CONFIDENTIAL

Vulnerability Assessment & Penetration Testing

Technical Report

Information Systems — crAPI (Lab)

Client	crAPI (Lab)
Target	http://127.0.0.1:8888
Asset type	API
Testing approach	Grey-box
Standards	OWASP API Security Top 10 2023
Overall risk	**Critical**
Report version	1.1
Report date	September 29, 2026
Prepared by	Muhammad Faqih

Table of Contents

List of Figures **3**

List of Tables **5**

1 Scope of Work **6**

1.1 Methodology 6

1.2 Standards and Frameworks 7

1.2.1 API (OWASP API Security Top 10 2023) 7

1.3 List of Assets 7

2 Summary of Vulnerabilities **8**

3 Findings on API **12**

3.1 crAPI Identity Service (http://127.0.0.1:8888/identity) 12

3.1.1 Broken Authentication (API2) – JWT Signature Not Verified — Authentication Bypass 12

3.1.2 Broken Authentication (API2) – Bypassable Brute-Force Protection on Password Reset OTP 16

3.1.3 Broken Object Level Authorization (API1) – Broken Object Level Authorization (IDOR) on Vehicle Location 18

3.1.4 Unrestricted Resource Consumption (API4) – Unrestricted Resource Consumption — No Login Rate Limiting 21

3.1.5 Improper Inventory Management (API9) – Improper Inventory Management — Deprecated OTP Endpoint (v2) Still Live 23

3.1.6 Broken Object Property Level Authorization (API3) – Mass Assignment on User Registration 25

3.1.7 Broken Object Level Authorization (API1) – Missing Authentication Check on Vehicle Location (Unauthenticated Access) 27

3.1.8 Broken Authentication (API2) – Password Reset OTP Per-Session Lockout 29

3.2 crAPI Workshop Service (http://127.0.0.1:8888/workshop) 31

3.2.1 Server Side Request Forgery (API7) – Server-Side Request Forgery (SSRF) via Contact Mechanic → Internal MailHog → Account Takeover 31

3.2.2 Broken Function Level Authorization (API5) – Unauthenticated Access to Order Details 35

3.2.3 Broken Function Level Authorization (API5) – Broken Function Level Authorization — Management Endpoint Exposes All Users 38

3.2.4 Broken Object Level Authorization (API1) – Broken Object Level Authorization (IDOR) on Mechanic Reports 40

3.2.5 Unsafe Consumption of APIs (API10) – Unsafe Consumption of APIs — Unvalidated Upstream Response in Contact Mechanic 42

3.2.6 Broken Object Level Authorization (API1) – Broken Object Level Authorization (IDOR) on Order Return 44

3.2.7 Broken Object Level Authorization (API1) – Missing Authentication Check on Mechanic Report (Unauthenticated Access) 46

3.2.8 Unrestricted Access to Sensitive Business Flows (API6) – Unrestricted Access to Sensitive Business Flow — Coupon Claim (Replay Protected) 48

3.3 crAPI Community Service (http://127.0.0.1:8888/community) 50

3.3.1 Security Misconfiguration (API8) – NoSQL Injection on Coupon Validation 50

3.3.2 Security Misconfiguration (API8) – Coupon Validation with Normal String Input 52

4 Summary of Recommendations **54**

5 Conclusion **56**

6 Attachments **57**

List of Figures

Figure 1 - Vulnerability Assessment and Penetration Testing Methodology 6
Figure 2 - JWT Signature Not Verified — Authentication Bypass (API2) - http://127.0.0.1:8888/identity/api/v2/user/dashboard (JWT Authentication) (Grey-box) - normal: valid token, own account 13
Figure 3 - JWT Signature Not Verified — Authentication Bypass (API2) - http://127.0.0.1:8888/identity/api/v2/user/dashboard (JWT Authentication) (Grey-box) - process: craft alg:none token 13
Figure 4 - JWT Signature Not Verified — Authentication Bypass (API2) - http://127.0.0.1:8888/identity/api/v2/user/dashboard (JWT Authentication) (Grey-box) - exploit: alg:none token accepted 14
Figure 5 - JWT Signature Not Verified — Authentication Bypass (API2) - http://127.0.0.1:8888/identity/api/v2/user/dashboard (JWT Authentication) (Grey-box) - process: tamper RS256, keep old signature 14
Figure 6 - JWT Signature Not Verified — Authentication Bypass (API2) - http://127.0.0.1:8888/identity/api/v2/user/dashboard (JWT Authentication) (Grey-box) - impact: tampered token accepted, CIA 15
Figure 7 - Bypassable Brute-Force Protection on Password Reset OTP (API2) - http://127.0.0.1:8888/identity/api/auth/v3/check-otp (Password Reset OTP) (Grey-box) - Normal flow: request password-reset OTP 17
Figure 8 - Bypassable Brute-Force Protection on Password Reset OTP (API2) - http://127.0.0.1:8888/identity/api/auth/v3/check-otp (Password Reset OTP) (Grey-box) - Normal flow: one wrong OTP attempt 17
Figure 9 - Bypassable Brute-Force Protection on Password Reset OTP (API2) - http://127.0.0.1:8888/identity/api/auth/v3/check-otp (Password Reset OTP) (Grey-box) - Impact: forget-password resets lockout, enabling ATO 17
Figure 10 - Broken Object Level Authorization (IDOR) on Vehicle Location (API1) - http://127.0.0.1:8888 (Grey-box) - crAPI Dashboard showing the vehicle location map (browser) 19
Figure 11 - Broken Object Level Authorization (IDOR) on Vehicle Location (API1) - http://127.0.0.1:8888/identity/api/v2/vehicle/{vehicleId}/location (Vehicle Location) (Grey-box) - Normal flow: own vehicle location 19
Figure 12 - Broken Object Level Authorization (IDOR) on Vehicle Location (API1) - http://127.0.0.1:8888/identity/api/v2/vehicle/{vehicleId}/location (Vehicle Location) (Grey-box) - Exploit: victim UUID returns victim location + email 20
Figure 13 - Unrestricted Resource Consumption — No Login Rate Limiting (API4) - http://127.0.0.1:8888/identity/api/auth/login (Login) (Grey-box) - single failed login 22
Figure 14 - Unrestricted Resource Consumption — No Login Rate Limiting (API4) - http://127.0.0.1:8888/identity/api/auth/login (Login) (Grey-box) - 20 attempts, no rate limiting 22
Figure 15 - Improper Inventory Management — Deprecated OTP Endpoint (v2) Still Live (API9) - http://127.0.0.1:8888/identity/api/auth/v2/check-otp (Password Reset OTP - legacy) (Grey-box) - current version v3 24
Figure 16 - Improper Inventory Management — Deprecated OTP Endpoint (v2) Still Live (API9) - http://127.0.0.1:8888/identity/api/auth/v2/check-otp (Password Reset OTP - legacy) (Grey-box) - old v2 still live 24
Figure 17 - Mass Assignment on User Registration (API3) - http://127.0.0.1:8888/identity/api/auth/signup (User Registration) (Grey-box) - Normal flow: standard registration 25
Figure 18 - Mass Assignment on User Registration (API3) - http://127.0.0.1:8888/identity/api/auth/signup (User Registration) (Grey-box) - Attempt: inject privileged fields role/isAdmin 26
Figure 19 - Mass Assignment on User Registration (API3) - http://127.0.0.1:8888/identity/api/auth/signup (User Registration) (Grey-box) - Result: role stays user, control effective 26
Figure 20 - Missing Authentication Check on Vehicle Location (Unauthenticated Access) (API1) - http://127.0.0.1:8888/identity/api/v2/vehicle/{vehicleId}/location (Vehicle Location) (Grey-box) - Normal flow: own vehicle location with token 27

Figure 21 - Missing Authentication Check on Vehicle Location (Unauthenticated Access) (API1) - http://127.0.0.1:8888/identity/api/v2/vehicle/{vehicleId}/location (Vehicle Location) (Grey-box) - Attempt blocked: no token returns 401 28
Figure 22 - Password Reset OTP Per-Session Lockout (API2) - http://127.0.0.1:8888/identity/api/auth/v3/check-otp (Password Reset OTP) (Grey-box) - Normal flow: one wrong OTP attempt 29
Figure 23 - Password Reset OTP Per-Session Lockout (API2) - http://127.0.0.1:8888/identity/api/auth/v3/check-otp (Password Reset OTP) (Grey-box) - Attempt blocked: lockout triggers at attempt 10 30
Figure 24 - Server-Side Request Forgery (SSRF) via Contact Mechanic → Internal MailHog → Account Takeover (API7) - http://127.0.0.1:8888 (Grey-box) - crAPI Dashboard with the 'Contact Mechanic' feature (browser) 32
Figure 25 - Server-Side Request Forgery (SSRF) via Contact Mechanic → Internal MailHog → Account Takeover (API7) - http://127.0.0.1:8888/workshop/api/merchant/contact_mechanic (Contact Mechanic) (Grey-box) - normal: server fetches mechanic API 32
Figure 26 - Server-Side Request Forgery (SSRF) via Contact Mechanic → Internal MailHog → Account Takeover (API7) - http://127.0.0.1:8888/workshop/api/merchant/contact_mechanic (Contact Mechanic) (Grey-box) - SSRF reads internal MailHog inbox (45 emails) 33
Figure 27 - Server-Side Request Forgery (SSRF) via Contact Mechanic → Internal MailHog → Account Takeover (API7) - http://127.0.0.1:8888/workshop/api/merchant/contact_mechanic (Contact Mechanic) (Grey-box) - harvest victim OTP via SSRF 33
Figure 28 - Server-Side Request Forgery (SSRF) via Contact Mechanic → Internal MailHog → Account Takeover (API7) - http://127.0.0.1:8888/workshop/api/merchant/contact_mechanic (Contact Mechanic) (Grey-box) - reset victim password with harvested OTP 34
Figure 29 - Server-Side Request Forgery (SSRF) via Contact Mechanic → Internal MailHog → Account Takeover (API7) - http://127.0.0.1:8888/workshop/api/merchant/contact_mechanic (Contact Mechanic) (Grey-box) - login as victim — Account Takeover 34
Figure 30 - Unauthenticated Access to Order Details (API5) - http://127.0.0.1:8888 (Grey-box) - crAPI Past Orders page with 'Order Details' (browser) 36
Figure 31 - Unauthenticated Access to Order Details (API5) - http://127.0.0.1:8888/workshop/api/shop/orders/{order_id} (Order Details) (Grey-box) - Normal flow: authenticated order request 36
Figure 32 - Unauthenticated Access to Order Details (API5) - http://127.0.0.1:8888/workshop/api/shop/orders/{order_id} (Order Details) (Grey-box) - Exploit: no auth, PII disclosed 37
Figure 33 - Broken Function Level Authorization — Management Endpoint Exposes All Users (API5) - http://127.0.0.1:8888/workshop/api/management/users/all (User Management) (Grey-box) - normal: own account only 39
Figure 34 - Broken Function Level Authorization — Management Endpoint Exposes All Users (API5) - http://127.0.0.1:8888/workshop/api/management/users/all (User Management) (Grey-box) - exploit: dump all users' PII 39
Figure 35 - Broken Object Level Authorization (IDOR) on Mechanic Reports (API1) - http://127.0.0.1:8888/workshop/api/mechanic/mechanic_report?report_id={id} (Mechanic Report) (Grey-box) - Normal flow: reference report_id=1 41
Figure 36 - Broken Object Level Authorization (IDOR) on Mechanic Reports (API1) - http://127.0.0.1:8888/workshop/api/mechanic/mechanic_report?report_id={id} (Mechanic Report) (Grey-box) - Exploit: report_id=2 leaks another owner's data 41
Figure 37 - Unsafe Consumption of APIs — Unvalidated Upstream Response in Contact Mechanic (API10) - http://127.0.0.1:8888/workshop/api/merchant/contact_mechanic (Contact Mechanic) (Grey-box) - consuming mechanic API 43
Figure 38 - Unsafe Consumption of APIs — Unvalidated Upstream Response in Contact Mechanic (API10) - http://127.0.0.1:8888/workshop/api/merchant/contact_mechanic (Contact Mechanic) (Grey-box) - raw upstream reflected 43
Figure 39 - Broken Object Level Authorization (IDOR) on Order Return (API1) - http://127.0.0.1:8888 (Grey-box) - crAPI Past Orders page with the 'Return' action (browser) 44

Figure 40 - Broken Object Level Authorization (IDOR) on Order Return (API1) - http://127.0.0.1:8888/workshop/api/shop/orders/return_order?order_id={id} (Return Order) (Grey-box) - Attempt blocked: return of another user's order returns 403 45
Figure 41 - Missing Authentication Check on Mechanic Report (Unauthenticated Access) (API1) - http://127.0.0.1:8888/workshop/api/mechanic/mechanic_report?report_id={id} (Mechanic Report) (Grey-box) - Normal flow: report with valid token 46
Figure 42 - Missing Authentication Check on Mechanic Report (Unauthenticated Access) (API1) - http://127.0.0.1:8888/workshop/api/mechanic/mechanic_report?report_id={id} (Mechanic Report) (Grey-box) - Attempt blocked: no token returns 401 47
Figure 43 - Unrestricted Access to Sensitive Business Flow — Coupon Claim (Replay Protected) (API6) - http://127.0.0.1:8888/workshop/api/shop/apply_coupon (Coupon Claim) (Grey-box) - first coupon claim succeeds 48
Figure 44 - Unrestricted Access to Sensitive Business Flow — Coupon Claim (Replay Protected) (API6) - http://127.0.0.1:8888/workshop/api/shop/apply_coupon (Coupon Claim) (Grey-box) - replay rejected 49
Figure 45 - NoSQL Injection on Coupon Validation (API8) - http://127.0.0.1:8888/community/api/v2/coupon/validate-coupon (Coupon Validation) (Grey-box) - Normal flow: valid string coupon code 51
Figure 46 - NoSQL Injection on Coupon Validation (API8) - http://127.0.0.1:8888/community/api/v2/coupon/validate-coupon (Coupon Validation) (Grey-box) - Exploit: $ne operator leaks a valid coupon 51
Figure 47 - NoSQL Injection on Coupon Validation (API8) - http://127.0.0.1:8888/community/api/v2/coupon/validate-coupon (Coupon Validation) (Grey-box) - Impact: $gt also leaks coupon, injection confirmed 51
Figure 48 - Coupon Validation with Normal String Input (API8) - http://127.0.0.1:8888/community/api/v2/coupon/validate-coupon (Coupon Validation) (Grey-box) - Normal flow: valid coupon code accepted . 52
Figure 49 - Coupon Validation with Normal String Input (API8) - http://127.0.0.1:8888/community/api/v2/coupon/validate-coupon (Coupon Validation) (Grey-box) - Control: unknown string code rejected 53

List of Tables

Table 1 - Penetration Testing Approaches 6
Table 2 - OWASP API Security Top 10 2023 Categories 7
Table 3 - List of Assets 7
Table 4 - Vulnerability Summary – crAPI Identity Service 8
Table 5 - Vulnerability Summary – crAPI Workshop Service 9
Table 6 - Vulnerability Summary – crAPI Community Service 10

1 Scope of Work

1.1 Methodology

The methodology used in this vulnerability assessment and penetration test is described below.

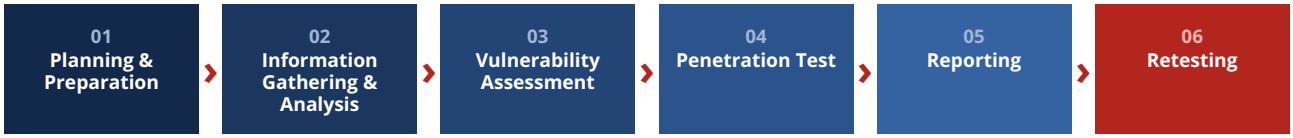


Figure 1 - Vulnerability Assessment and Penetration Testing Methodology

1. **Planning and understanding the target system and supporting applications.** Understanding the target system, supporting applications, public and internal IP addresses, operating systems (OS), databases, and application topology within the relevant work unit, through interviews or the documentation provided.
2. **Conducting the vulnerability assessment.** Assessing vulnerabilities and identifying security weaknesses within the scope of the system and its supporting network infrastructure, including web applications, operating systems, databases, and networks.
3. **Performing penetration testing based on the assessment results.** Exploitation aims to prove that the security gaps or vulnerabilities found exist, and to assess their level of risk and impact on the company's applications, information, and business reputation.
4. **Reporting and discussing results with Team crAPI (Lab).** Preparing the report and discussing the penetration test results and recommendations for improvement, and reaching a mutual agreement on reducing the existing security gaps. Recommendations are based on the standards and frameworks used in the test.
5. **Retesting the implemented improvements.** Retesting the improvements implemented to ensure that security gaps and risks have been effectively reduced.

The testing approaches used in penetration testing are black-box, grey-box and white-box:

Table 1 - Penetration Testing Approaches

Approach	Description	This engagement
Black-box	Conducted without any prior information or knowledge about the target. Testing uses only a list of IP addresses or domains, without detailed knowledge of the system profile or application topology.	—
Grey-box	Conducted with some information and knowledge about the target, including user access to web applications or servers but without full access rights. Testing is still limited by firewall filtering or a WAF (Web Application Firewall).	✔ Applied
White-box	Conducted with comprehensive information about the target, such as the application topology, test user IDs, internal network access, and the IP addresses of the target and users, without any firewall or WAF filtering.	—

1.2 Standards and Frameworks

Findings in this report are classified against the following standards and frameworks: **OWASP API Security Top 10 2023**.

1.2.1 API (OWASP API Security Top 10 2023)

The OWASP API Security Top 10 (2023) lists the most critical security risks specific to APIs. Reference: https://owasp.org/API-Security/editions/2023/en/0x11-t10/

Table 2 - OWASP API Security Top 10 2023 Categories

ID	Category
API1:2023	Broken Object Level Authorization
API2:2023	Broken Authentication
API3:2023	Broken Object Property Level Authorization
API4:2023	Unrestricted Resource Consumption
API5:2023	Broken Function Level Authorization
API6:2023	Unrestricted Access to Sensitive Business Flows
API7:2023	Server Side Request Forgery
API8:2023	Security Misconfiguration
API9:2023	Improper Inventory Management
API10:2023	Unsafe Consumption of APIs

1.3 List of Assets

The scope of security testing and penetration testing for the crAPI (Lab) web application is as follows:

Table 3 - List of Assets

No	Name	URL	Type	Status
1	**crAPI Identity Service**	`http://127.0.0.1:8888/identity`	API	FAIL
2	**crAPI Workshop Service**	`http://127.0.0.1:8888/workshop`	API	FAIL
3	**crAPI Community Service**	`http://127.0.0.1:8888/community`	API	FAIL

Status legend: **PASS** — no vulnerability found **FAIL** — vulnerability found

2 Summary of Vulnerabilities

Testing of the crAPI (Lab) web application identified **11 vulnerabilities**: **2 Critical**, **3 High** and **6 Medium**; in addition, **7 tested scenarios** did not result in a vulnerability (PASS).

2	3	6	0	0	7
CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL	PASS

crAPI Identity Service

http://127.0.0.1:8888/identity · API · 5 vulnerabilities, 3 PASS

Table 4 - Vulnerability Summary – crAPI Identity Service

No	Vulnerability	Severity	CVSS	Exploited	Recommendation
1	**Broken Authentication (API2) – JWT Signature Not Verified — Authentication Bypass** API2:2023 Broken Authentication	Critical	9.8	Yes	Verify the JWT signature with the RS256 public key on every request; reject alg:none and allow-list algorithms.
2	**Broken Authentication (API2) – Bypassable Brute-Force Protection on Password Reset OTP** API2:2023 Broken Authentication	High	7.4	Yes	Enforce persistent per-account lockout and rate-limit/CAPTCHA the forget-password endpoint; use longer OTPs.
3	**Broken Object Level Authorization (API1) – Broken Object Level Authorization (IDOR) on Vehicle Location** API1:2023 Broken Object Level Authorization	Medium	6.5	Yes	Verify vehicle ownership against the authenticated user before returning location.
4	**Unrestricted Resource Consumption (API4) – Unrestricted Resource Consumption — No Login Rate Limiting** API4:2023 Unrestricted Resource Consumption	Medium	6.5	Yes	Enforce rate limiting and account lockout on login; add CAPTCHA/anti-automation.
5	**Improper Inventory Management (API9) – Improper Inventory Management — Deprecated OTP Endpoint (v2) Still Live** API9:2023 Improper Inventory Management	Medium	5.3	Yes	Retire deprecated API versions (v2) and maintain an inventory of live endpoints.

