WEB Application Penetration Testing Report for

{REDACTED CLIENT}

Findings, Attack Narrative, and Recommendations

31/03/2024

Table of Contents

Executive Summary

This report presents the results of the White Box & Black Box penetration testing for the {REDACTED CLIENT}. The recommendations provided in this report are structured to facilitate remediation of the identified security risks. This document serves as a formal letter of attestation for the recent Arthentication Service penetration testing. Evaluation ratings compare information gathered during the engagement to “best in class” criteria for security standards. We believe that the statements made in this document provide an accurate assessment of the current security of the Authentication.

We highly recommend reviewing the Summary section of business risks and High-Level Recommendations to better understand risks and discovered security issues

The intent of an application assessment is to dynamically identify and assess the impact of potential security vulnerabilities within the application. During this assessment, manual testing tools and techniques were employed to discover and exploit possible vulnerabilities.

Testing was conducted from both an unauthenticated and authenticated context. Unauthenticated testing examines the exterior security posture of an application and looks for vulnerabilities that do not require authentication to exploit, while authenticated tests focus on discovering and exploiting vulnerabilities on portions of the internal application that are only accessible after successful authentication. Assessors were provided both a regular user and an administrative user account to assess the internal security controls of the application.

Introduction

1.1 Project Objectives

Our primary goal within this project was to provide **{REDACTED CLIENT}** with an understanding of the current level of security in the web application authentication. We completed the following objectives to accomplish this goal:

- Identifying authentication-based threats and vulnerabilities in the application
- Comparing current security measures with industry best practices.
- Providing recommendations that can be implemented to mitigate threats and vulnerabilities and meet industry best practices

The Common Vulnerability Scoring System (CVSS) version 3.0 was used to calculate the scores of the vulnerabilities found.

1.2 Scope & Timeframe

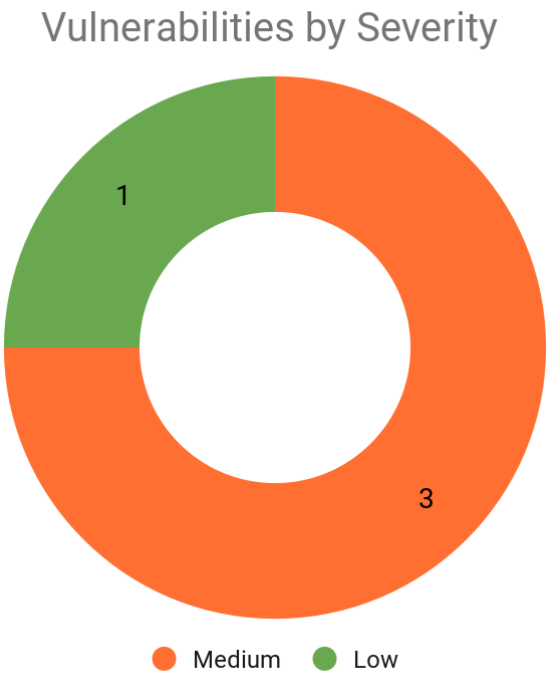
We conducted the tests using a staging (non-production) environment of **{REDACTED CLIENT}**. All other applications and servers were out of scope. The following hosts were considered to be in scope for testing.

1.2.1 Hostnames & IP-addresses

Scope:	Description:
https://{REDACTED HOST}	Merchant portal
https://{REDACTED HOST}	API & Payment Page

1.3 Summary of Findings

Our assessment of the **{REDACTED CLIENT}** web application revealed the following vulnerabilities



Security experts performed manual security testing according to the OWASP Web Application Testing Methodology, which demonstrates the following results.

Severity	Critical	High	Medium	Low	Informational
Number of issues	0	0	3	1	0

Severity scoring:

- Critical – Immediate threat to key business processes.

- High – Direct threat to key business processes.
- Medium – Indirect threat to key business processes or partial threat to business processes.

