NETHEMBA. S R.O., GRÖSSLINGOVÁ 2503/62
811 09 BRATISLAVA

Comprehensive Penetration Test Unimus API

(Summary)

Customer: NetCore j.s.a.
Author: Nethemba
Consultant: Roman Fulop
Creation date: 2022-03-17
Last update: 2022-04-26

Table of Contents

1 Management Summary

From the 1st until 17thof March 2022 we have performed a comprehensive penetration test of the Unimus API according to the RFP of NetCore j.s.a.

The objective of this review was full coverage of the OWASP version 4.2 web application testing guide.

During the review we have identified two medium-risk issues, three low-risk issues and two informational-risk issues affecting the server and application:

The medium-risk findings were:

• Insecure Direct Object Reference (IDOR)	Medium
• No expiration on JWT tokens	Medium
• No Function Limiting	Low
• Invalid Credential UUID Accepted For Delete	Low
• Missing Lock Out	Low
• Business Logic Data Validation	Informational
• Stored Cross Site Scripting	Informational

In order to prevent exploitation of the above-mentioned application in the future we strongly advise that you follow these security recommendations:

- Implement a reasonable expiration of the API JWT auth tokens and further implement a refresh token to automatically renew the JWT token prior to expiration.
- Change the device number (and other internal identifiers like schedule id, etc.) from a sequential easily guessable number to a more randomized string (e.g., UUID).

Given the overall low risk of the identified findings, we consider the application secure enough for production operation.

All our findings and remediations are described in this report. We recommend that you follow all solutions prior, during, and after the integration and deployment.

2 Scope, Methodology, Tools, Risk Levels, Limiting Factors

2.1 Scope of the Review

The API application and server based at:

- https://10.31.3.79:8085

NetCore j.s.a. provided a core server and additional devices for testing. The testing was conducted on an internal VM network setup to mimic a typical client internal installation.

API authentication tokens were able to self generate.

API documentation was provided.

The following areas were explicitly excluded from the scope of this penetration test on customer's request:

- SSL/TLS configuration
- Missing HTTP response headers

2.2 Methodology and Tools

The comprehensive web application penetration test strictly follows the OWASP Web Application Testing Guide that focuses primarily on OWASP Top Ten:

- A01:2021-Broken Access Control
- A02:2021-Cryptographic Failures
- A03:2021-Injection
- A04:2021-Insecure Design
- A05:2021-Security Misconfiguration
- A06:2021-Vulnerable and Outdated Components
- A07:2021-Identification and Authentication Failures
- A08:2021-Software and Data Integrity Failures
- A09:2021-Security Logging and Monitoring Failures
- A10:2021-Server-Side Request Forgery

Although we have used many automated commercial (e.g. BurpSuite Pro, Nessus) and open-source tools (see https://owasp.org/www-community/Vulnerability_Scanning_Tools), many parts of the applications required manual testing and inspection.

The majority of our testing involves manual inspection that follows current best-practices in web application security.

2.3 Risk Levels

To describe a severity of revealed vulnerabilities we define the following risk levels:

- Critical – revealed vulnerabilities can be immediately exploited to take over the application or its users. (Critical vulnerabilities are SQL injection, persistent XSS, serious vulnerabilities in business logic, buffer overflows, privilege attacks, possibility of DOS attacks ...)
- High – revealed vulnerabilities can be critical in combination with other vulnerabilities (e. g. reflected/DOM-based XSS, Cross Site Request Forgery (CSRF), session fixation attacks ...)
- Medium – certain conditions have to be fulfilled to exploit these vulnerabilities or potential damage is limited (e. g. weak SSL ciphers, absence of cookie "Secure" or "HttpOnly" flag, ...)
- Low – minor security issues (e. g. enabled TRACE/TRACK HTTP methods)

2.4 Limiting Factors

No limiting factors were observed during testing.