No	Vulnerability	Severity	CVSS	Exploited	Recommendation
6	**Broken Object Property Level Authorization (API3) – Mass Assignment on User Registration** API3:2023 Broken Object Property Level Authorization	PASS	—	—	Keep the current allow-list of client-settable fields at registration.
7	**Broken Object Level Authorization (API1) – Missing Authentication Check on Vehicle Location (Unauthenticated Access)** API1:2023 Broken Object Level Authorization	PASS	—	—	Keep authentication mandatory; also add object-ownership checks (see the related FAIL finding).
8	**Broken Authentication (API2) – Password Reset OTP Per-Session Lockout** API2:2023 Broken Authentication	PASS	—	—	Make the lockout persistent per account so it cannot be reset via a new OTP request (see the related FAIL finding).

crAPI Workshop Service

http://127.0.0.1:8888/workshop · API · 5 vulnerabilities, 3 PASS

Table 5 - Vulnerability Summary – crAPI Workshop Service

No	Vulnerability	Severity	CVSS	Exploited	Recommendation
9	**Server Side Request Forgery (API7) – Server-Side Request Forgery (SSRF) via Contact Mechanic → Internal MailHog → Account Takeover** API7:2023 Server Side Request Forgery	Critical	9.6	Yes	Allow-list mechanic endpoints; block internal hosts (mailhog/identity); segment internal services; don't reflect upstream responses.
10	**Broken Function Level Authorization (API5) – Unauthenticated Access to Order Details** API5:2023 Broken Function Level Authorization API1:2023 Broken Object Level Authorization	High	7.5	Yes	Require authentication and verify order ownership; use non-sequential identifiers (UUID).
11	**Broken Function Level Authorization (API5) – Broken Function Level Authorization — Management Endpoint Exposes All Users** API5:2023 Broken Function Level Authorization API3:2023 Broken Object Property Level Authorization	Medium	6.5	Yes	Enforce server-side role-based authorization on all management endpoints.

No	Vulnerability	Severity	CVSS	Exploited	Recommendation
12	**Broken Object Level Authorization (API1) – Broken Object Level Authorization (IDOR) on Mechanic Reports** API1:2023 Broken Object Level Authorization API3:2023 Broken Object Property Level Authorization	Medium	6.5	Yes	Verify the requested report belongs to the caller; use UUIDs and minimize returned fields.
13	**Unsafe Consumption of APIs (API10) – Unsafe Consumption of APIs — Unvalidated Upstream Response in Contact Mechanic** API10:2023 Unsafe Consumption of APIs	Medium	5.4	Yes	Validate and sanitize upstream API responses; restrict upstream destinations to a trusted allow-list.
14	**Broken Object Level Authorization (API1) – Broken Object Level Authorization (IDOR) on Order Return** API1:2023 Broken Object Level Authorization	PASS	—	—	Maintain the ownership check on the return action.
15	**Broken Object Level Authorization (API1) – Missing Authentication Check on Mechanic Report (Unauthenticated Access)** API1:2023 Broken Object Level Authorization	PASS	—	—	Keep authentication mandatory; also add object-ownership checks (see the related FAIL finding).
16	**Unrestricted Access to Sensitive Business Flows (API6) – Unrestricted Access to Sensitive Business Flow — Coupon Claim (Replay Protected)** API6:2023 Unrestricted Access to Sensitive Business Flows	PASS	—	—	Claiming the same coupon twice was rejected ('already claimed'); the per-user limit is enforced.

crAPI Community Service

http://127.0.0.1:8888/community · API · 1 vulnerability, 1 PASS

Table 6 - Vulnerability Summary – crAPI Community Service

No	Vulnerability	Severity	CVSS	Exploited	Recommendation
17	**Security Misconfiguration (API8) – NoSQL Injection on Coupon Validation** API8:2023 Security Misconfiguration	High	7.1	Yes	Coerce coupon_code to a string and reject objects/operators; validate the input schema.

No	Vulnerability	Severity	CVSS	Exploited	Recommendation
18	**Security Misconfiguration (API8) – Coupon Validation with Normal String Input** API8:2023 Security Misconfiguration	PASS	—	—	Enforce string type and reject operators/ objects (see the related FAIL finding).

References used for every finding: OWASP API Security Top 10 2023, Bugcrowd VRT, CWE 4.0, CVSS v3.1.

3 Findings on API

3.1 crAPI Identity Service (http://127.0.0.1:8888/identity)

3.1.1 Broken Authentication (API2) – JWT Signature Not Verified — Authentication Bypass

SEVERITY	CVSS V3.1 BASE SCORE	EXPLOITED	WEAKNESS
Critical	9.8 / 10	Yes	CWE-347

Standard mapping	**API2:2023** Broken Authentication (OWASP API Security Top 10 2023)
Impact rating	Critical
Bugcrowd VRT	P1 - Broken Authentication and Session Management (VRT 2.1)
CVSS vector	`CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H`
Affected feature / URL	http://127.0.0.1:8888/identity/api/v2/user/dashboard (JWT Authentication)
Testing approach	Grey-box

DESCRIPTION

A broken authentication vulnerability was found in the crAPI identity service where JSON Web Tokens (JWT) are accepted without verifying their signature. Tokens crafted with the header alg:none (no signature) are accepted, and a legitimate RS256 token whose payload is modified without re-signing is also accepted. Because the signature is never validated, an attacker can forge a token for any user and elevate the role claim to admin.

IMPACT

Attackers can use this vulnerability to fully take over any account without credentials. During testing, the team forged a token containing the victim's identity and role=admin and accessed the victim's account data, breaking both Confidentiality and Integrity of the authentication model.

RECOMMENDATION

1. Verify the JWT signature with the server's RS256 public key on every request.
2. Explicitly reject the alg:none algorithm and enforce algorithm allow-listing (RS256 only).
3. Use a mature JWT library with mandatory verification and validate exp, iss, and aud claims.

REFERENCES

- VRT 2.1 - Broken Authentication and Session Management
- https://owasp.org/API-Security/editions/2023/en/0xa2-broken-authentication/
- https://cwe.mitre.org/data/definitions/347.html
- https://bugcrowd.com/vulnerability-rating-taxonomy

EVIDENCE

Affected URL: http://127.0.0.1:8888/identity/api/v2/user/dashboard (JWT Authentication)

a. The team logged in and used its own valid RS256 token (shown in full) to access its own data, which returned only the attacker's own resources.

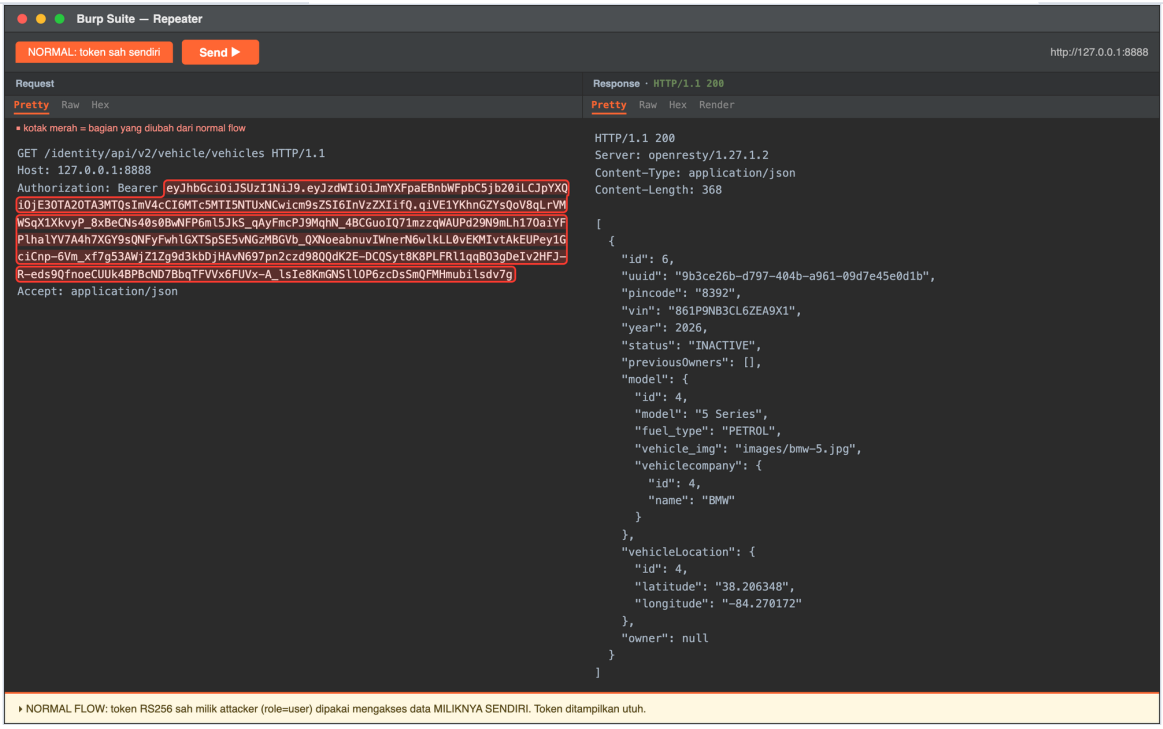


Figure 2 - JWT Signature Not Verified — Authentication Bypass (API2) - http://127.0.0.1:8888/identity/api/v2/user/dashboard (JWT Authentication) (Grey-box) - normal: valid token, own account

b. The team decoded the JWT (base64url header and payload), changed the algorithm to "none" with an empty signature, set role=admin and sub=admin@example.com (impersonating the administrator), and re-encoded the complete forged token (shown end-to-end, untruncated).

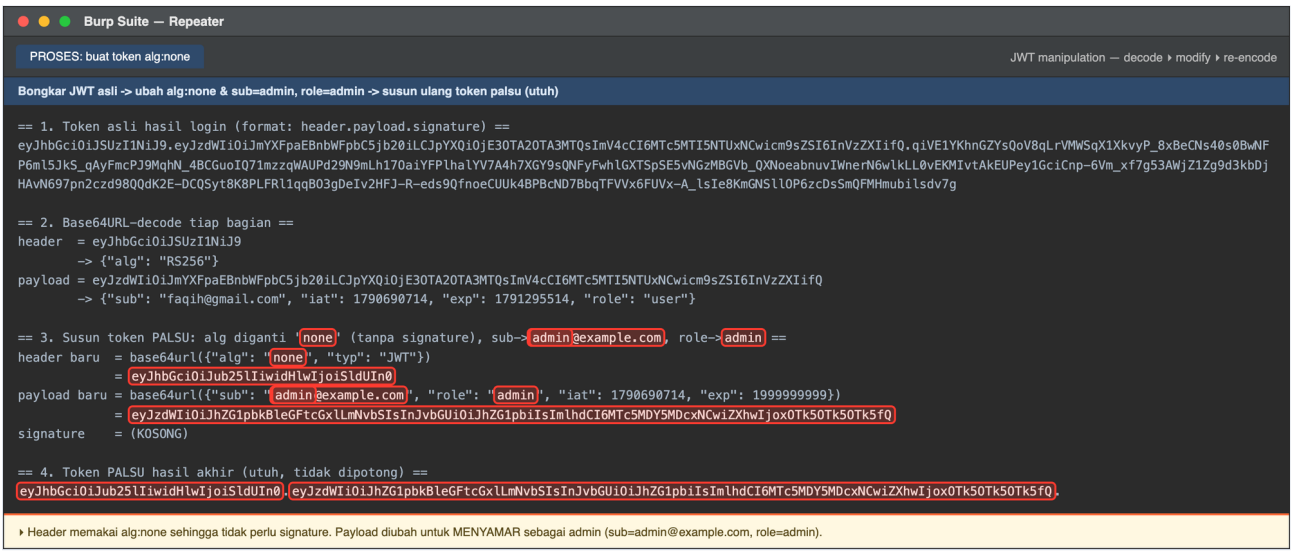


Figure 3 - JWT Signature Not Verified — Authentication Bypass (API2) - http://127.0.0.1:8888/identity/api/v2/user/dashboard (JWT Authentication) (Grey-box) - process: craft alg:none token

c. The team sent the forged alg:none token; the server accepted it and returned the ADMIN account (id=5, ROLE_ADMIN).

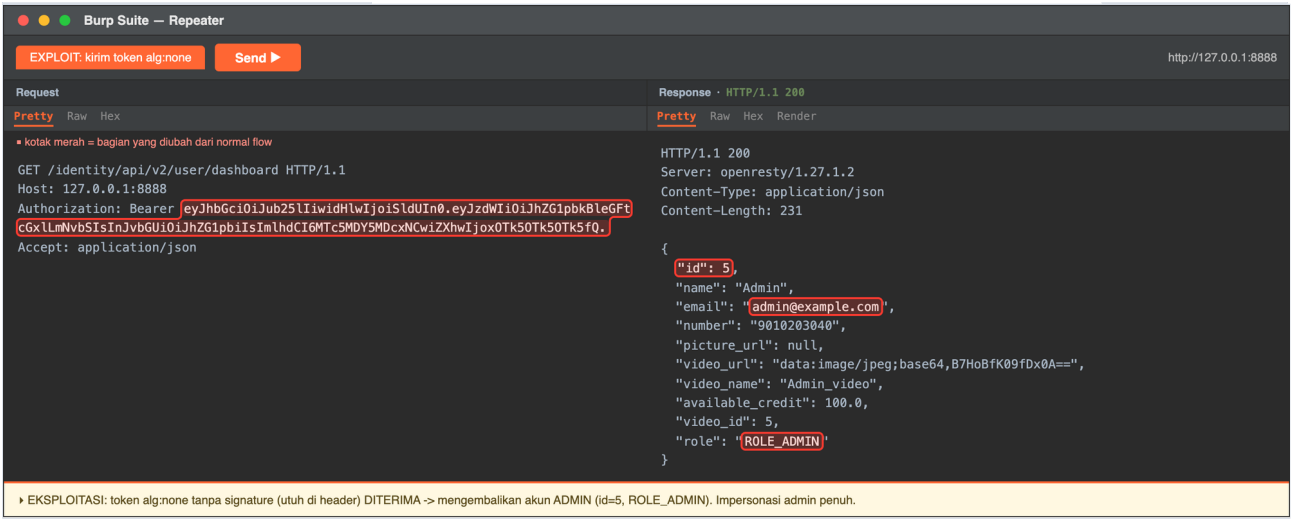


Figure 4 - JWT Signature Not Verified — Authentication Bypass (API2) - http://127.0.0.1:8888/identity/api/v2/user/dashboard (JWT Authentication) (Grey-box) - exploit: alg:none token accepted

d. The team took the valid RS256 token, edited its payload to role=admin and sub=admin@example.com, and reattached the ORIGINAL signature without re-signing, producing the complete tampered token (shown in full).

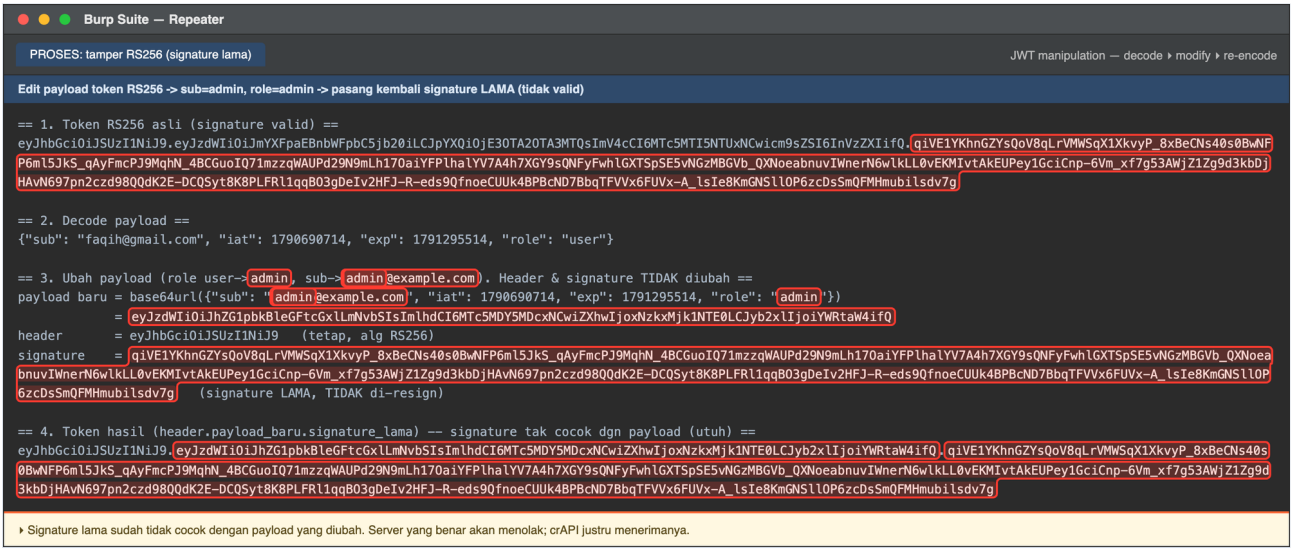


Figure 5 - JWT Signature Not Verified — Authentication Bypass (API2) - http://127.0.0.1:8888/identity/api/v2/user/dashboard (JWT Authentication) (Grey-box) - process: tamper RS256, keep old signature

e. The team sent the tampered RS256 token; the server still returned HTTP 200 with the victim's data, proving the signature is not verified (Confidentiality + Integrity).

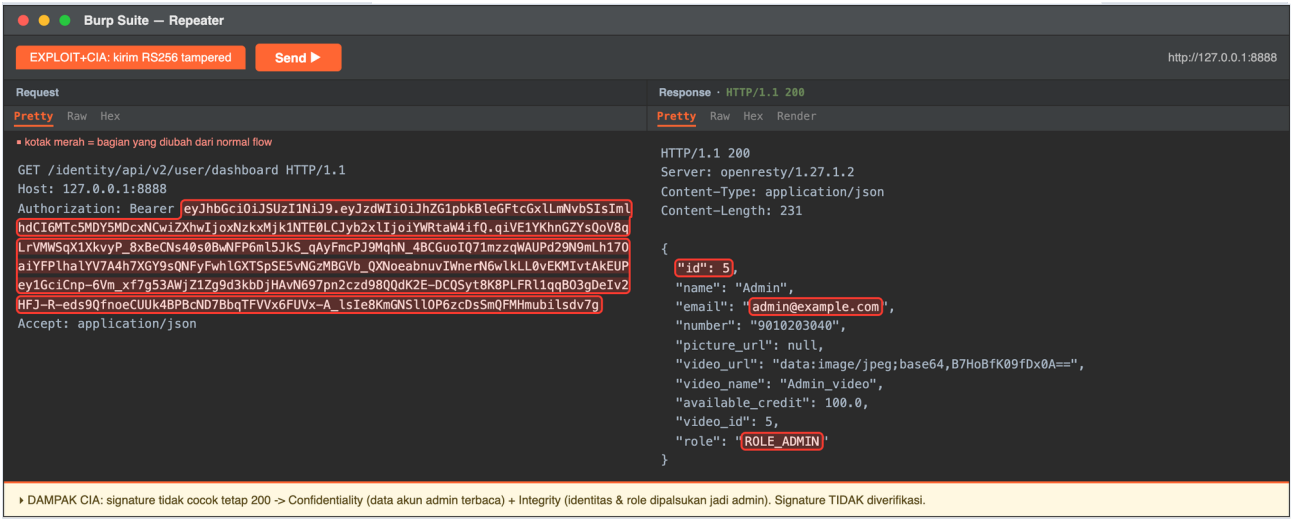


Figure 6 - JWT Signature Not Verified — Authentication Bypass (API2) - http://127.0.0.1:8888/identity/api/v2/user/dashboard (JWT Authentication) (Grey-box) - impact: tampered token accepted, CIA

3.1.2 Broken Authentication (API2) – Bypassable Brute-Force Protection on Password Reset OTP

SEVERITY	CVSS V3.1 BASE SCORE	EXPLOITED	WEAKNESS
High	7.4 / 10	Yes	CWE-307

Standard mapping	**API2:2023** Broken Authentication (OWASP API Security Top 10 2023)
Impact rating	**Medium**
Bugcrowd VRT	P2 - Broken Authentication and Session Management (VRT 2.4)
CVSS vector	`CVSS:3.1/AV:N/AC:H/PR:N/UI:N/S:U/C:H/I:H/A:N`
Affected feature / URL	http://127.0.0.1:8888/identity/api/auth/v3/check-otp (Password Reset OTP)
Testing approach	Grey-box

DESCRIPTION

A weak brute-force protection vulnerability was found in the password reset flow. The OTP verification endpoint locks after 9 wrong attempts per OTP session, but this counter is reset every time forget-password is called, and forget-password is itself neither rate-limited nor CAPTCHA-protected. The protection can therefore be bypassed with a cyclic pattern against the 4-digit OTP.