- Low – No direct threat exists. The vulnerability may be exploited using other vulnerabilities.
- Informational – This finding does not indicate vulnerability, but states a comment that notifies about design flaws and improper implementation that might cause a problem in the long run.

The exploitation of found vulnerabilities may cause full compromise of some services, stealing users' accounts, and gaining organisation's and users' sensitive information.

1.4 Summary of Business Risks

In the case of **{REDACTED CLIENT}** applications and related infrastructure

Critical severity issues can lead to:

- Disruption and unavailability of main services, which company provide to their users
- Prolonged recovery from backups phase as a result of a focused attack against internal infrastructure
- Company-wide ransomware attack with the following unavailability of certain parts of the infrastructure and possible financial loss due to insufficient security insurance

High severity issues can lead to:

- Usage of infrastructure for illegitimate activity (scanning of the internal network, Denial of Service attacks)
- Disclosure of confidential and Personally Identifiable Information
- Theft or exploitation of the credentials of a higher-level account

Medium severity issues can lead to:

- Disclosure of system components versions, logs and additional information about systems that might allow disgruntled employees or external malicious actors to misuse or download sensitive information outside of the company perimeter.

- Disclosure of confidential, sensitive and proprietary information related to users and companies which use CRM services

Low and **Informational** severity issues can lead to:

- Abusing business logic of main services to gain competitive advantage

- Unauthorised access to user or company confidential, private, or sensitive data

- Repudiation attacks against other users of services which allow maintaining plausible deniability

1.5 High-Level Recommendations

Taking into consideration all issues that have been discovered, we highly recommend to:

- Allow only trusted domains for CORS policy.
- Ensure that you have input validation and a strict whitelisting of allowed URLs to prevent SSRF vulnerabilities.
- Implement rigorous input validation and output encoding to sanitize user inputs and prevent XSS/HTML-injection vulnerabilities.

Technical Details

2.1 Methodology

Our Penetration Testing Methodology is grounded on the following guides and standards:

- OWASP Web Security Testing Guide

Open Web Application Security Project (OWASP) is an industry initiative for web application security. OWASP has identified the 10 most common attacks that succeed against APIs. Besides, OWASP has

created Application Security Verification Standard (ASVS) which helps to identify threats, provides a basis for testing web application technical security controls, and can be used to establish a level of confidence in the security of Web applications.

The **OWASP Web Security Testing Guide (WSTG)** is a comprehensive guide to testing the security of web applications and web services. Created by the collaborative efforts of security professionals and dedicated volunteers, the WSTG provides a framework of best practices used by penetration testers and organisations all over the world.

2.2 Security tools used

- Manual testing: Burp Suite Pro
- Vulnerability Scan: nikto
- Network Scan: nmap
- Directory enumeration: gobuster
- Injection testing tools: XSSHunter, SQLmap
- Secure Code Review: IntelliJ IDEA CE

2.3 Project limitations

The Assessment was conducted against a testing environment with all limitations it provides.

Findings Details

3.1 Medium severity findings

3.1.1 Insecure CORS configuration on {REDACTED HOST}

Vulnerability ID: 1
Vulnerability Category: Client Side Security Misconfiguration
Threat level: High

CWE	CWE-942
CVSS 3	8.1
Description	The product uses a cross-domain policy file that includes domains that should not be trusted.
Security Impact	A CORS misconfiguration can leave the application at a high risk of compromises resulting in an impact on the confidentiality and integrity of data by allowing third-party sites to carry out privileged requests through your website's authenticated users such as retrieving user setting information or saved payment card data.
Affected endpoint	/
Remediation	- Ensure that only "{REDACTED HOST}" is allowed.
External References	https://securityintelligence.com/posts/cors-how-to-use-and-secure-a-cors-policy-with-origin/

