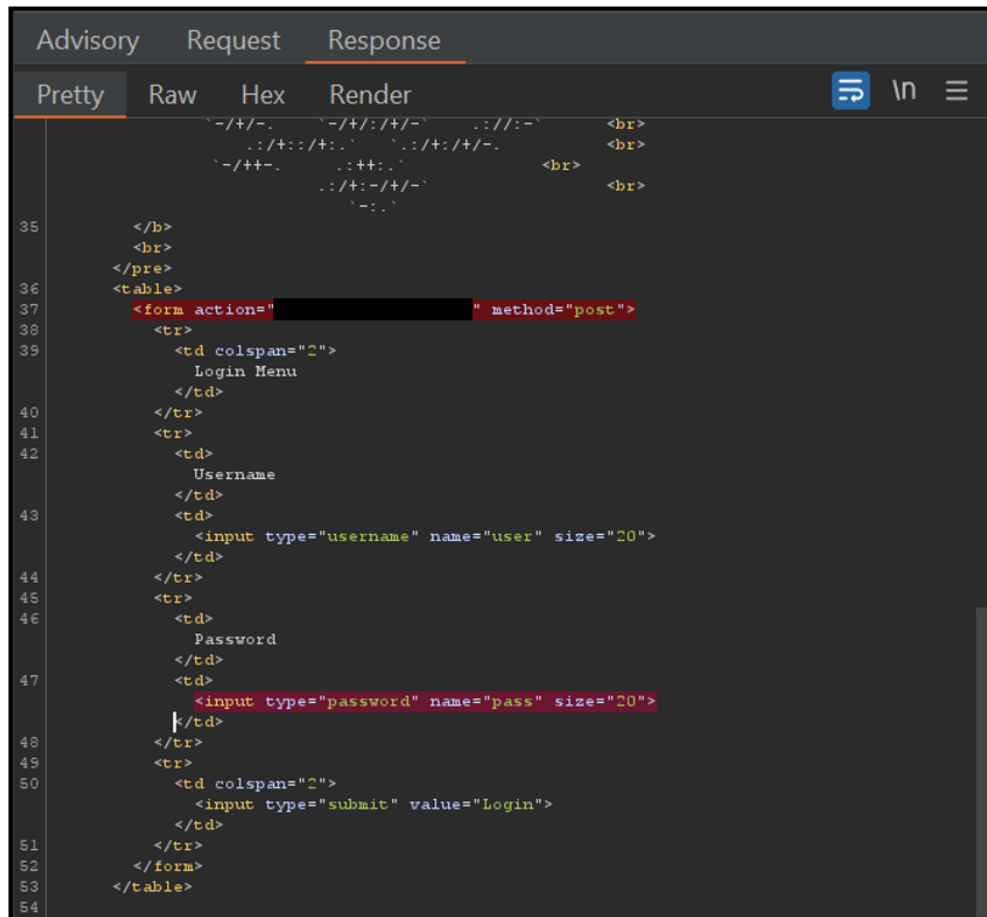
Applications should use transport-level encryption (SSL/TLS) to protect all communications passing between the client and the server. The Strict-Transport-Security HTTP header should be used to ensure that clients refuse to access the server over an insecure connection.



Finding WPT-008: Cleartext Submission of Password

<b>Description:</b>	User credentials are transmitted over an unencrypted channel. This information should always be transferred via an encrypted channel (HTTPS) to avoid being intercepted by malicious users.
<b>Risk:</b>	<b>Low</b> - Likelihood: Low, Impact: Low
<b>Location:</b>	http://supersecret.company/index.php
<b>Impact:</b>	The attacker can use Man-in-The-Middle (MiTM) attacks to intercept and compromise credentials passed from the client (user) to the web application (server)
<b>References:</b>	<a href="#">CWE-319: Cleartext Transmission of Sensitive Information</a> <a href="#">CWE-523: Unprotected Transport of Credentials</a>

Proof of Concept



```

35 </b>
36 <br>
37 </pre>
38 <table>
39   <tr>
40     <td colspan="2">
41       Login Menu
42     </td>
43   </tr>
44   <tr>
45     <td>
46       Username
47     </td>
48     <td>
49       <input type="username" name="user" size="20">
50     </td>
51   </tr>
52   <tr>
53     <td>
54       Password
55     </td>
56     <td>
57       <input type="password" name="pass" size="20">
58     </td>
59   </tr>
60   <tr>
61     <td colspan="2">
62       <input type="submit" value="Login">
63     </td>
64   </tr>
65 </table>

```

Figure 10: The form action value is in HTTP that makes password sending in plaintext

Remediation

Because user credentials are considered sensitive information, should always be transferred to the server over an encrypted connection (HTTPS).