IMPACT

Attackers can use this vulnerability to take over any account whose email is known. Because the OTP is only 4 digits (10,000 combinations) and each cycle grants 9 unimpeded guesses against a fresh OTP, an automated attacker can reset the victim's password (Confidentiality + Integrity).

RECOMMENDATION

1. Enforce a persistent per-account lockout that is not reset by requesting a new OTP.
2. Rate-limit and add CAPTCHA/anti-automation to the forget-password endpoint.
3. Use OTPs of at least 6 digits with short expiry and cumulative failure invalidation.

REFERENCES

- VRT 2.4 - Broken Authentication and Session Management
- https://owasp.org/API-Security/editions/2023/en/0xa2-broken-authentication/
- https://cwe.mitre.org/data/definitions/307.html
- https://bugcrowd.com/vulnerability-rating-taxonomy

EVIDENCE

Affected URL: http://127.0.0.1:8888/identity/api/auth/v3/check-otp (Password Reset OTP)

a. The team triggered the password reset by calling forget-password for the victim's email, which returned HTTP 200 (normal flow).

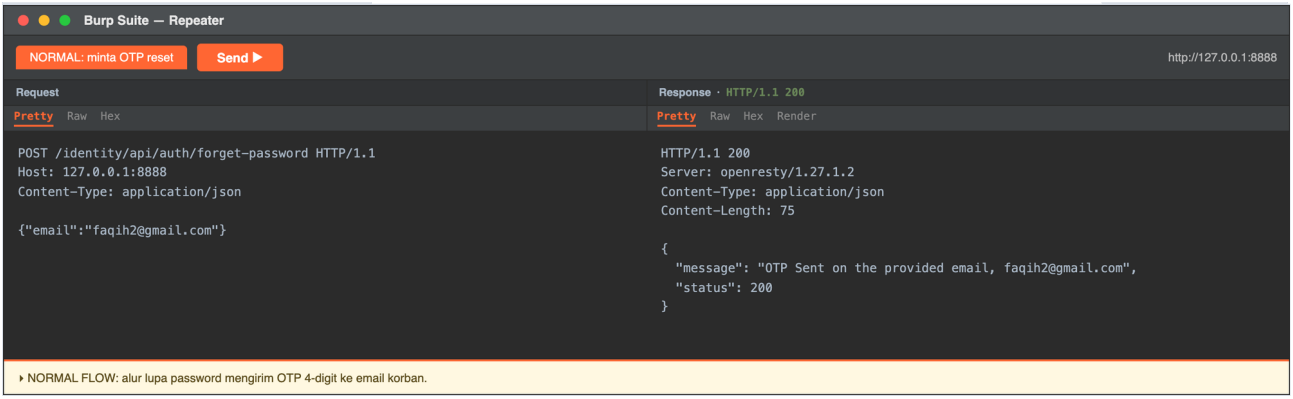


Figure 7 - Bypassable Brute-Force Protection on Password Reset OTP (API2) - http://127.0.0.1:8888/identity/api/auth/v3/check-otp (Password Reset OTP) (Grey-box) - Normal flow: request password-reset OTP

b. The team submitted one wrong OTP to check-otp, which returned Invalid OTP (normal single attempt).

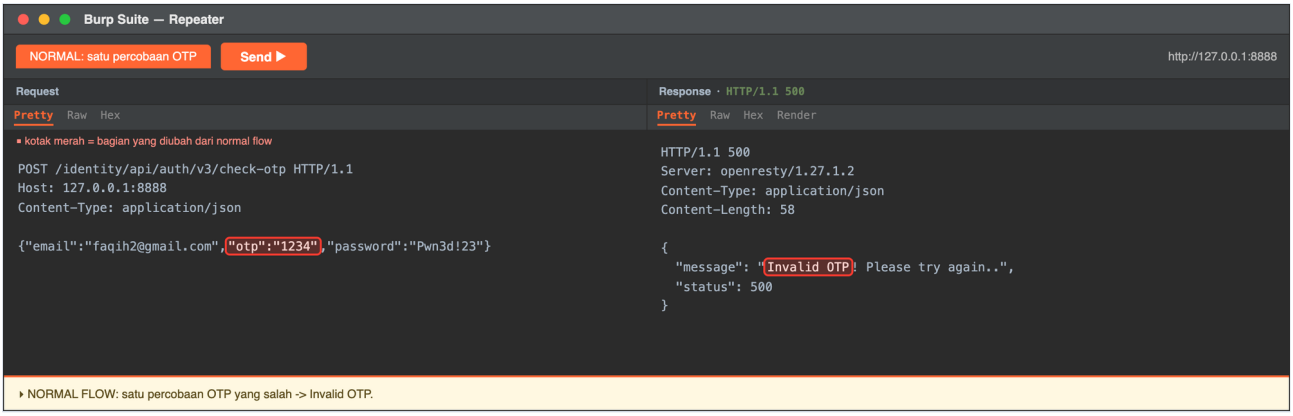


Figure 8 - Bypassable Brute-Force Protection on Password Reset OTP (API2) - http://127.0.0.1:8888/identity/api/auth/v3/check-otp (Password Reset OTP) (Grey-box) - Normal flow: one wrong OTP attempt

c. After the lockout would trigger, the team called forget-password again (HTTP 200, no throttle), resetting the attempt counter and issuing a new OTP. Repeating the cycle [reset -> 9 guesses] brute-forces the 4-digit OTP and resets the victim's password (Confidentiality + Integrity).

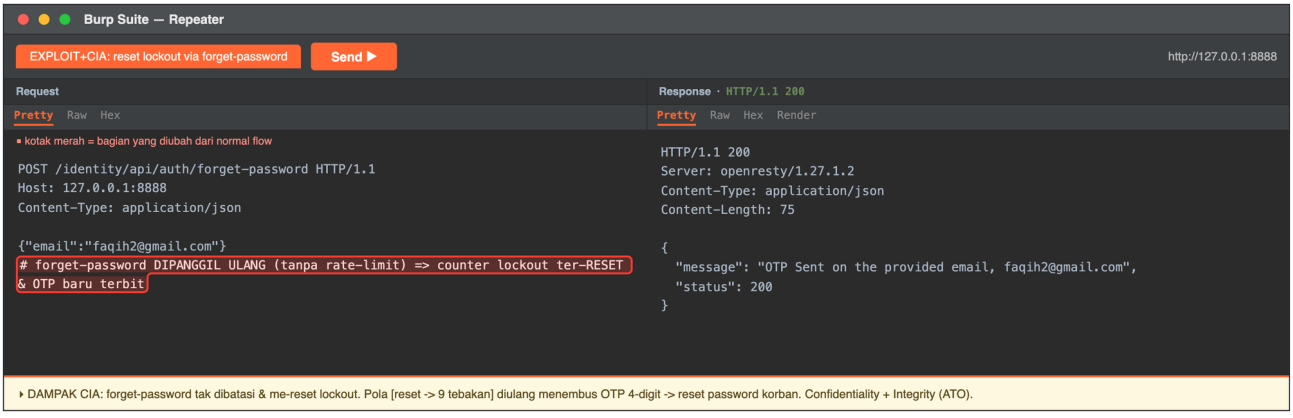


Figure 9 - Bypassable Brute-Force Protection on Password Reset OTP (API2) - http://127.0.0.1:8888/identity/api/auth/v3/check-otp (Password Reset OTP) (Grey-box) - Impact: forget-password resets lockout, enabling ATO

3.1.3 Broken Object Level Authorization (API1) – Broken Object Level Authorization (IDOR) on Vehicle Location

SEVERITY	CVSS V3.1 BASE SCORE	EXPLOITED	WEAKNESS
Medium	6.5 / 10	Yes	CWE-639

Standard mapping	**API1:2023** Broken Object Level Authorization (OWASP API Security Top 10 2023)
Impact rating	High
Bugcrowd VRT	P2 - Insecure Direct Object References (IDOR) (VRT 3.3)
CVSS vector	`CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:N`
Affected feature / URL	http://127.0.0.1:8888/identity/api/v2/vehicle/{vehicleId}/location (Vehicle Location)
Testing approach	Grey-box

DESCRIPTION

A Broken Object Level Authorization (BOLA/IDOR) vulnerability was found in the vehicle location endpoint. The server authenticates the caller but does not verify that the requested vehicle belongs to that user, so supplying another user's vehicle UUID returns that vehicle's GPS location.

IMPACT

Attackers can use this vulnerability to physically track other users in real time and to obtain their name and email. During testing, the team read the victim's live GPS coordinates and email using the attacker's own token (Confidentiality).

RECOMMENDATION

1. Verify that the requested vehicle belongs to the user in the token before returning its location.
2. Implement centralized object-level authorization checks.
3. Avoid exposing owner PII in the location response.

REFERENCES

- VRT 3.3 - Insecure Direct Object References
- https://owasp.org/API-Security/editions/2023/en/0xa1-broken-object-level-authorization/
- https://cwe.mitre.org/data/definitions/639.html
- https://bugcrowd.com/vulnerability-rating-taxonomy

EVIDENCE

Affected URL: http://127.0.0.1:8888/identity/api/v2/vehicle/{vehicleId}/location (Vehicle Location)

a. The team logged into the crAPI web application. The Dashboard shows the vehicle's live location on a map — the feature backed by the vulnerable API tested below.

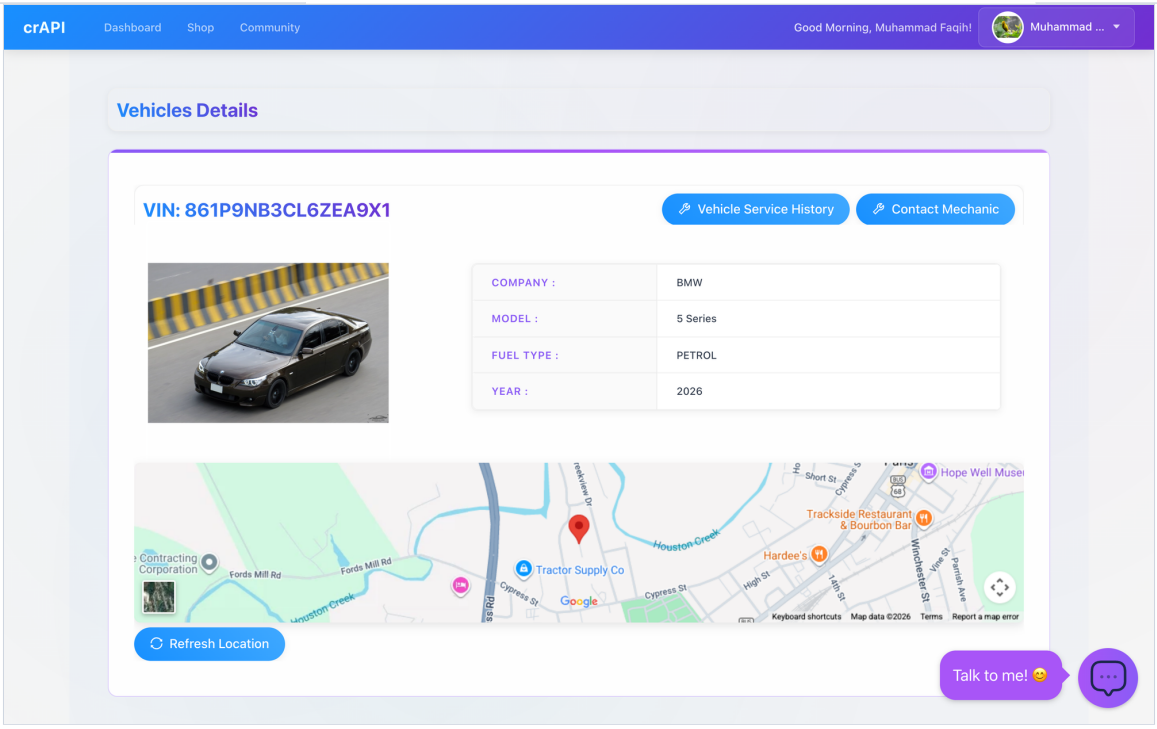


Figure 10 - Broken Object Level Authorization (IDOR) on Vehicle Location (API1) - http://127.0.0.1:8888 (Grey-box) - crAPI Dashboard showing the vehicle location map (browser)

b. The team requested the location of its OWN vehicle (uuid 9b3ce26b-d797-404b-a961-09d7e45e0d1b) using the attacker's token, which returned HTTP 200 (normal behavior).

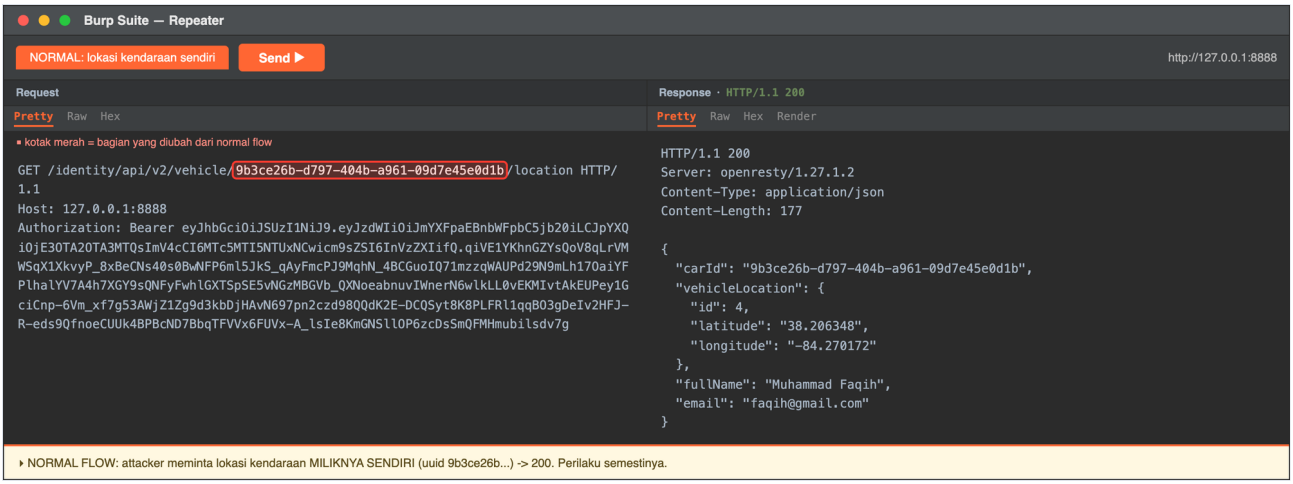


Figure 11 - Broken Object Level Authorization (IDOR) on Vehicle Location (API1) - http://127.0.0.1:8888/identity/api/v2/vehicle/{vehicleId}/location (Vehicle Location) (Grey-box) - Normal flow: own vehicle location

c. The team changed only the vehicle UUID in the path to the VICTIM's vehicle (8ba9066d-8484-4ef7-8a7b-c0fa3cb2e687) while keeping the same attacker token. The server returned HTTP 200 with the victim's GPS coordinates and email (Confidentiality).

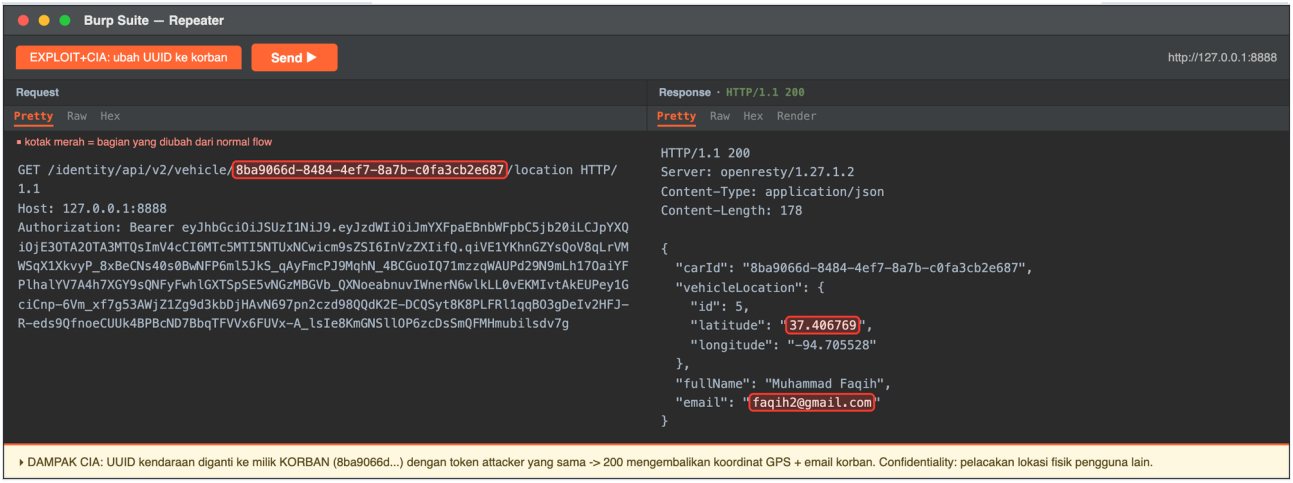


Figure 12 - Broken Object Level Authorization (IDOR) on Vehicle Location (API1) - http://127.0.0.1:8888/identity/api/v2/vehicle/{vehicleId}/location (Vehicle Location) (Grey-box) - Exploit: victim UUID returns victim location + email

3.1.4 Unrestricted Resource Consumption (API4) – Unrestricted Resource Consumption — No Login Rate Limiting

SEVERITY	CVSS V3.1 BASE SCORE	EXPLOITED	WEAKNESS
Medium	6.5 / 10	Yes	CWE-307

Standard mapping	**API4:2023** Unrestricted Resource Consumption (OWASP API Security Top 10 2023)
Impact rating	**Medium**
Bugcrowd VRT	P4 - Broken Authentication and Session Management - Lack of Rate Limiting (VRT 2.6)
CVSS vector	`CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:L`
Affected feature / URL	http://127.0.0.1:8888/identity/api/auth/login (Login)
Testing approach	Grey-box

DESCRIPTION

The login endpoint enforces no rate limiting or account lockout. Repeated failed logins are not throttled, enabling brute-force/credential stuffing and unbounded resource consumption.

IMPACT

Attackers can use this to brute-force credentials and to exhaust server resources with unlimited authentication requests (Availability, and Confidentiality via brute-force).

RECOMMENDATION

1. Enforce per-account/per-IP rate limiting and progressive account lockout.
2. Add CAPTCHA/anti-automation and anomaly detection.
3. Limit concurrent requests per client.

REFERENCES

- VRT 2.6 - Broken Authentication and Session Management - Lack of Rate Limiting
- https://owasp.org/API-Security/editions/2023/en/0xa4-unrestricted-resource-consumption/
- https://cwe.mitre.org/data/definitions/307.html
- https://bugcrowd.com/vulnerability-rating-taxonomy

EVIDENCE

Affected URL: http://127.0.0.1:8888/identity/api/auth/login (Login)

a. The team submitted a single login attempt with a wrong password, which returned HTTP 401 (normal failed login).