Finding Evidence

Auth service has a weak configuration of CORS. It is supposed to allow access for the frontend domain of merchant portal "{REDACTED HOST}", but due to insecure regexp the domain "{REDACTED HOST}" may have access to the site too. It's a critical vulnerability on the client side which may allow to hack a user account if he visits a malicious domain.

```
OPTIONS /auth/login HTTP/2
Host: [redacted]
Accept: */*
Access-Control-Request-Method: POST
Access-Control-Request-Headers: content-type
Origin: [redacted]
User-Agent: Mozilla/5.0 (Windows NT 10.0;
Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/122.0.6261.112 Safari/537.36
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-site
Sec-Fetch-Dest: empty
Referer: [redacted]
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Priority: u=1, i
```

```
1  HTTP/2 204 No Content
2  Date: Tue, 30 Mar 2024 15:40:50 GMT
3  Access-Control-Allow-Origin:
   [redacted]
4  Access-Control-Allow-Credentials: true
5  Access-Control-Allow-Methods: GET, POST, PUT,
   PATCH, DELETE, OPTIONS
6  Access-Control-Allow-Headers:
   Accept,Authorization,Cache-Control,Content-Type
   ,DNT,If-Modified-Since,Keep-Alive,Origin,User-A
   gent,X-Requested-With,Range
7  Access-Control-Max-Age: 1728000
8  Cf-Cache-Status: DYNAMIC
9  Server: cloudflare
10 Cf-Ray: 86e4aa8f1cae180f-ATL
11
```

Ensure that only "{REDACTED HOST}" is allowed.

3.1.2 Server Side Request Forgery (SSRF) on {REDACTED HOST}

Vulnerability ID: 2

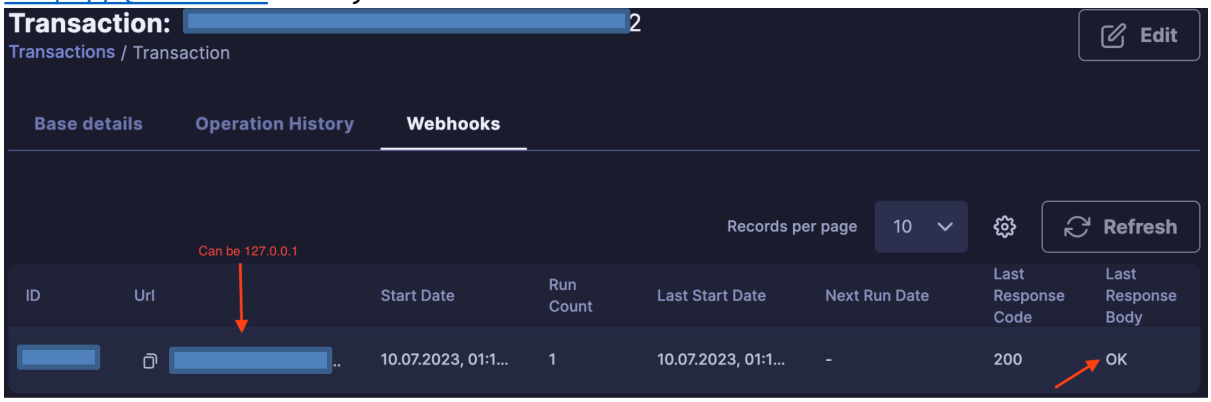
Vulnerability Category: Server Side Input Validation

Threat level: High

CWE	CWE-918
CVSS 3	7.4
Description	The merchant has the ability to set a webhook (URL) to be aware of transaction status. The problem is that the webhook URL can be a link to the internal host. The web server receives a URL and retrieves the contents of this URL, but it does not sufficiently ensure that the request is being sent to the expected destination.
Security Impact	This can lead to accessing sensitive data like cloud server metadata, database interfaces, or internal REST endpoints. Attackers exploit this vulnerability to bypass input validation and potentially extract confidential information or execute unauthorized actions within the system.
Affected endpoint	/
Remediation	- Ensure that webhook requests are sent from the host that doesn't have access to the hosts with sensitive information.
External References	https://cheatsheetseries.owasp.org/cheatsheets/Server_Side_Request_Forgery_Prevention_Cheat_Sheet.html

Finding Evidence

http://{REDACTED HOST} - resolves to 127.0.0.1



This way an attacker can make http requests to any internal host and be able to see responses. The service that makes requests should be out of the private network. Another option is to use a proxy that doesn't have access to a private network.