Finding WPT-009: Directory Listing

<b>Description:</b>	SSC allowed directory listing on the “/upload” endpoint that can be accessed by the public. Directory listing is a web server function that displays the directory contents when there is no index file in a specific website directory.
<b>Risk:</b>	<b>Informational</b> - Likelihood: Low, Impact: Low
<b>Location:</b>	http://supersecret.company/uploads/
<b>Impact:</b>	The attacker will utilize the presence of directory listing to discover sensitive files, download protected content, or even just learn how the web application is structured.
<b>References:</b>	<a href="#">CWE-548: Exposure of Information Through Directory Listing</a> <a href="#">OWASP Periodic Table of Vulnerabilities - Directory Indexing</a>

Proof of Concept

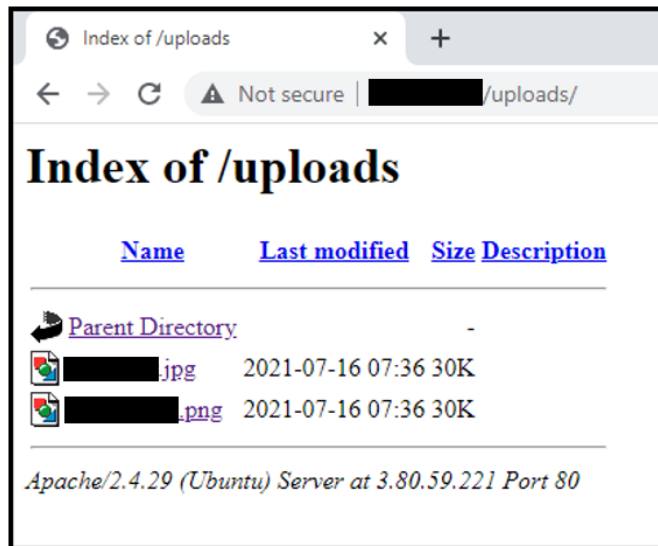


Figure 11: Directory listing enabled on the uploads directory

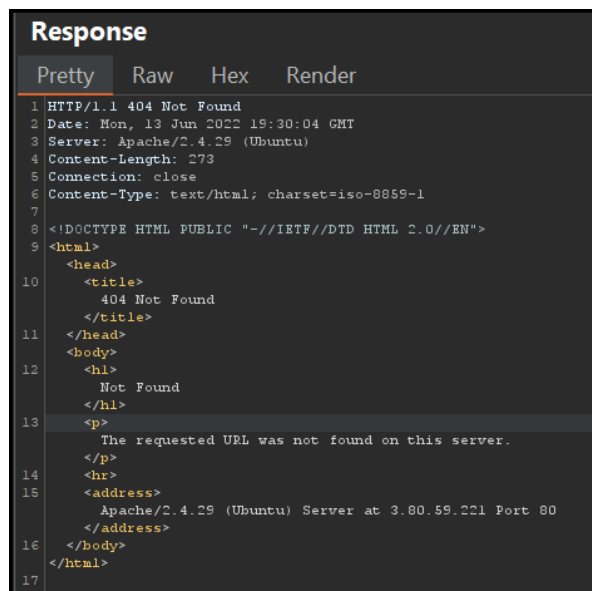
Remediation

Make sure no sensitive information is disclosed on the directory listing or restrict directory listings from the web server configuration.

Finding WPT-010: HTTP Header Information Disclosure

<b>Description:</b>	The HTTP responses returned by the remote web server include a header named “Server” that disclosed information about what kind of web server (version) and the server operating system information
<b>Risk:</b>	<b>Informational</b> - Likelihood: Low, Impact: Low
<b>Location:</b>	http://supersecret.company/profile.php
<b>Impact:</b>	The attacker can use this disclosed information to search for common vulnerabilities (CVE) and public exploits to gain access to the system
<b>References:</b>	<a href="#">CWE - CWE-200: Exposure of Sensitive Information to an Unauthorized Actor</a> <a href="#">The Web Application Security Consortium / Fingerprinting</a>

Proof of Concept



```

Response
Pretty Raw Hex Render
1 HTTP/1.1 404 Not Found
2 Date: Mon, 13 Jun 2022 19:30:04 GMT
3 Server: Apache/2.4.29 (Ubuntu)
4 Content-Length: 273
5 Connection: close
6 Content-Type: text/html; charset=iso-8859-1
7
8 <!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
9 <html>
10 <head>
11 <title>
12 404 Not Found
13 </title>
14 </head>
15 <body>
16 <h1>
17 Not Found
18 </h1>
19 <p>
20 The requested URL was not found on this server.
21 </p>
22 <hr>
23 <address>
24 Apache/2.4.29 (Ubuntu) Server at 3.80.59.221 Port 80
25 </address>
26 </body>
27 </html>
    
```

Figure 12: HTTP response disclosed web server & operating system information

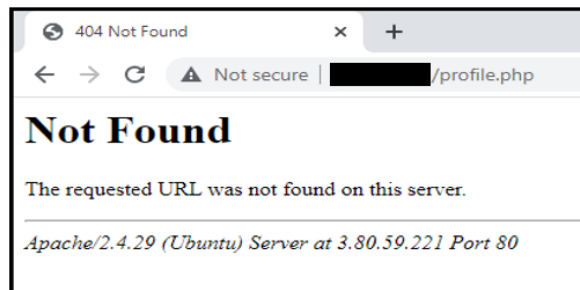


Figure 13: Information disclosure that showed on the browser response

Remediation

Modify the HTTP headers of the webserver to not disclose detailed information about the underlying web server.

## Additional Scans and Reports

HO provides all clients with all report information gathered during testing. This includes vulnerability scans and a detailed findings. For more information, please see the following documents:

Web Application:

- [SSC\\_Web\\_Unauthenticated\\_Scan.html](#)
- [SSC\\_Web\\_Authenticated\\_Scan.html](#)

hacker  otodidak

Last Page