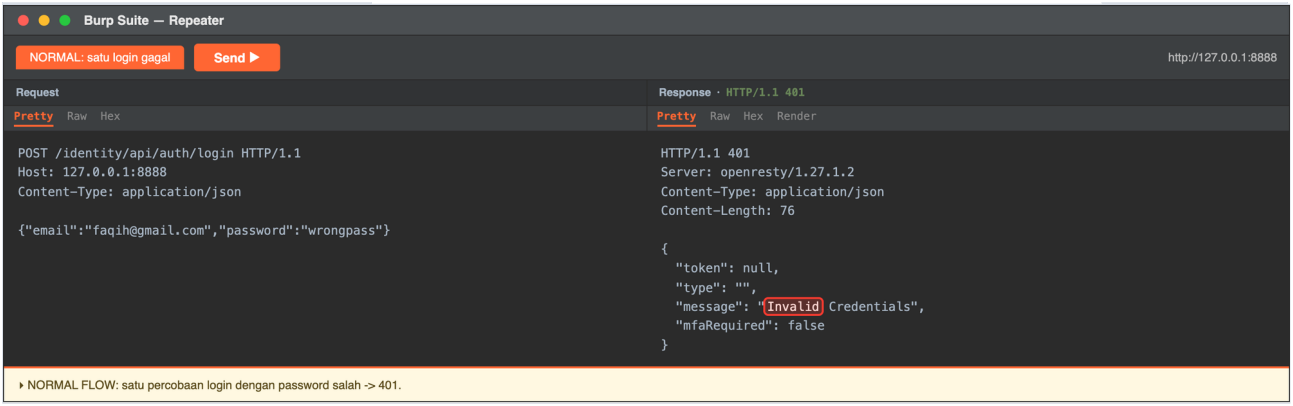


Figure 13 - Unrestricted Resource Consumption — No Login Rate Limiting (API4) - http://127.0.0.1:8888/identity/api/auth/login (Login) (Grey-box) - single failed login

b. The team then sent 20 consecutive login attempts; all returned HTTP 401 with no HTTP 429 or lockout at any point, proving there is no rate limiting.

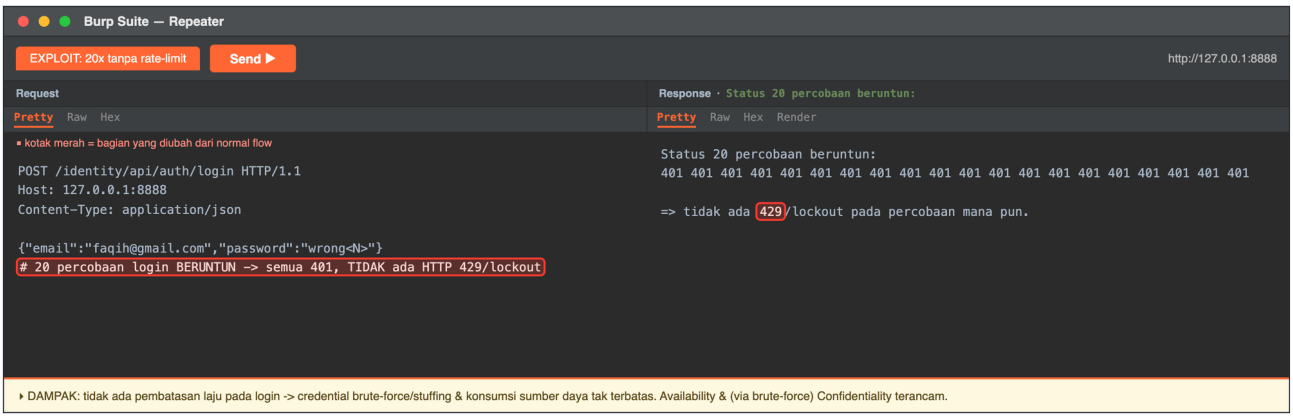


Figure 14 - Unrestricted Resource Consumption — No Login Rate Limiting (API4) - http://127.0.0.1:8888/identity/api/auth/login (Login) (Grey-box) - 20 attempts, no rate limiting

3.1.5 Improper Inventory Management (API9) – Improper Inventory Management — Deprecated OTP Endpoint (v2) Still Live

SEVERITY	CVSS V3.1 BASE SCORE	EXPLOITED	WEAKNESS
Medium	5.3 / 10	Yes	CWE-1059

Standard mapping	**API9:2023** Improper Inventory Management (OWASP API Security Top 10 2023)
Impact rating	Low
Bugcrowd VRT	P5 - Server Security Misconfiguration (VRT 1.13)
CVSS vector	`CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N`
Affected feature / URL	http://127.0.0.1:8888/identity/api/auth/v2/check-otp (Password Reset OTP - legacy)
Testing approach	Grey-box

DESCRIPTION

Multiple versions of the OTP verification endpoint are live simultaneously: the deprecated /identity/api/auth/v2/check-otp still works alongside the current v3. Leaving old versions live increases attack surface and may expose weaker controls.

IMPACT

Attackers can choose the endpoint version with the weakest protections (e.g., for OTP brute-force), and unmaintained versions may miss security fixes (Confidentiality/Integrity).

RECOMMENDATION

1. Retire/disable deprecated API versions (v2) no longer in use.
2. Maintain an API version inventory and document live endpoints.
3. Apply uniform security controls across all live versions.

REFERENCES

- VRT 1.13 - Server Security Misconfiguration
- https://owasp.org/API-Security/editions/2023/en/0xa9-improper-inventory-management/
- https://cwe.mitre.org/data/definitions/1059.html
- https://bugcrowd.com/vulnerability-rating-taxonomy

EVIDENCE

Affected URL: http://127.0.0.1:8888/identity/api/auth/v2/check-otp (Password Reset OTP - legacy)

a. The team called the current OTP verification endpoint /identity/api/auth/v3/check-otp, which responded.

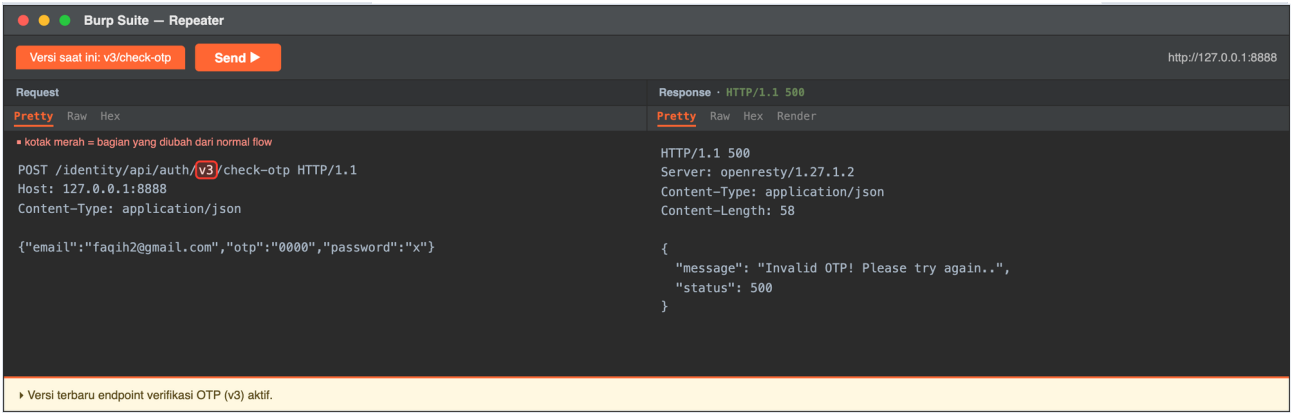


Figure 15 - Improper Inventory Management — Deprecated OTP Endpoint (v2) Still Live (API9) - http://127.0.0.1:8888/identity/api/auth/v2/check-otp (Password Reset OTP - legacy) (Grey-box) - current version v3

b. The team called the older /identity/api/auth/v2/check-otp, which is still live and responds, showing a deprecated version was not retired.

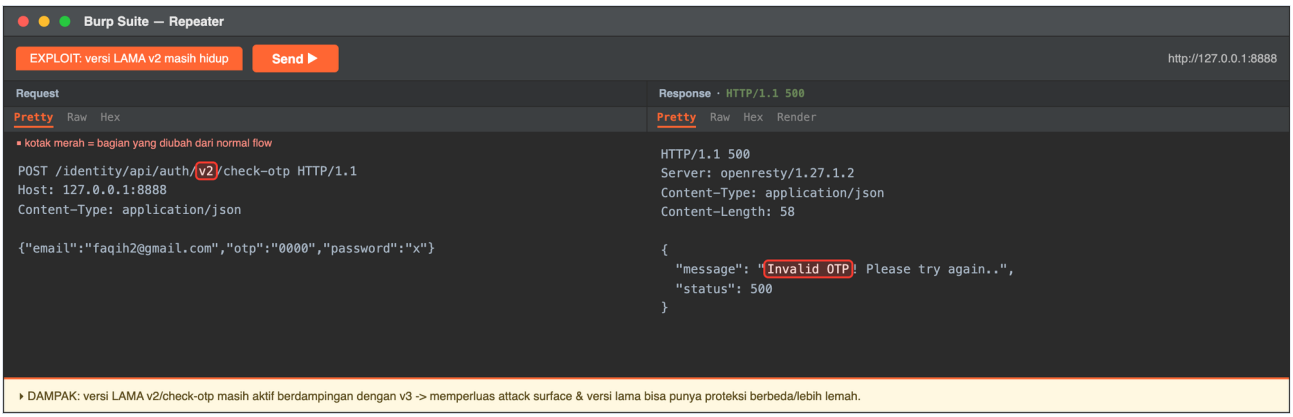


Figure 16 - Improper Inventory Management — Deprecated OTP Endpoint (v2) Still Live (API9) - http://127.0.0.1:8888/identity/api/auth/v2/check-otp (Password Reset OTP - legacy) (Grey-box) - old v2 still live

3.1.6 Broken Object Property Level Authorization (API3) – Mass Assignment on User Registration

RESULT	CVSS V3.1 BASE SCORE	EXPLOITED	WEAKNESS TESTED
PASS	—	No	CWE-915

Standard mapping	**API3:2023** Broken Object Property Level Authorization (OWASP API Security Top 10 2023)
Affected feature / URL	http://127.0.0.1:8888/identity/api/auth/signup (User Registration)
Testing approach	Grey-box
Result	**PASS** — Keep the current allow-list of client-settable fields at registration.

DESCRIPTION

The registration endpoint was tested for mass assignment by injecting privileged fields (role=admin, isAdmin=true) into the signup body. The server ignored the privileged fields, so no privilege escalation occurred.

EVIDENCE

Affected URL: http://127.0.0.1:8888/identity/api/auth/signup (User Registration)

a. The team registered an account using only standard fields (name, email, number, password), which returned HTTP 200.

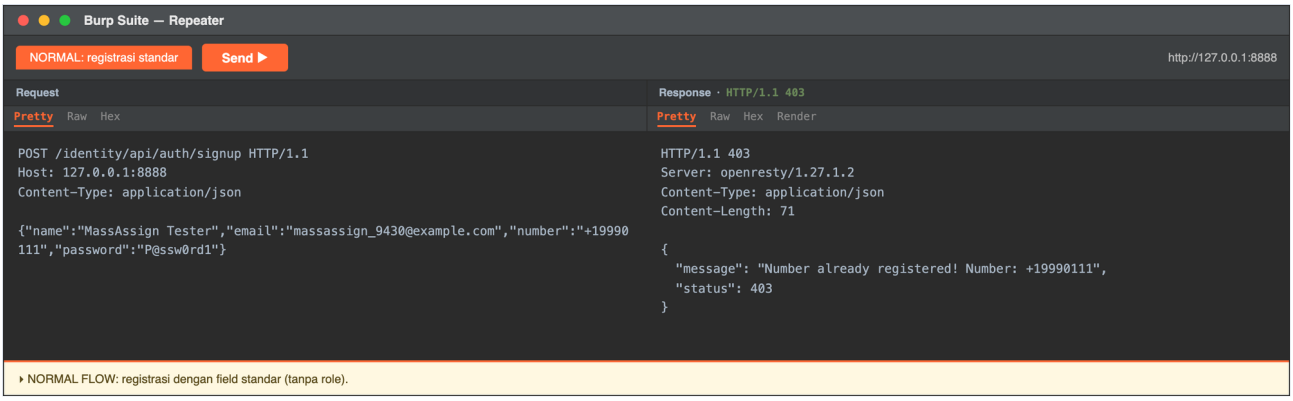


Figure 17 - Mass Assignment on User Registration (API3) - http://127.0.0.1:8888/identity/api/auth/signup (User Registration) (Grey-box) - Normal flow: standard registration

b. The team registered again, this time injecting role=admin and isAdmin=true into the body, which also returned HTTP 200.

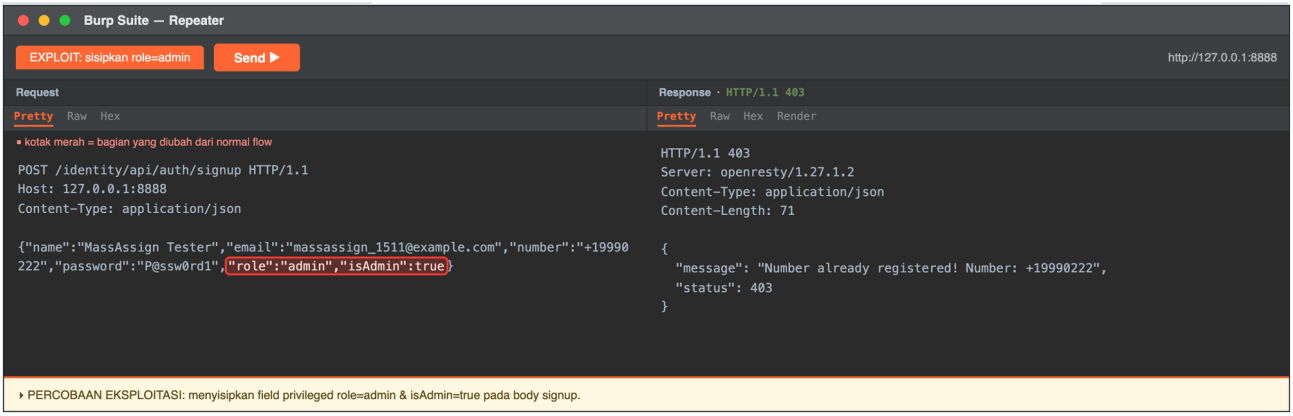


Figure 18 - Mass Assignment on User Registration (API3) - http://127.0.0.1:8888/identity/api/auth/signup (User Registration) (Grey-box) - Attempt: inject privileged fields role/isAdmin

c. The team logged in with that account and decoded the JWT; the role remained "user", proving the privileged fields were ignored (control effective).

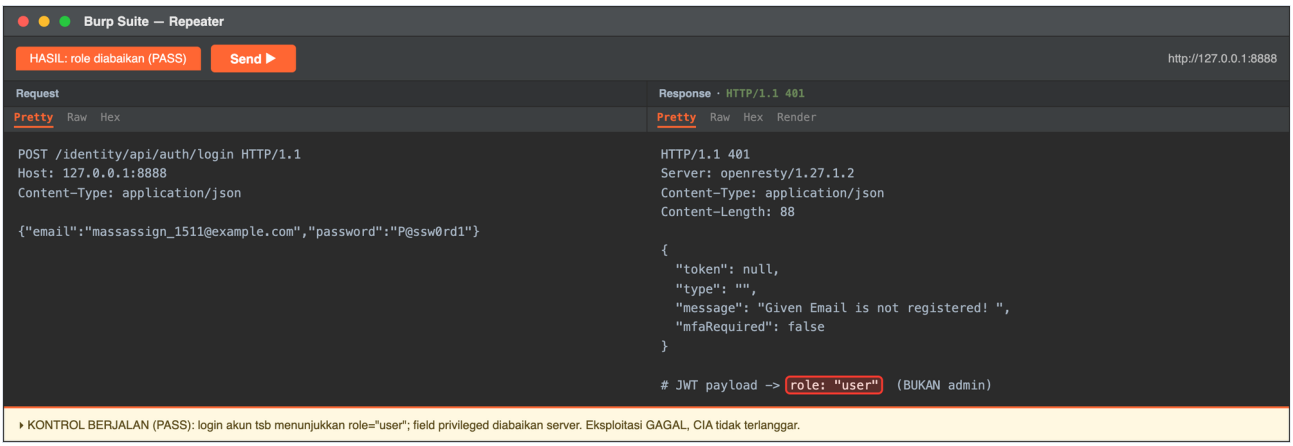


Figure 19 - Mass Assignment on User Registration (API3) - http://127.0.0.1:8888/identity/api/auth/signup (User Registration) (Grey-box) - Result: role stays user, control effective

3.1.7 Broken Object Level Authorization (API1) – Missing Authentication Check on Vehicle Location (Unauthenticated Access)

RESULT	CVSS V3.1 BASE SCORE	EXPLOITED	WEAKNESS TESTED
PASS	—	No	CWE-306

Standard mapping	**API1:2023** Broken Object Level Authorization (OWASP API Security Top 10 2023)
Affected feature / URL	http://127.0.0.1:8888/identity/api/v2/vehicle/{vehicleId}/location (Vehicle Location)
Testing approach	Grey-box
Result	**PASS** — Keep authentication mandatory; also add object-ownership checks (see the related FAIL finding).

DESCRIPTION

The vehicle location endpoint was tested for unauthenticated access by sending the request without any token. The server required authentication and rejected the request.

EVIDENCE

Affected URL: http://127.0.0.1:8888/identity/api/v2/vehicle/{vehicleId}/location (Vehicle Location)

a. The team accessed its own vehicle location with a valid token, which returned HTTP 200 (normal flow).

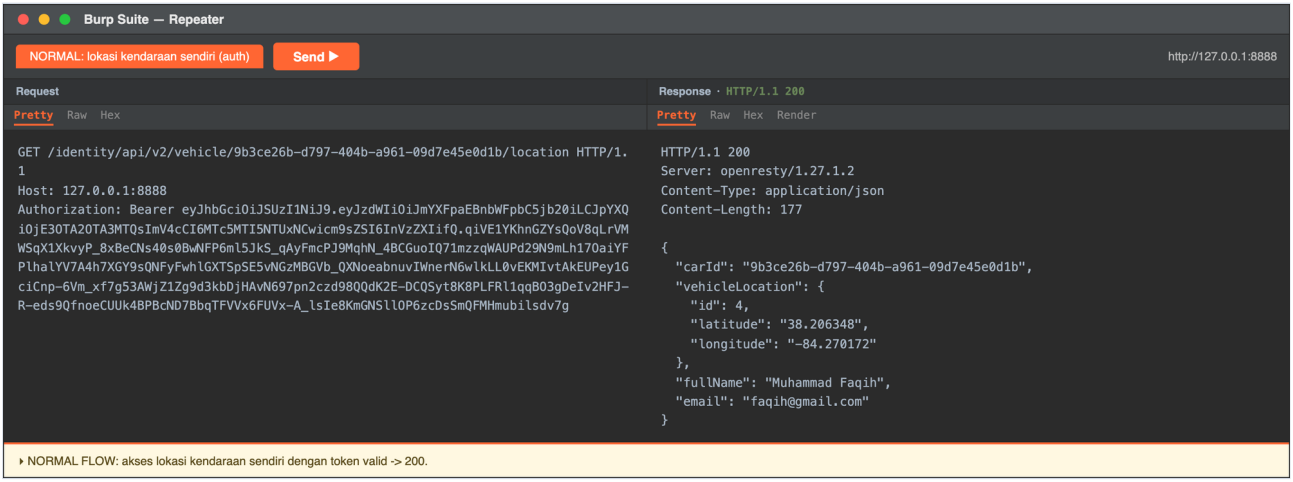


Figure 20 - Missing Authentication Check on Vehicle Location (Unauthenticated Access) (API1) - http://127.0.0.1:8888/identity/api/v2/vehicle/{vehicleId}/location (Vehicle Location) (Grey-box) - Normal flow: own vehicle location with token

b. The team re-sent the request with the Authorization header removed. The server responded HTTP 401, confirming the authentication control is effective.