3.1.3 Reflected HTML injection via invalid requests

Vulnerability ID: 3

Vulnerability Category: Injection

Threat level: Medium

CWE	CWE-80
CVSS 3	5.8
Description	Web page parses HTML tags in error message which could lead to XSS
Security Impact	HTML injection could potentially lead to XSS which could affect client's data
Affected endpoint	/{REDACTED ENDPOINT}*
Remediation	- Ensure that error pages response with "application/json" Content-Type or at least they validate special characters used in HTML tags
External References	https://www.wallarm.com/what/html-injection

Finding Evidence

Request

Pretty Raw Hex Hackvertor

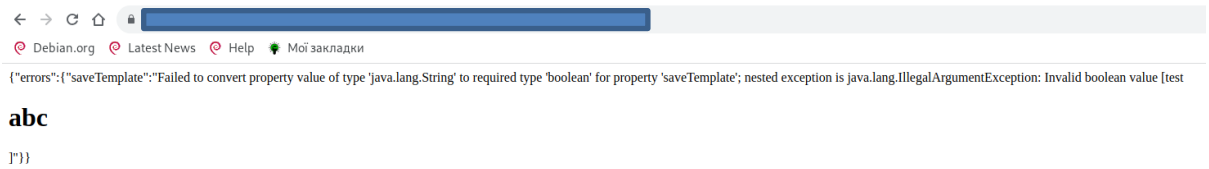
```
1 POST [redacted]
  HTTP/2
2 Host: [redacted]
3 Content-Length: 415
4 Sec-Ch-Ua: "Not)A;Brand";v="24", "Chromium";v="116"
5 Accept: application/json, text/plain, */*
6 Content-Type: application/x-www-form-urlencoded
7 Sec-Ch-Ua-Mobile: ?0
8 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like
   Gecko) Chrome/116.0.0.0 Safari/537.36
9 Sec-Ch-Ua-Platform: "Linux"
10 Origin: [redacted]
11 Sec-Fetch-Site: same-origin
12 Sec-Fetch-Mode: cors
13 Sec-Fetch-Dest: empty
14 Referer:
   [redacted]
15 Accept-Encoding: gzip, deflate, br
16 Accept-Language: en-US,en;q=0.9
17
18 _csrf=
   [redacted]
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/2 400 Bad Request
2 Date: Tue, 02 Apr 2024 14:42:59 GMT
3 Content-Type: text/html;charset=UTF-8
4 Cache-Control: no-cache, no-store, max-age=0, must-revalidate
5 Pragma: no-cache
6 Expires: 0
7 Content-Language: en-US
8 X-Version: 2.1180-master
9 [redacted]
10 Cf-Cache-Status: DYNAMIC
11 Server: cloudflare
12 Cf-Ray: 86e196b1db8e8876-WAW
13
14 {"errors":{"saveTemplate":"Failed to convert property value of type
   'java.lang.String' to required type 'boolean' for property 'saveTemplate';
   nested exception is java.lang.IllegalArgumentException: Invalid boolean value
   [test<h1>
     abc
   </h1>
```

As we can see, html tags passed in parameter saveTemplate are parsed and then displayed on a response page



3.2 Low severity findings

3.2.1 Vulnerable JavaScript library

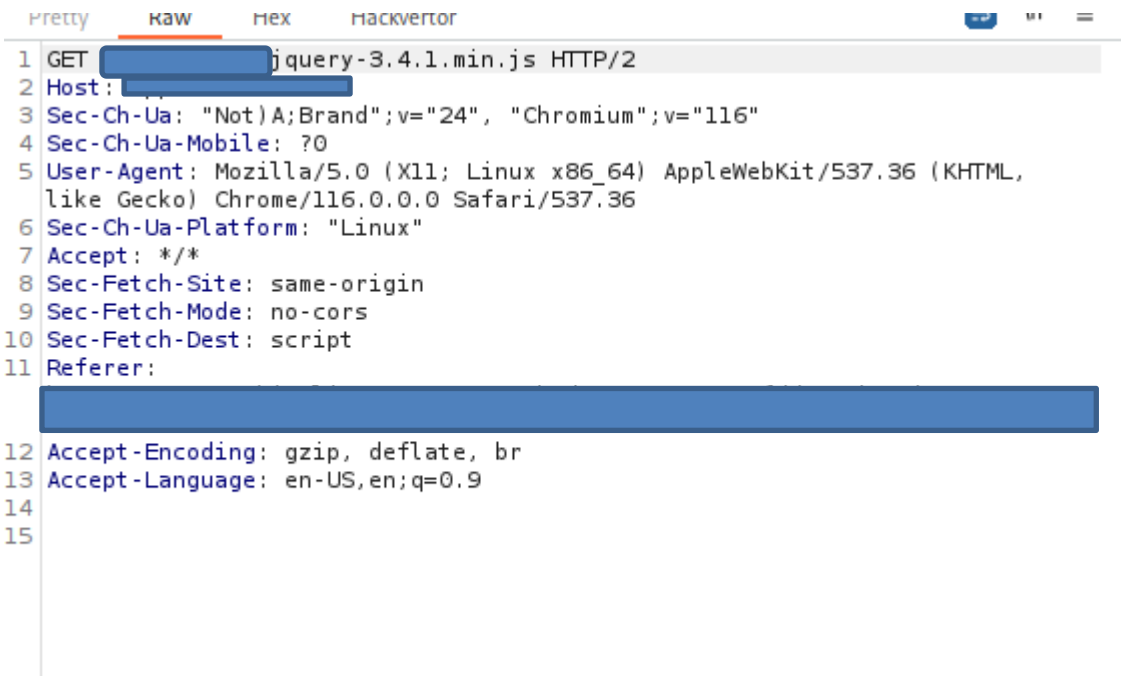
Vulnerability ID: 4
Vulnerability Category: Deprecated Component
Threat level: Low