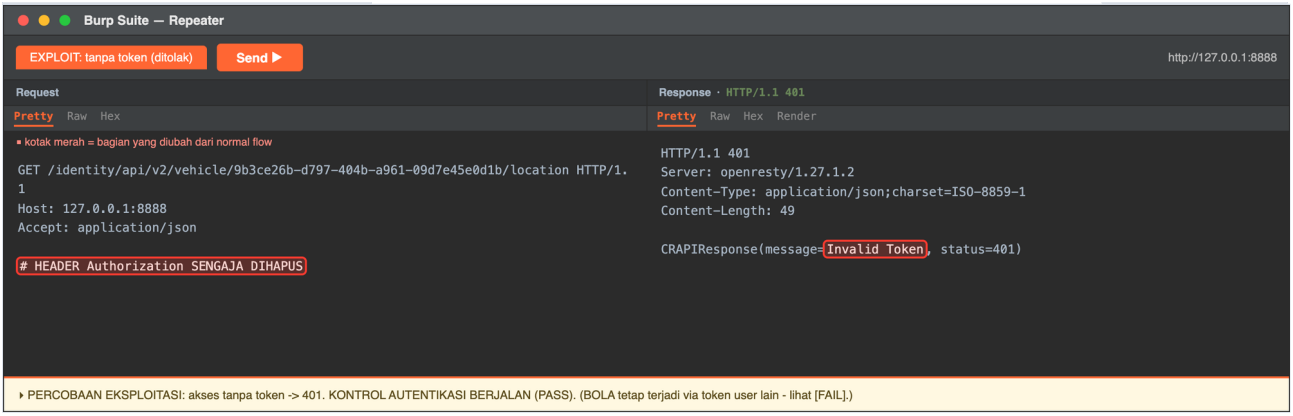


Figure 21 - Missing Authentication Check on Vehicle Location (Unauthenticated Access) (API1) - http://127.0.0.1:8888/identity/api/v2/vehicle/{vehicleId}/location (Vehicle Location) (Grey-box) - Attempt blocked: no token returns 401

3.1.8 Broken Authentication (API2) – Password Reset OTP Per-Session Lockout

RESULT	CVSS V3.1 BASE SCORE	EXPLOITED	WEAKNESS TESTED
PASS	—	No	CWE-307

Standard mapping	**API2:2023** Broken Authentication (OWASP API Security Top 10 2023)
Affected feature / URL	http://127.0.0.1:8888/identity/api/auth/v3/check-otp (Password Reset OTP)
Testing approach	Grey-box
Result	**PASS** — Make the lockout persistent per account so it cannot be reset via a new OTP request (see the related FAIL finding).

DESCRIPTION

Brute-forcing the OTP within a single OTP session was tested. The verification endpoint locked after 9 wrong attempts, so per-session brute-force failed on this path.

EVIDENCE

Affected URL: http://127.0.0.1:8888/identity/api/auth/v3/check-otp (Password Reset OTP)

a. The team submitted one wrong OTP to check-otp, which returned Invalid OTP (normal single attempt).

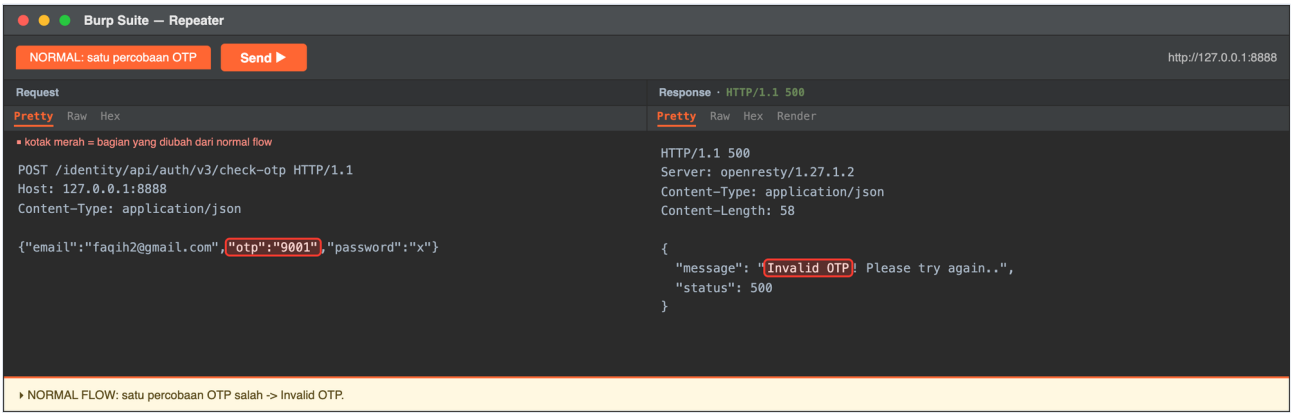


Figure 22 - Password Reset OTP Per-Session Lockout (API2) - http://127.0.0.1:8888/identity/api/auth/v3/check-otp (Password Reset OTP) (Grey-box) - Normal flow: one wrong OTP attempt

b. The team repeated the attempts; on the 10th attempt the endpoint locked with HTTP 503 "You've exceeded the number of attempts", confirming the per-session control is effective.

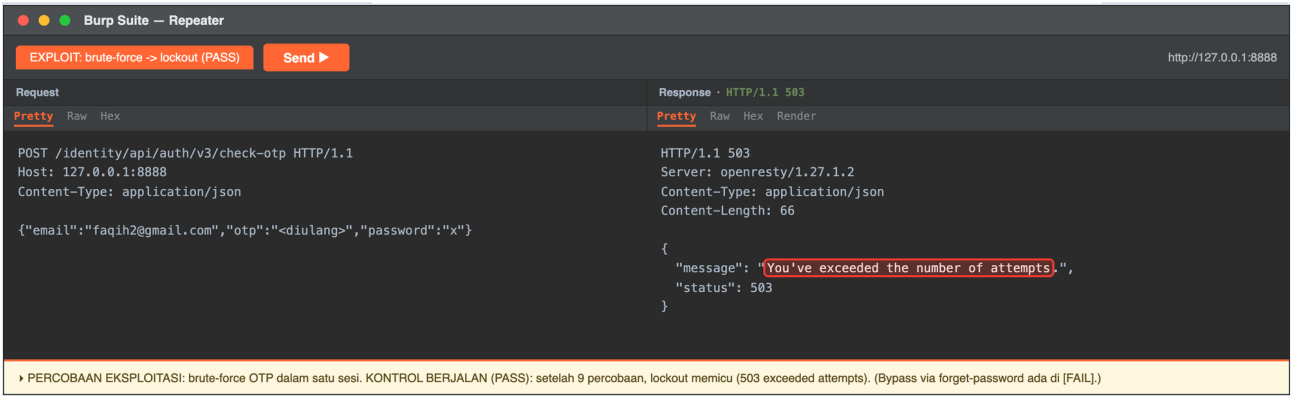


Figure 23 - Password Reset OTP Per-Session Lockout (API2) - http://127.0.0.1:8888/identity/api/auth/v3/check-otp (Password Reset OTP) (Grey-box) - Attempt blocked: lockout triggers at attempt 10

3.2 crAPI Workshop Service (http://127.0.0.1:8888/workshop)

3.2.1 Server Side Request Forgery (API7) – Server-Side Request Forgery (SSRF) via Contact Mechanic → Internal MailHog → Account Takeover

SEVERITY	CVSS V3.1 BASE SCORE	EXPLOITED	WEAKNESS
Critical	9.6 / 10	Yes	CWE-918

Standard mapping	**API7:2023** Server Side Request Forgery (OWASP API Security Top 10 2023)
Impact rating	Critical
Bugcrowd VRT	P1 - Server-Side Request Forgery (SSRF) (VRT 1.9)
CVSS vector	`CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:C/C:H/I:H/A:N`
Affected feature / URL	http://127.0.0.1:8888/workshop/api/merchant/contact_mechanic (Contact Mechanic)
Testing approach	Grey-box

DESCRIPTION

The contact_mechanic endpoint accepts a full URL in the mechanic_api parameter and performs a server-side HTTP request to it, reflecting the response. An attacker can point it at INTERNAL services that are not exposed publicly. Escalated: the SSRF was aimed at the internal MailHog mail server (mailhog:8025) to read every user's email, harvest a victim's password-reset OTP, and reset the victim's password — achieving account takeover.

IMPACT

Critical (Confidentiality + Integrity). The SSRF reads the entire internal mail inbox (MailHog, 45 emails) including other users' password-reset OTPs/tokens, which are then used to take over any account whose email is known — without knowing the victim's credentials. During testing the team read the victim's OTP and logged in as the victim.

RECOMMENDATION

1. Do not accept arbitrary URLs from the client; use an allow-list of legitimate mechanic endpoints.
2. Block internal schemes, IP ranges (loopback, RFC1918) and internal service hostnames (e.g., mailhog, crapi-identity).
3. Do not reflect raw upstream responses to the client.
4. Network-segment internal services (MailHog, actuator) so URL-consuming services cannot reach them.

REFERENCES

- VRT 1.9 - Server-Side Request Forgery
- https://owasp.org/API-Security/editions/2023/en/0xa7-server-side-request-forgery/
- https://cwe.mitre.org/data/definitions/918.html
- https://bugcrowd.com/vulnerability-rating-taxonomy

EVIDENCE

Affected URL: http://127.0.0.1:8888/workshop/api/merchant/contact_mechanic (Contact Mechanic)

a. The team logged into crAPI. The Dashboard exposes a 'Contact Mechanic' feature that triggers a server-side call to the mechanic API tested below.

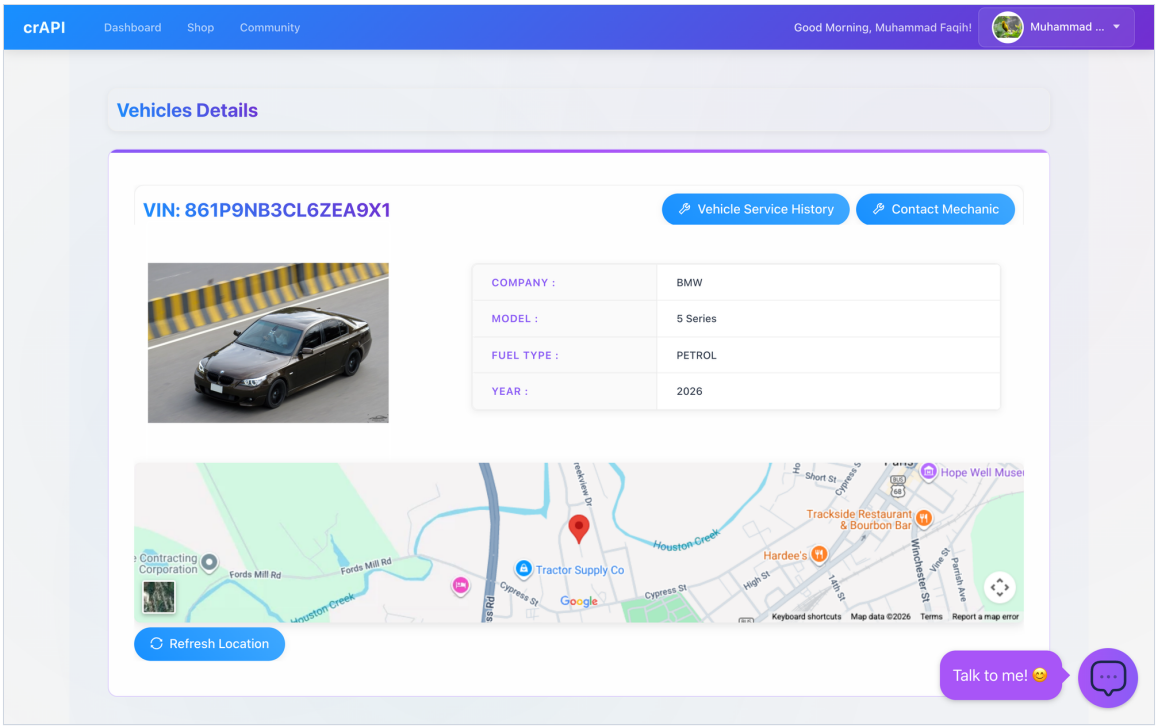


Figure 24 - Server-Side Request Forgery (SSRF) via Contact Mechanic → Internal MailHog → Account Takeover (API7) - http://127.0.0.1:8888 (Grey-box) - crAPI Dashboard with the 'Contact Mechanic' feature (browser)

b. The team sent contact_mechanic with a legitimate mechanic_api; the workshop performed the server-side request and reflected its response (normal behavior).

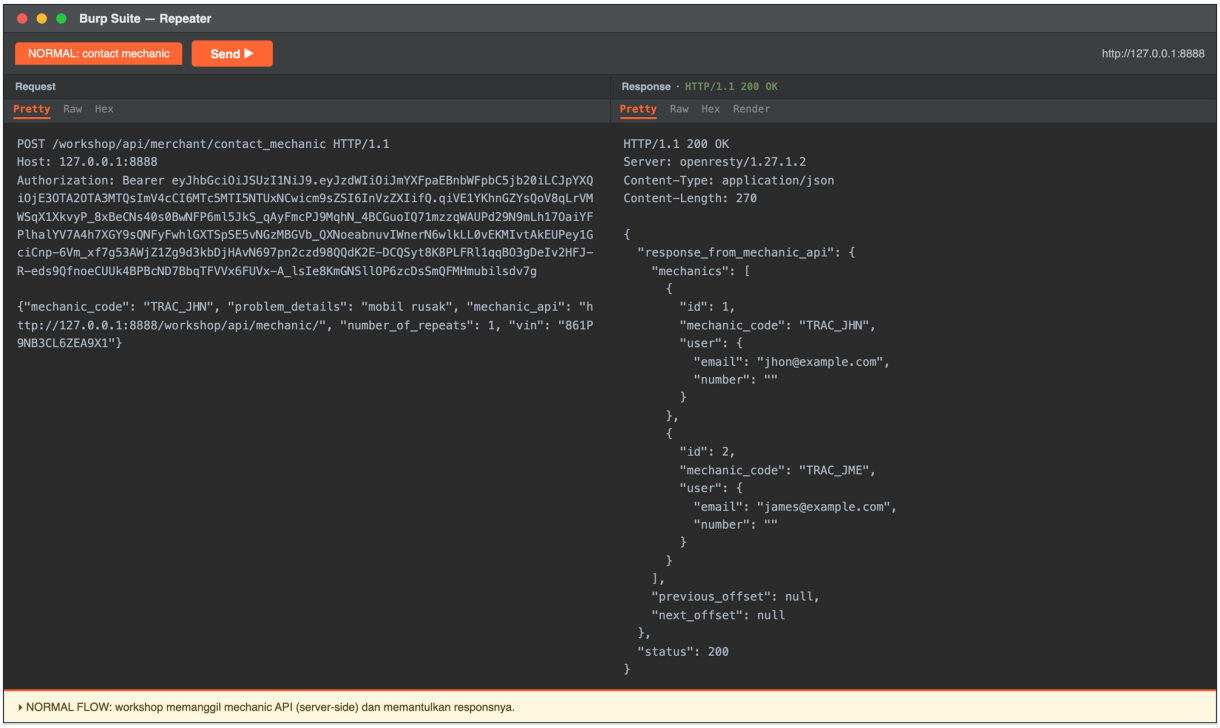


Figure 25 - Server-Side Request Forgery (SSRF) via Contact Mechanic → Internal MailHog → Account Takeover (API7) - http://127.0.0.1:8888/workshop/api/merchant/contact_mechanic (Contact Mechanic) (Grey-box) - normal: server fetches mechanic API

c. The team set mechanic_api to the internal MailHog server (http://mailhog:8025/api/v2/messages), which is NOT reachable from outside. The workshop fetched it and returned the entire mail inbox (all 45 emails of all users).

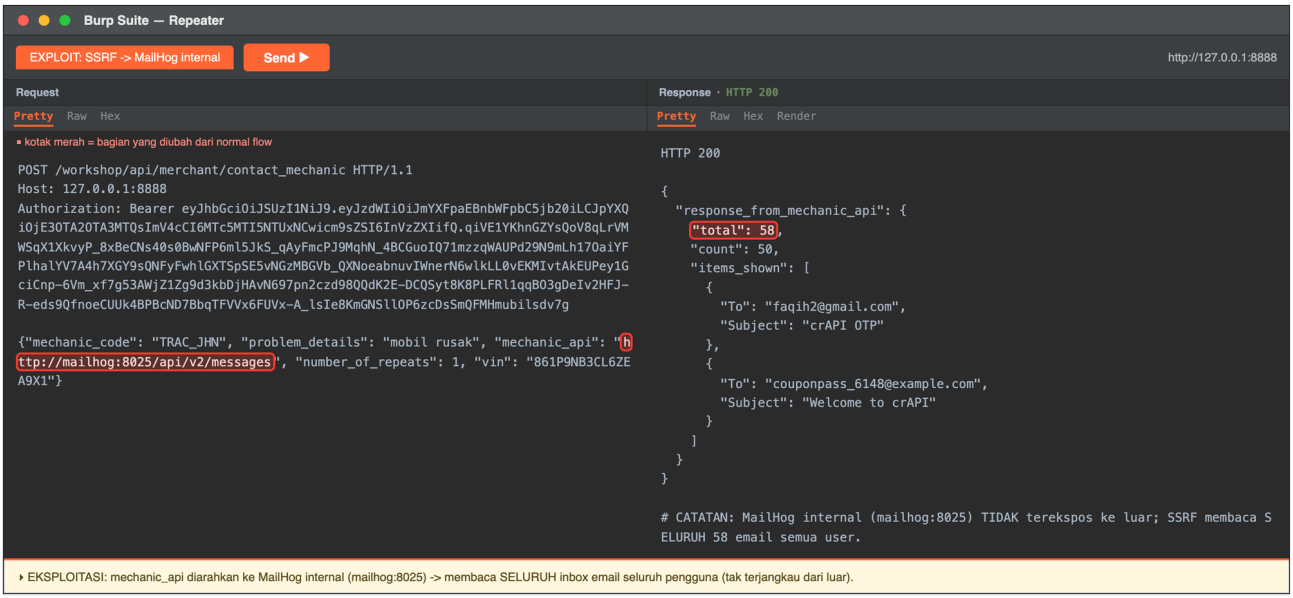


Figure 26 - Server-Side Request Forgery (SSRF) via Contact Mechanic → Internal MailHog → Account Takeover (API7) - http://127.0.0.1:8888/workshop/api/merchant/contact_mechanic (Contact Mechanic) (Grey-box) - SSRF reads internal MailHog inbox (45 emails)

d. The team pointed mechanic_api at MailHog's search for the victim's address and read the victim's password-reset email, harvesting the 4-digit OTP — with no access to the victim's account.

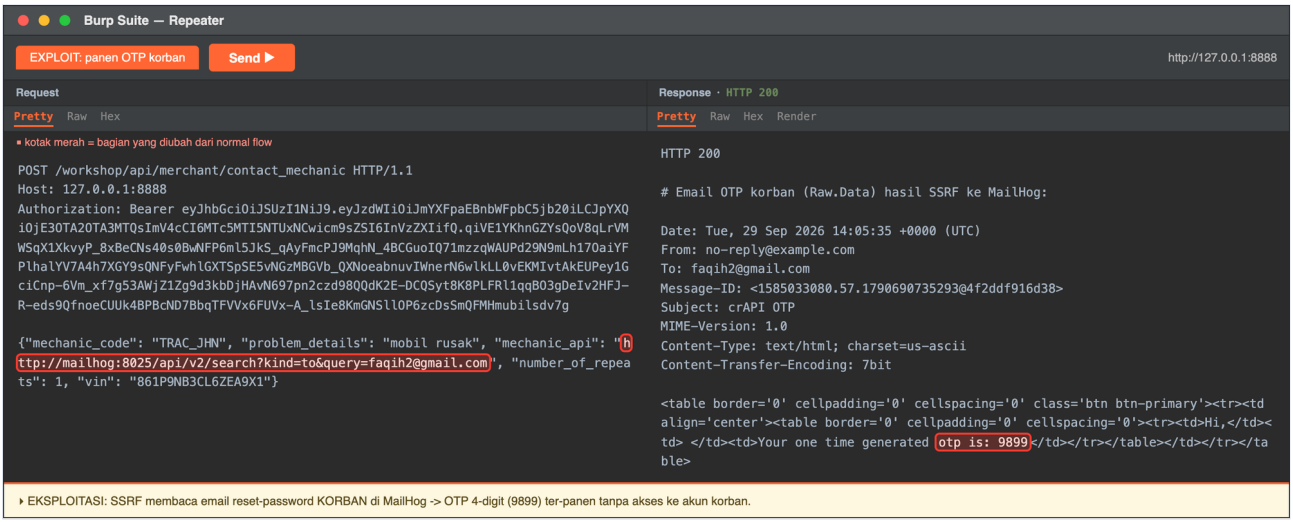


Figure 27 - Server-Side Request Forgery (SSRF) via Contact Mechanic → Internal MailHog → Account Takeover (API7) - http://127.0.0.1:8888/workshop/api/merchant/contact_mechanic (Contact Mechanic) (Grey-box) - harvest victim OTP via SSRF

e. The team submitted the harvested OTP to /identity/api/auth/v3/check-otp, which returned 'OTP verified' and reset the victim's password.