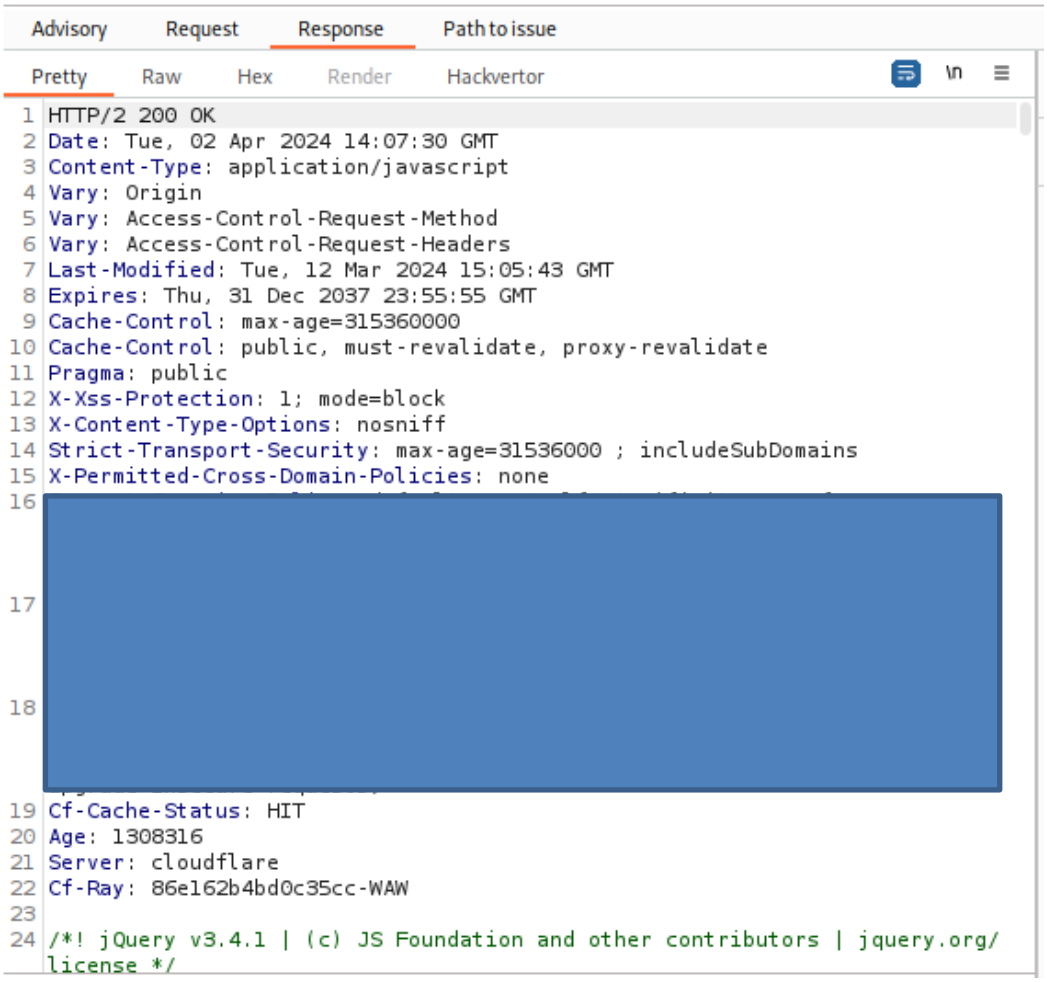
CWE	CWE-1104
CVSS 3	3.1
Description	Web page uses deprecated and vulnerable JavaScript library
Security Impact	Usage of deprecated libraries could lead to other vulnerabilities
Affected endpoint	/{REDACTED ENDPOINT}*
Remediation	- Ensure that library is updated and none outdated components are used
External References	https://security.snyk.io/package/npm/jquery/3.4.1

We detected **jquery** version **3.4.1.min**, which has the following vulnerabilities:

- CVE-2020-11022: Regex in its jQuery.htmlPrefilter sometimes may introduce XSS
- CVE-2020-11023, CVE-2020-23064: passing HTML containing <option> elements from untrusted sources - even after sanitizing it - to one of jQuery's DOM manipulation methods (i.e. .html(), .append(), and others) may execute untrusted code.

Finding Evidence




Advisory
Request
Response
Path to issue
Pretty
Raw
Hex
Render
Hackvertor
\n
HTTP/2 200 OK
Date: Tue, 02 Apr 2024 14:07:30 GMT
Content-Type: application/javascript
Vary: Origin
Vary: Access-Control-Request-Method
Vary: Access-Control-Request-Headers
Last-Modified: Tue, 12 Mar 2024 15:05:43 GMT
Expires: Thu, 31 Dec 2037 23:55:55 GMT
Cache-Control: max-age=315360000
Cache-Control: public, must-revalidate, proxy-revalidate
Pragma: public
X-Xss-Protection: 1; mode=block
X-Content-Type-Options: nosniff
Strict-Transport-Security: max-age=31536000 ; includeSubDomains
X-Permitted-Cross-Domain-Policies: none
Cf-Cache-Status: HIT
Age: 1308316
Server: cloudflare
Cf-Ray: 86e162b4bd0c35cc-WAW
/*! jQuery v3.4.1 | (c) JS Foundation and other contributors | jquery.org/
license */