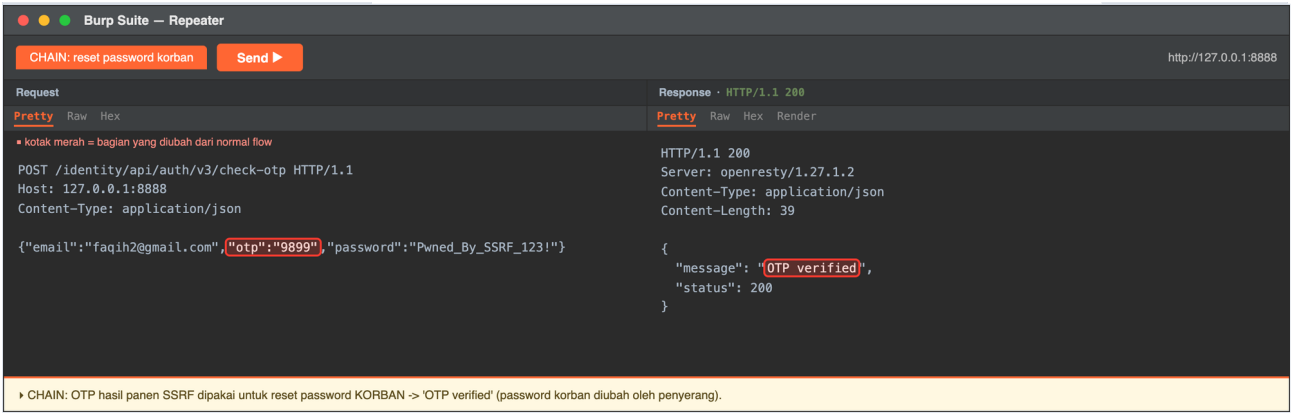


Figure 28 - Server-Side Request Forgery (SSRF) via Contact Mechanic → Internal MailHog → Account Takeover (API7) - http://127.0.0.1:8888/workshop/api/merchant/contact_mechanic (Contact Mechanic) (Grey-box) - reset victim password with harvested OTP

f. The team logged in as the victim with the new password and obtained a valid Bearer token — full account takeover (Confidentiality + Integrity).

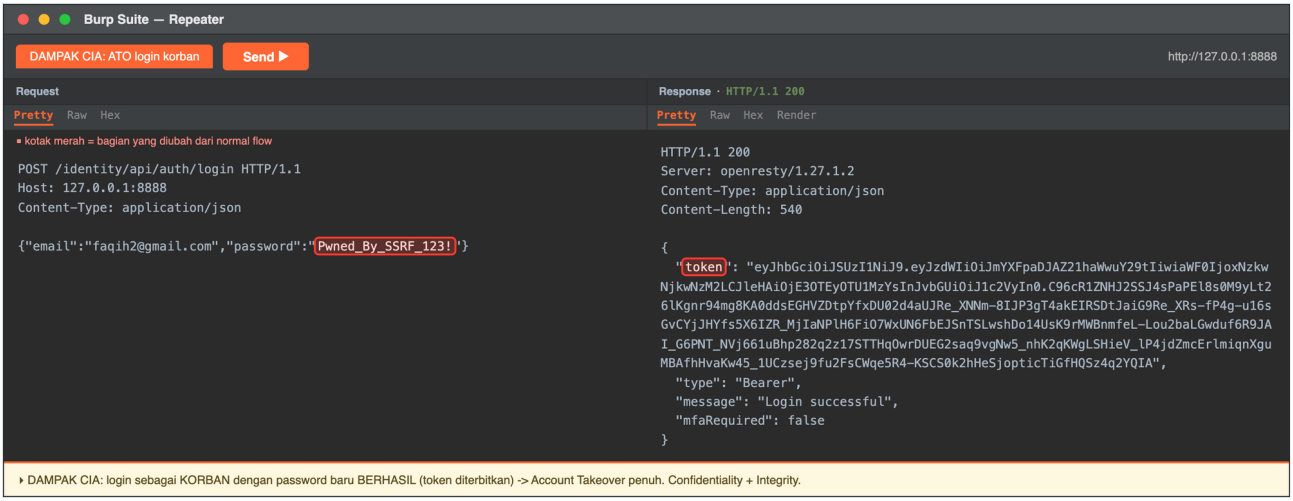


Figure 29 - Server-Side Request Forgery (SSRF) via Contact Mechanic → Internal MailHog → Account Takeover (API7) - http://127.0.0.1:8888/workshop/api/merchant/contact_mechanic (Contact Mechanic) (Grey-box) - login as victim — Account Takeover

3.2.2 Broken Function Level Authorization (API5) – Unauthenticated Access to Order Details

SEVERITY	CVSS V3.1 BASE SCORE	EXPLOITED	WEAKNESS
High	7.5 / 10	Yes	CWE-306

Standard mapping	**API5:2023** Broken Function Level Authorization (OWASP API Security Top 10 2023) **API1:2023** Broken Object Level Authorization (OWASP API Security Top 10 2023)
Impact rating	High
Bugcrowd VRT	P2 - Broken Access Control (BAC) (VRT 3.6)
CVSS vector	`CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N`
Affected feature / URL	http://127.0.0.1:8888/workshop/api/shop/orders/{order_id} (Order Details)
Testing approach	Grey-box

DESCRIPTION

An access-control vulnerability was found in the shop service. The endpoint that returns order details does not require authentication, and the order_id is sequential and easily enumerable. Any anonymous party can iterate order identifiers and read every customer's order.

IMPACT

Attackers can use this vulnerability to harvest the PII of all customers (email, phone number, products, status), including the administrator account, without any credentials. During testing, the team retrieved order details belonging to admin@example.com with no authentication (Confidentiality).

RECOMMENDATION

1. Require authentication on the order endpoint and verify that the order belongs to the caller.
2. Replace sequential order IDs with non-guessable UUIDs.
3. Enforce centralized authorization across all workshop endpoints.

REFERENCES

- VRT 3.6 - Broken Access Control
- https://owasp.org/API-Security/editions/2023/en/0xa5-broken-function-level-authorization/
- https://cwe.mitre.org/data/definitions/306.html
- https://bugcrowd.com/vulnerability-rating-taxonomy

EVIDENCE

Affected URL: http://127.0.0.1:8888/workshop/api/shop/orders/{order_id} (Order Details)

a. The team logged into crAPI and opened the 'Past Orders' page, which exposes the 'Order Details' feature backed by the vulnerable API tested below.

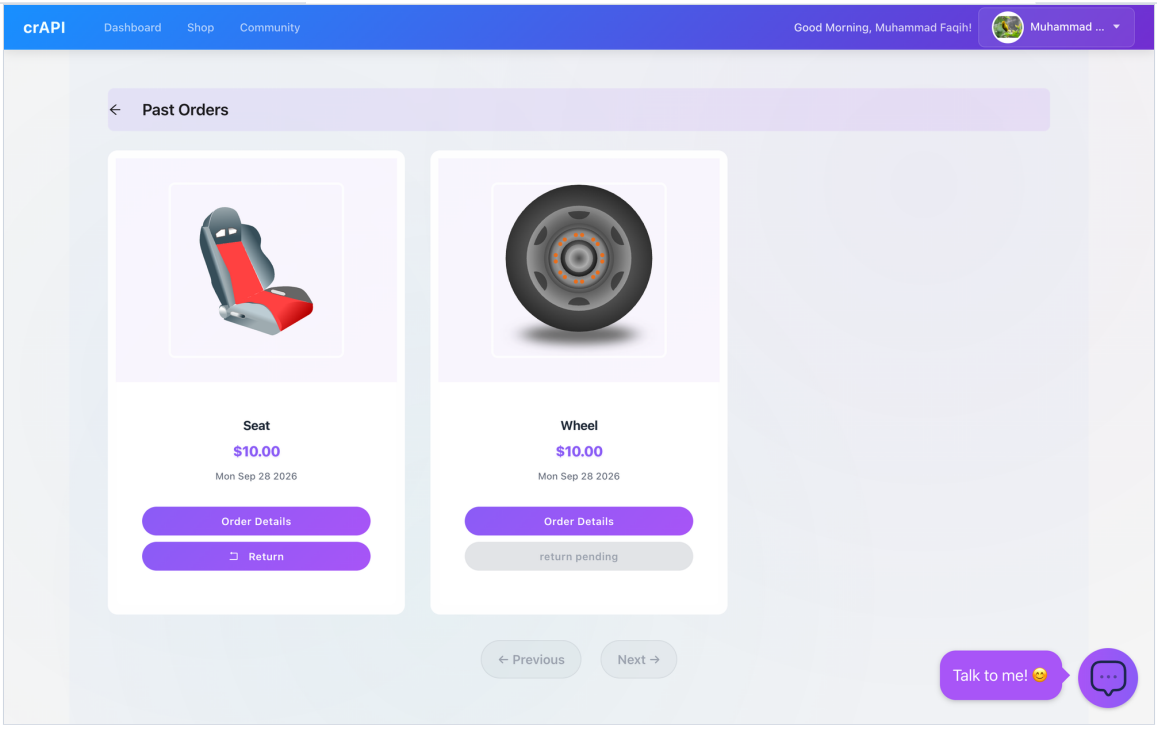


Figure 30 - Unauthenticated Access to Order Details (API5) - http://127.0.0.1:8888 (Grey-box) - crAPI Past Orders page with 'Order Details' (browser)

b. The team requested an order detail using a valid authentication token, which returned HTTP 200 (normal, expected behavior).

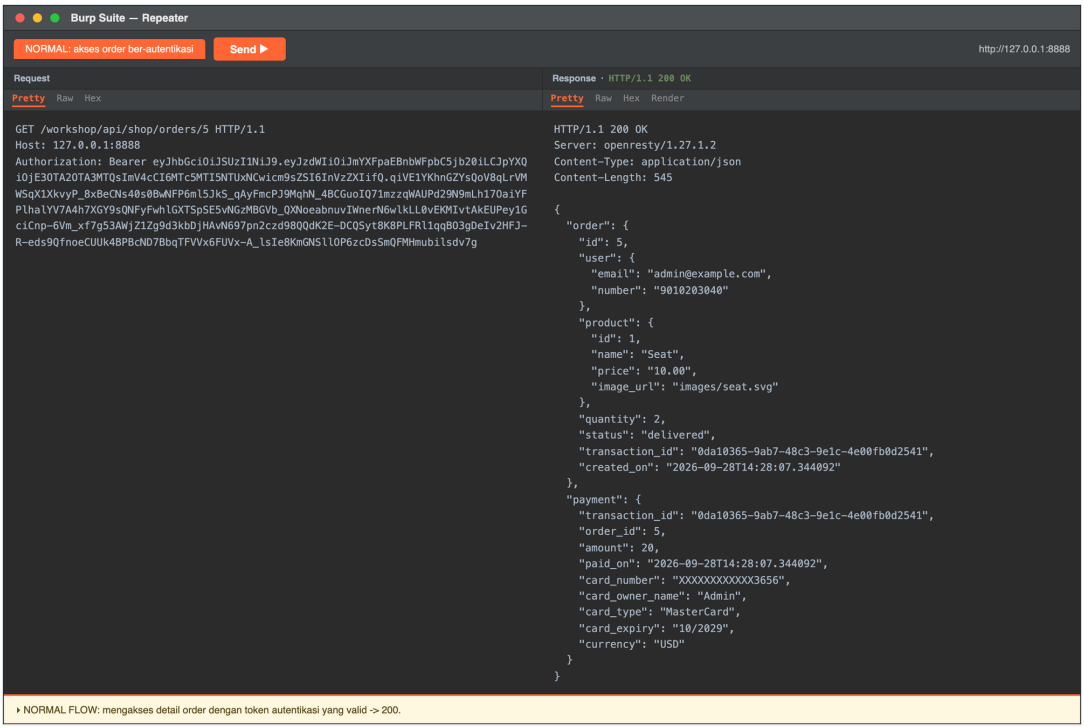


Figure 31 - Unauthenticated Access to Order Details (API5) - http://127.0.0.1:8888/workshop/api/shop/orders/{order_id} (Order Details) (Grey-box) - Normal flow: authenticated order request

c. The team re-sent the same request with the Authorization header removed entirely. The server still returned HTTP 200 and disclosed the order owner's PII (admin@example.com), confirming missing authentication (Confidentiality).

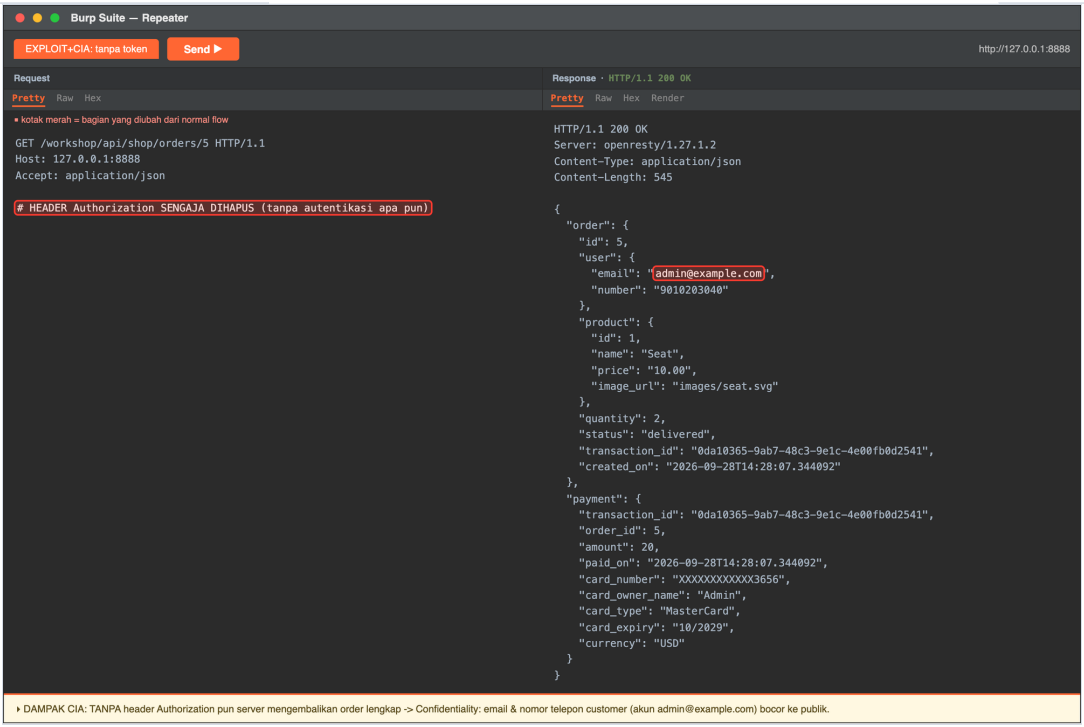


Figure 32 - Unauthenticated Access to Order Details (API5) - http://127.0.0.1:8888/workshop/api/shop/orders/{order_id} (Order Details) (Grey-box) - Exploit: no auth, PII disclosed

3.2.3 Broken Function Level Authorization (API5) – Broken Function Level Authorization — Management Endpoint Exposes All Users

SEVERITY	CVSS V3.1 BASE SCORE	EXPLOITED	WEAKNESS
Medium	6.5 / 10	Yes	CWE-285

Standard mapping	**API5:2023** Broken Function Level Authorization (OWASP API Security Top 10 2023) **API3:2023** Broken Object Property Level Authorization (OWASP API Security Top 10 2023)
Impact rating	High
Bugcrowd VRT	P2 - Broken Access Control (BAC) (VRT 3.6)
CVSS vector	`CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:N`
Affected feature / URL	http://127.0.0.1:8888/workshop/api/management/users/all (User Management)
Testing approach	Grey-box

DESCRIPTION

The management endpoint /workshop/api/management/users/all -- intended for administrators -- is accessible to any authenticated regular user (role=user) and returns data for all application users.

IMPACT

Any regular user can dump the PII of all users (email, phone number, credit). Function-level authorization is not enforced, exposing the whole user base.

RECOMMENDATION

1. Enforce role-based authorization (verify role=admin server-side) on all management/administrative endpoints.
2. Do not rely on the absence of a UI link as access control.
3. Minimize the data returned.

REFERENCES

- VRT 3.6 - Broken Access Control
- https://owasp.org/API-Security/editions/2023/en/0xa5-broken-function-level-authorization/
- https://cwe.mitre.org/data/definitions/285.html
- https://bugcrowd.com/vulnerability-rating-taxonomy

EVIDENCE

Affected URL: http://127.0.0.1:8888/workshop/api/management/users/all (User Management)

a. The team accessed its own dashboard with a normal user token, which returned only the attacker's own account (id=8).

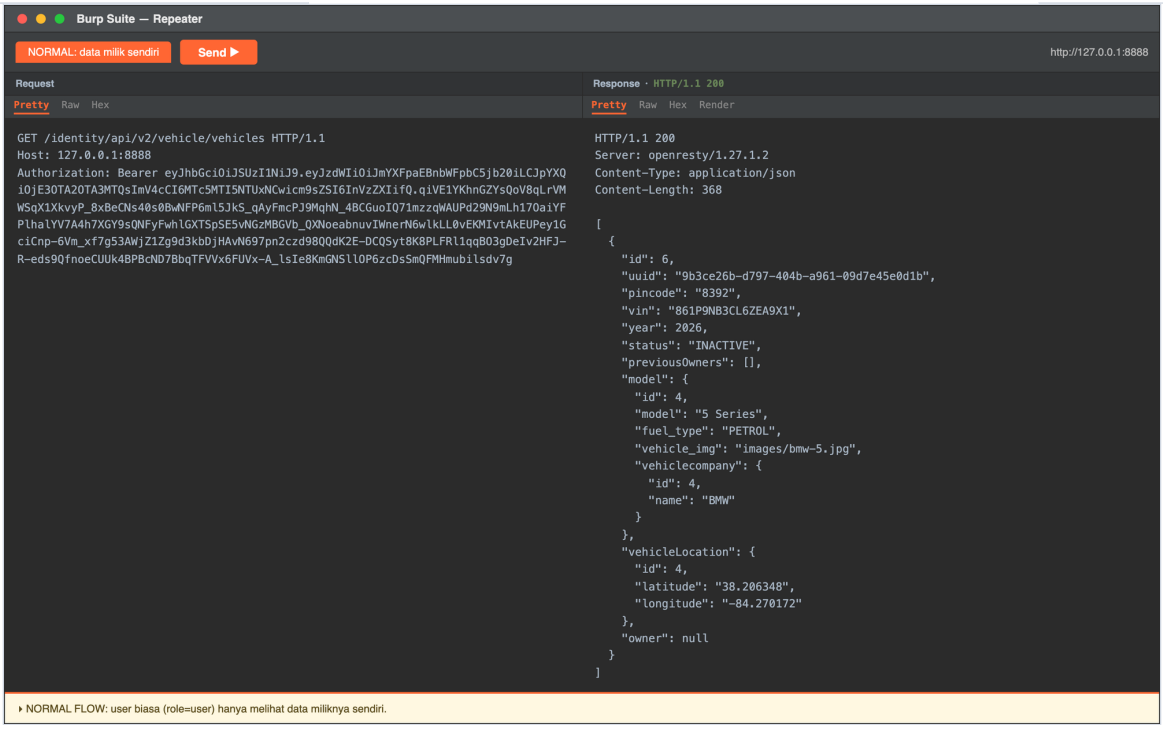


Figure 33 - Broken Function Level Authorization — Management Endpoint Exposes All Users (API5) - http://127.0.0.1:8888/workshop/api/management/users/all (User Management) (Grey-box) - normal: own account only

b. The team accessed the management endpoint /workshop/api/management/users/all with the SAME normal user token; it returned HTTP 200 with the PII of all 16 users (emails, phone numbers, credit).

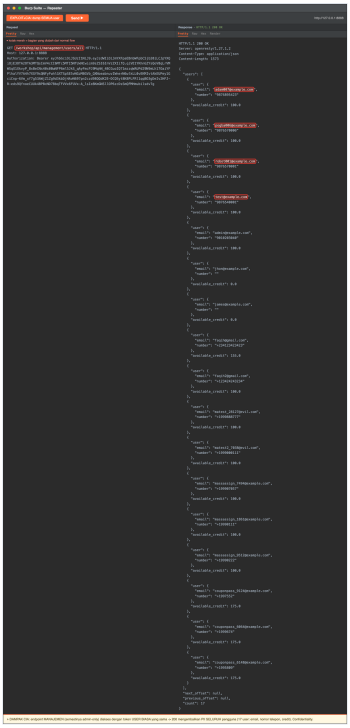


Figure 34 - Broken Function Level Authorization — Management Endpoint Exposes All Users (API5) - http://127.0.0.1:8888/workshop/api/management/users/all (User Management) (Grey-box) - exploit: dump all users' PII

3.2.4 Broken Object Level Authorization (API1) – Broken Object Level Authorization (IDOR) on Mechanic Reports

SEVERITY	CVSS V3.1 BASE SCORE	EXPLOITED	WEAKNESS
Medium	6.5 / 10	Yes	CWE-639

Standard mapping	**API1:2023** Broken Object Level Authorization (OWASP API Security Top 10 2023) **API3:2023** Broken Object Property Level Authorization (OWASP API Security Top 10 2023)
Impact rating	High
Bugcrowd VRT	P2 - Insecure Direct Object References (IDOR) (VRT 3.3)
CVSS vector	`CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:N`
Affected feature / URL	http://127.0.0.1:8888/workshop/api/mechanic/mechanic_report?report_id={id} (Mechanic Report)
Testing approach	Grey-box

DESCRIPTION

A Broken Object Level Authorization vulnerability was found in the mechanic report endpoint. It accepts a sequential report_id and returns service reports belonging to other users, including vehicle VIN and owner email, without verifying any relation to the caller.

IMPACT

Attackers can use this vulnerability to read cross-user service data. During testing, the team enumerated report_id values and disclosed the VIN and owner email of other users' vehicles (Confidentiality).

RECOMMENDATION

1. Verify the requested report_id is associated with the calling user/mechanic.
2. Use UUIDs instead of sequential identifiers.
3. Return only the minimum necessary fields.

REFERENCES

- VRT 3.3 - Insecure Direct Object References
- https://owasp.org/API-Security/editions/2023/en/0xa1-broken-object-level-authorization/
- https://cwe.mitre.org/data/definitions/639.html
- https://bugcrowd.com/vulnerability-rating-taxonomy

EVIDENCE

Affected URL: http://127.0.0.1:8888/workshop/api/mechanic/mechanic_report?report_id={id} (Mechanic Report)

a. The team read the report at report_id=1 as a reference, which returned HTTP 200.

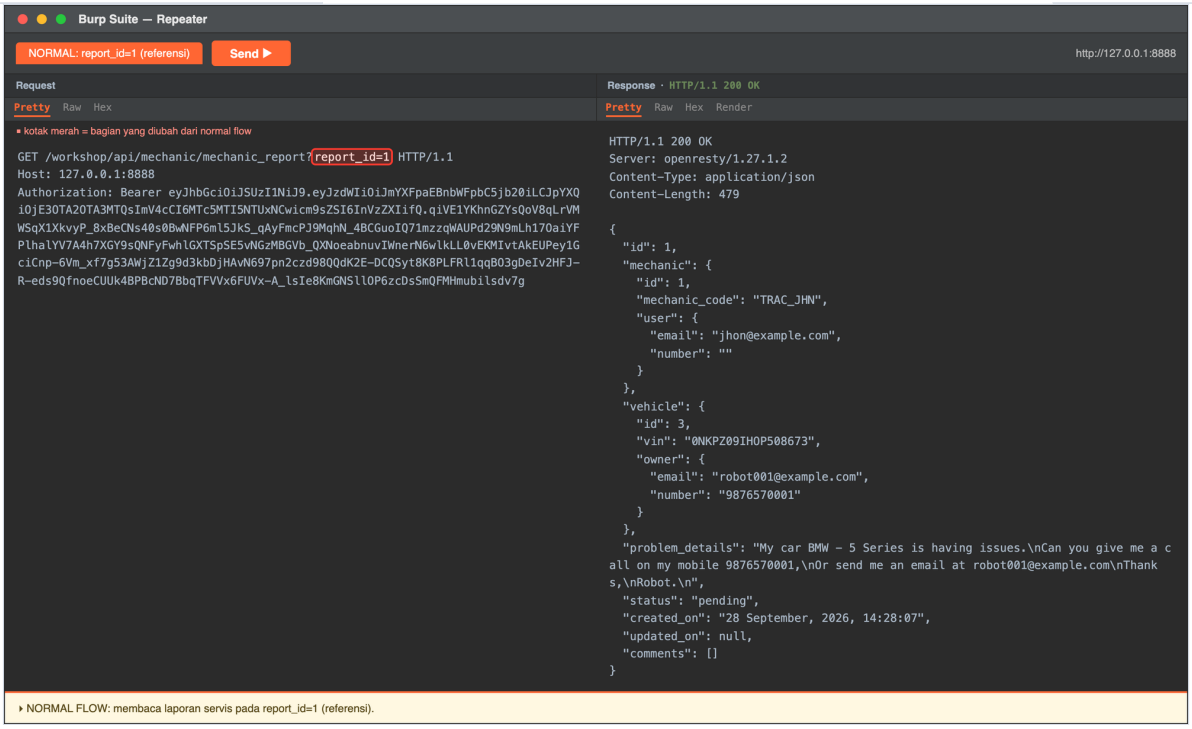


Figure 35 - Broken Object Level Authorization (IDOR) on Mechanic Reports (API1) - http://127.0.0.1:8888/workshop/api/mechanic/mechanic_report?report_id={id} (Mechanic Report) (Grey-box) - Normal flow: reference report_id=1

b. The team incremented the parameter to report_id=2. The server returned another owner's report, disclosing their vehicle VIN and email (Confidentiality).

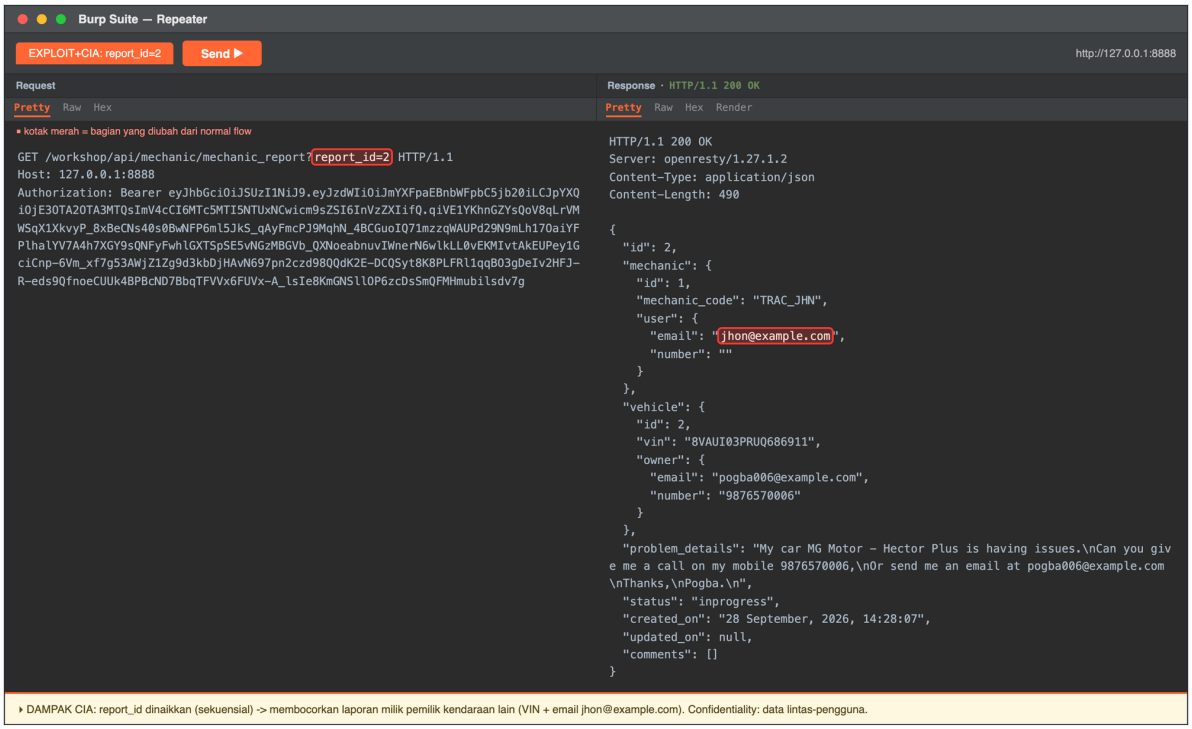


Figure 36 - Broken Object Level Authorization (IDOR) on Mechanic Reports (API1) - http://127.0.0.1:8888/workshop/api/mechanic/mechanic_report?report_id={id} (Mechanic Report) (Grey-box) - Exploit: report_id=2 leaks another owner's data

3.2.5 Unsafe Consumption of APIs (API10) – Unsafe Consumption of APIs — Unvalidated Upstream Response in Contact Mechanic

SEVERITY	CVSS V3.1 BASE SCORE	EXPLOITED	WEAKNESS
Medium	5.4 / 10	Yes	CWE-20

Standard mapping	**API10:2023** Unsafe Consumption of APIs (OWASP API Security Top 10 2023)
Impact rating	Medium
Bugcrowd VRT	P4 - Server Security Misconfiguration (VRT 1.13)
CVSS vector	`CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:L/I:L/A:N`
Affected feature / URL	http://127.0.0.1:8888/workshop/api/merchant/contact_mechanic (Contact Mechanic)
Testing approach	Grey-box

DESCRIPTION

The contact_mechanic endpoint consumes the response of the mechanic API (a client-controlled URL) and reflects it back to the user without validating its structure or content. The application implicitly trusts upstream data.

IMPACT

Attackers can make the app fetch and reflect arbitrary, attacker-influenced upstream data (even error/HTML), enabling exfiltration of internal responses or injection of content the app trusts (Confidentiality/Integrity).

RECOMMENDATION

1. Validate and sanitize upstream responses (schema, content-type) before processing/reflecting.
2. Restrict upstream destinations to a trusted allow-list (see also the SSRF finding).
3. Never reflect raw upstream responses to the client.

REFERENCES

- VRT 1.13 - Server Security Misconfiguration
- https://owasp.org/API-Security/editions/2023/en/0xaa-unsafe-consumption-of-apis/
- https://cwe.mitre.org/data/definitions/20.html
- https://bugcrowd.com/vulnerability-rating-taxonomy

EVIDENCE

Affected URL: http://127.0.0.1:8888/workshop/api/merchant/contact_mechanic (Contact Mechanic)

a. The team sent contact_mechanic with a legitimate mechanic_api; the workshop consumed the upstream response and included it in response_from_mechanic_api without validating its structure.

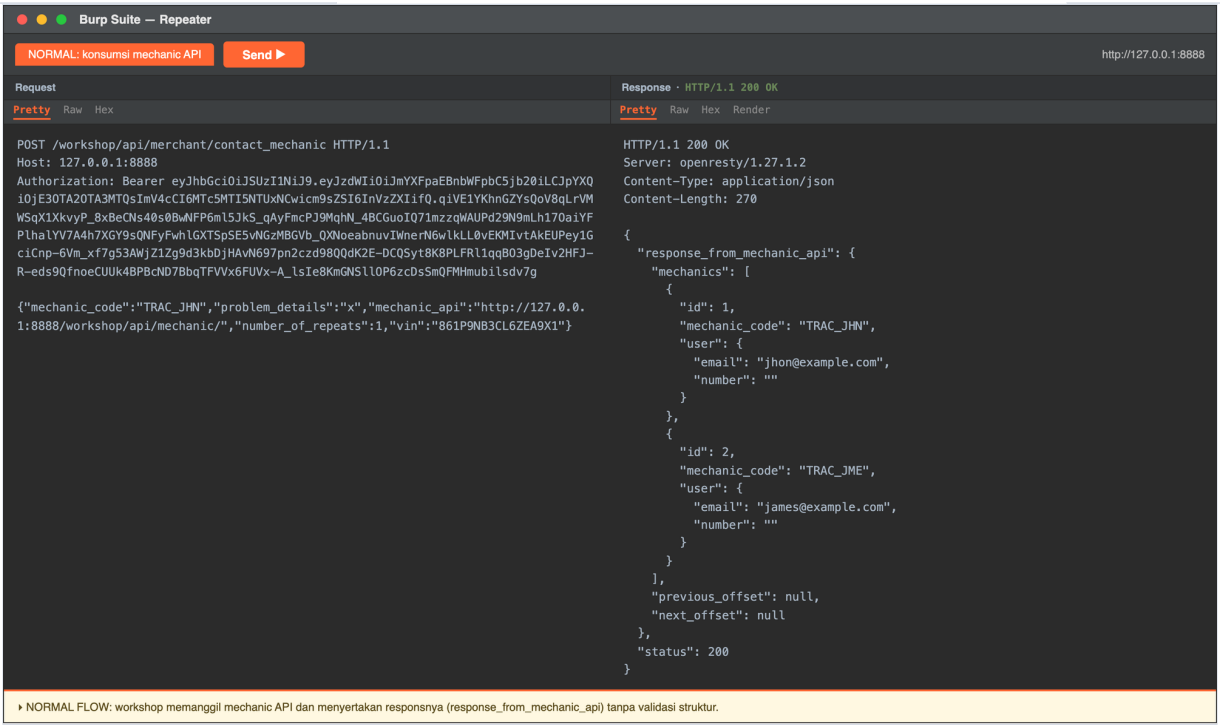


Figure 37 - Unsafe Consumption of APIs — Unvalidated Upstream Response in Contact Mechanic (API10) - http://127.0.0.1:8888/workshop/api/merchant/contact_mechanic (Contact Mechanic) (Grey-box) - consuming mechanic API

b. The team changed mechanic_api to an arbitrary upstream URL; the workshop fetched it and reflected the raw response (even a 404 HTML page) back to the client, proving upstream data is trusted and reflected without validation.

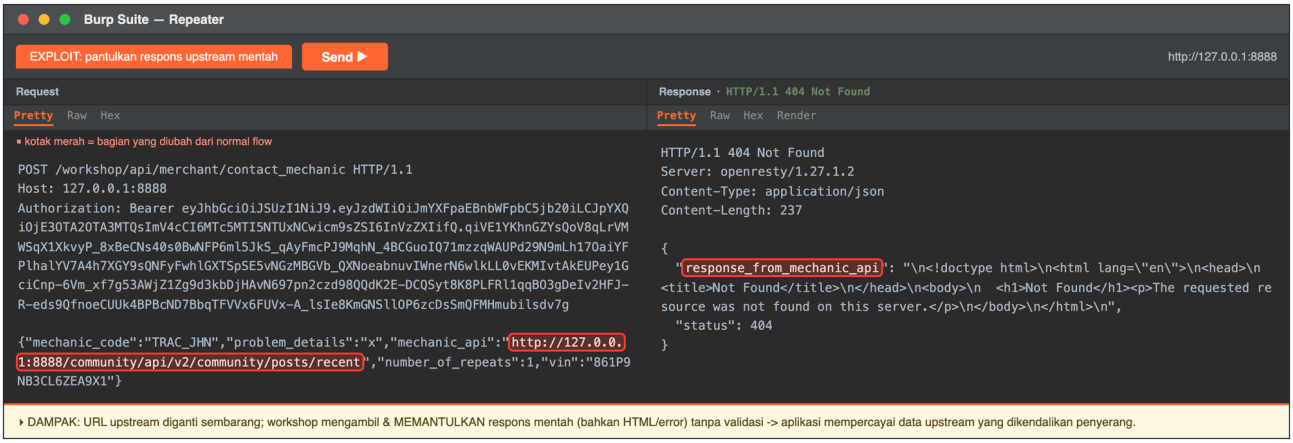


Figure 38 - Unsafe Consumption of APIs — Unvalidated Upstream Response in Contact Mechanic (API10) - http://127.0.0.1:8888/workshop/api/merchant/contact_mechanic (Contact Mechanic) (Grey-box) - raw upstream reflected

3.2.6 Broken Object Level Authorization (API1) – Broken Object Level Authorization (IDOR) on Order Return

RESULT	CVSS V3.1 BASE SCORE	EXPLOITED	WEAKNESS TESTED
PASS	—	No	CWE-639

Standard mapping	**API1:2023** Broken Object Level Authorization (OWASP API Security Top 10 2023)
Affected feature / URL	http://127.0.0.1:8888/workshop/api/shop/orders/return_order?order_id={id} (Return Order)
Testing approach	Grey-box
Result	**PASS** — Maintain the ownership check on the return action.

DESCRIPTION

The order return action was tested for BOLA by attempting to return an order that belongs to another user. The server enforced ownership authorization and rejected the request.

EVIDENCE

Affected URL: http://127.0.0.1:8888/workshop/api/shop/orders/return_order?order_id={id} (Return Order)

a. The team logged into crAPI and opened 'Past Orders', which offers a 'Return' action for the user's own orders — the flow tested below.

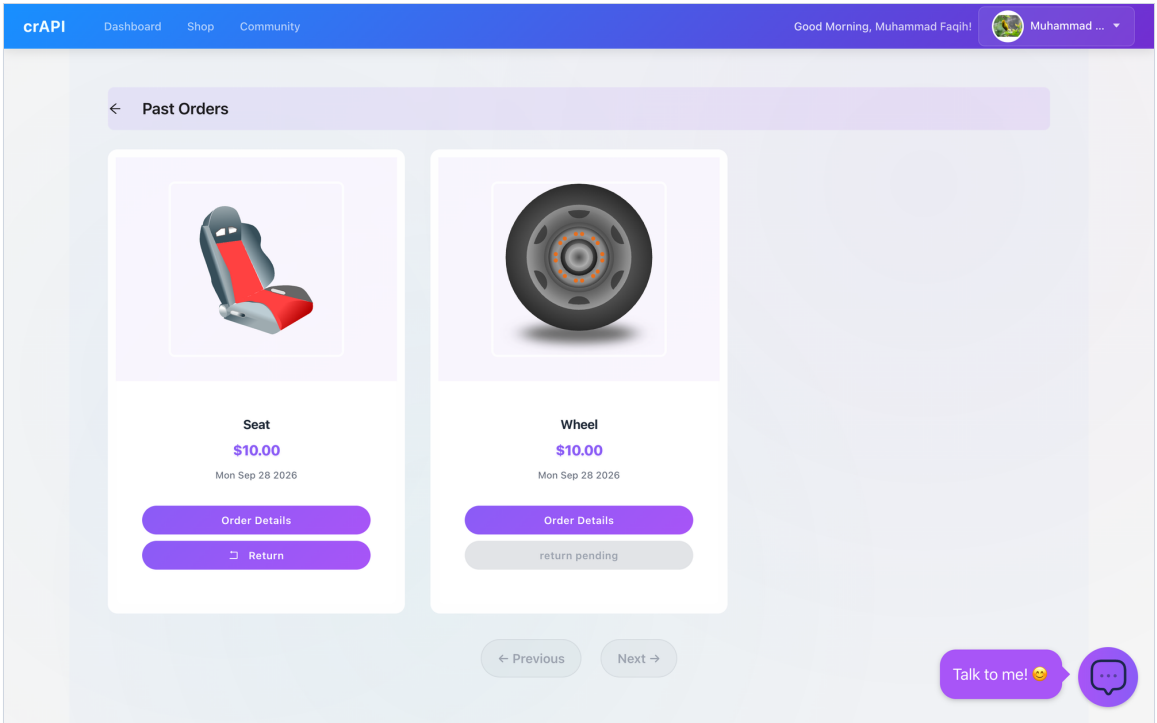


Figure 39 - Broken Object Level Authorization (IDOR) on Order Return (API1) - http://127.0.0.1:8888 (Grey-box) - crAPI Past Orders page with the 'Return' action (browser)

b. The team attempted to process a return for order_id=1, which is not owned by the attacker. The server responded HTTP 403 "You are not allowed to access this resource!", confirming the control is effective.

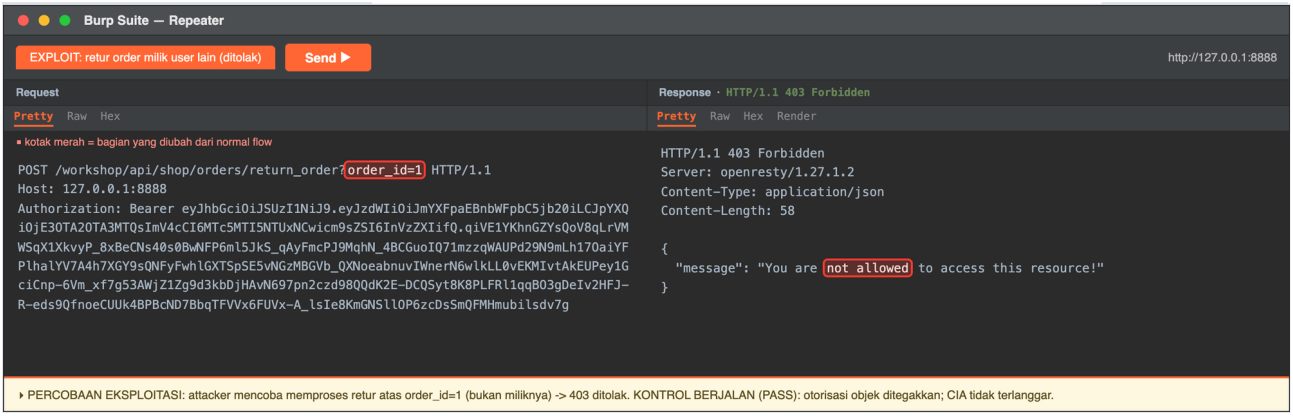


Figure 40 - Broken Object Level Authorization (IDOR) on Order Return (API1) - http://127.0.0.1:8888/workshop/api/shop/orders/return_order?order_id={id} (Return Order) (Grey-box) - Attempt blocked: return of another user's order returns 403

3.2.7 Broken Object Level Authorization (API1) – Missing Authentication Check on Mechanic Report (Unauthenticated Access)

RESULT	CVSS V3.1 BASE SCORE	EXPLOITED	WEAKNESS TESTED
PASS	—	No	CWE-306

Standard mapping	**API1:2023** Broken Object Level Authorization (OWASP API Security Top 10 2023)
Affected feature / URL	http://127.0.0.1:8888/workshop/api/mechanic/mechanic_report?report_id={id} (Mechanic Report)
Testing approach	Grey-box
Result	**PASS** — Keep authentication mandatory; also add object-ownership checks (see the related FAIL finding).

DESCRIPTION

The mechanic report endpoint was tested for unauthenticated access. The server required authentication and rejected the request without a token.

EVIDENCE

Affected URL: http://127.0.0.1:8888/workshop/api/mechanic/mechanic_report?report_id={id} (Mechanic Report)

a. The team read a mechanic report with a valid token, which returned HTTP 200 (normal flow).

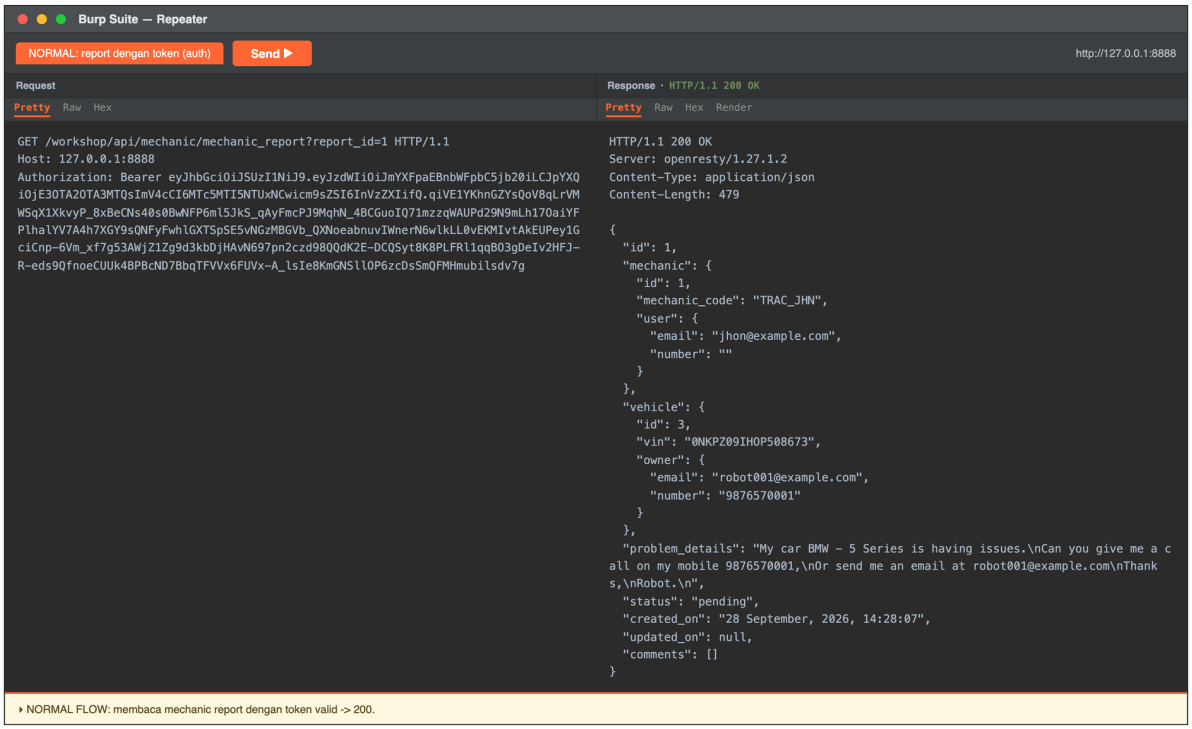


Figure 41 - Missing Authentication Check on Mechanic Report (Unauthenticated Access) (API1) - http://127.0.0.1:8888/workshop/api/mechanic/mechanic_report?report_id={id} (Mechanic Report) (Grey-box) - Normal flow: report with valid token

b. The team re-sent the request without the Authorization header. The server responded HTTP 401, confirming the authentication control is effective.

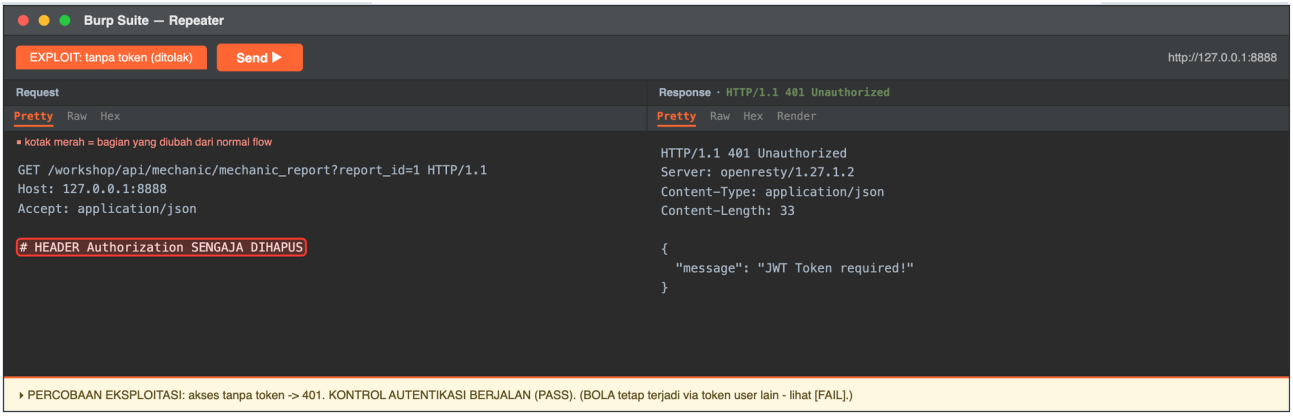


Figure 42 - Missing Authentication Check on Mechanic Report (Unauthenticated Access) (API1) - http://127.0.0.1:8888/workshop/api/mechanic/mechanic_report?report_id={id} (Mechanic Report) (Grey-box) - Attempt blocked: no token returns 401

3.2.8 Unrestricted Access to Sensitive Business Flows (API6) – Unrestricted Access to Sensitive Business Flow — Coupon Claim (Replay Protected)

RESULT	CVSS V3.1 BASE SCORE	EXPLOITED	WEAKNESS TESTED
PASS	—	No	CWE-837

Standard mapping	**API6:2023** Unrestricted Access to Sensitive Business Flows (OWASP API Security Top 10 2023)
Affected feature / URL	http://127.0.0.1:8888/workshop/api/shop/apply_coupon (Coupon Claim)
Testing approach	Grey-box
Result	**PASS** — Claiming the same coupon twice was rejected ('already claimed'); the per-user limit is enforced.

DESCRIPTION

The team tested whether the sensitive coupon-claim business flow (which increases account balance) could be abused by replaying the same coupon. The server enforced a per-user claim limit and rejected the replay.

EVIDENCE

Affected URL: http://127.0.0.1:8888/workshop/api/shop/apply_coupon (Coupon Claim)

a. Using a fresh account, the team claimed coupon TRAC075 for the first time, which succeeded and increased the account balance.

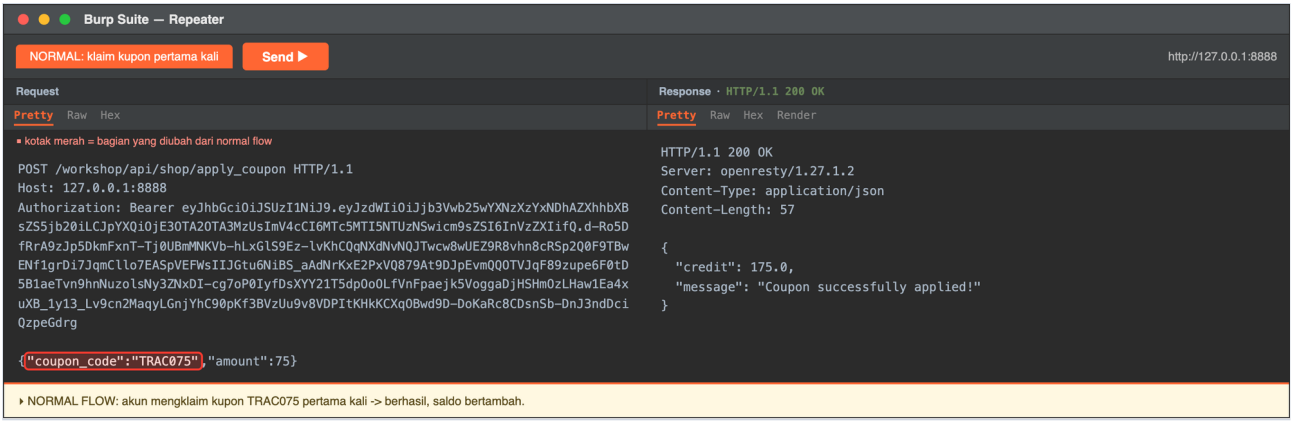


Figure 43 - Unrestricted Access to Sensitive Business Flow — Coupon Claim (Replay Protected) (API6) - http://127.0.0.1:8888/workshop/api/shop/apply_coupon (Coupon Claim) (Grey-box) - first coupon claim succeeds

b. The team attempted to claim the same coupon again; the server rejected it with 'already claimed', showing the business flow is limited per user.

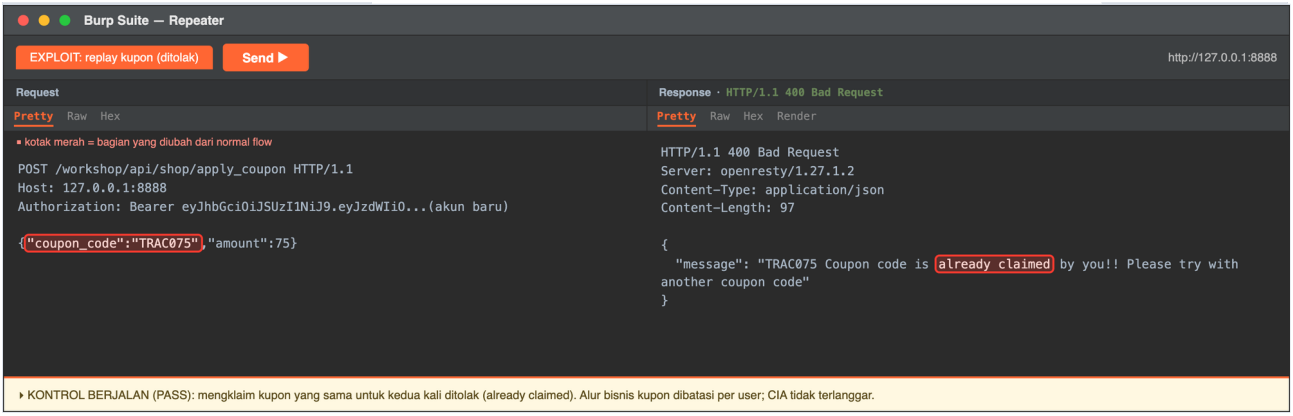


Figure 44 - Unrestricted Access to Sensitive Business Flow — Coupon Claim (Replay Protected) (API6) - http://127.0.0.1:8888/workshop/api/shop/apply_coupon (Coupon Claim) (Grey-box) - replay rejected

3.3 crAPI Community Service (http://127.0.0.1:8888/community)

3.3.1 Security Misconfiguration (API8) – NoSQL Injection on Coupon Validation

SEVERITY	CVSS V3.1 BASE SCORE	EXPLOITED	WEAKNESS
High	7.1 / 10	Yes	CWE-943

Standard mapping	**API8:2023** Security Misconfiguration (OWASP API Security Top 10 2023)
Impact rating	High
Bugcrowd VRT	P1 - Server-Side Injection (VRT 1.12)
CVSS vector	`CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:L/A:N`
Affected feature / URL	http://127.0.0.1:8888/community/api/v2/coupon/validate-coupon (Coupon Validation)
Testing approach	Grey-box

DESCRIPTION

A NoSQL injection vulnerability was found in the coupon validation endpoint. The JSON value of coupon_code is passed directly into a MongoDB query without type coercion, so supplying a query operator (e.g., {" ne":"x"} or {" gt":""}) matches any coupon without knowing its code.

IMPACT

Attackers can use this vulnerability to disclose valid coupon codes and their values and to bypass coupon validation for illegitimate discounts. During testing, the team retrieved a valid coupon without knowing its code (Confidentiality + Integrity).

RECOMMENDATION

1. Coerce coupon_code to a string type before querying; reject objects and operators.
2. Use parameterized/sanitized queries and enable input schema validation.
3. Apply the same input hardening to other MongoDB-backed queries.

REFERENCES

- VRT 1.12 - Server-Side Injection
- https://owasp.org/API-Security/editions/2023/en/0xa8-security-misconfiguration/
- https://cwe.mitre.org/data/definitions/943.html
- https://bugcrowd.com/vulnerability-rating-taxonomy

EVIDENCE

Affected URL: http://127.0.0.1:8888/community/api/v2/coupon/validate-coupon (Coupon Validation)

a. The team submitted a coupon code as a normal string it already knew (TRAC075), which returned HTTP 200 (normal usage).

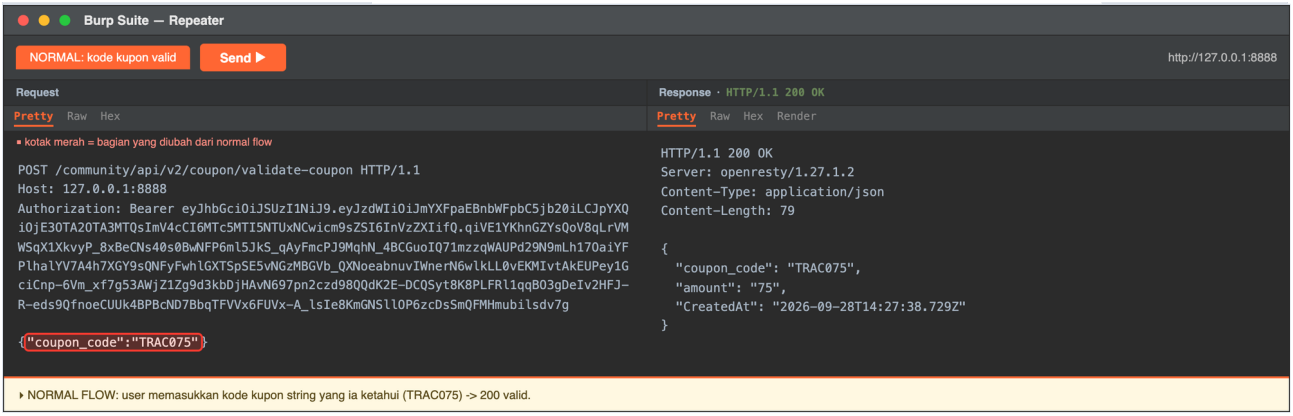


Figure 45 - NoSQL Injection on Coupon Validation (API8) - http://127.0.0.1:8888/community/api/v2/coupon/validate-coupon (Coupon Validation) (Grey-box) - Normal flow: valid string coupon code

b. The team replaced coupon_code with the NoSQL operator {"$ne":"x"}. The server returned a valid coupon without the team knowing any code.

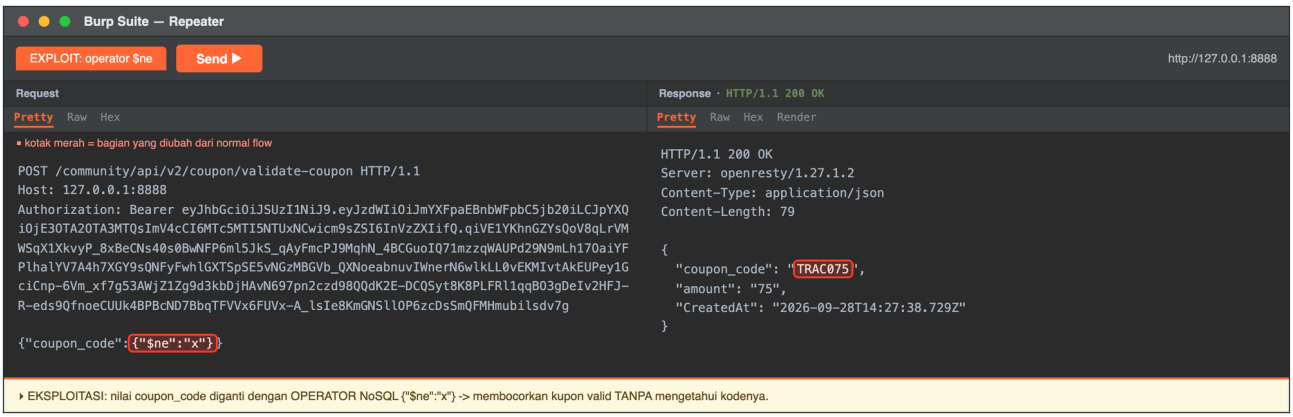


Figure 46 - NoSQL Injection on Coupon Validation (API8) - http://127.0.0.1:8888/community/api/v2/coupon/validate-coupon (Coupon Validation) (Grey-box) - Exploit: $ne operator leaks a valid coupon

c. The team used the operator {"$gt":""}, which also returned a valid coupon, confirming NoSQL injection (Confidentiality + Integrity).

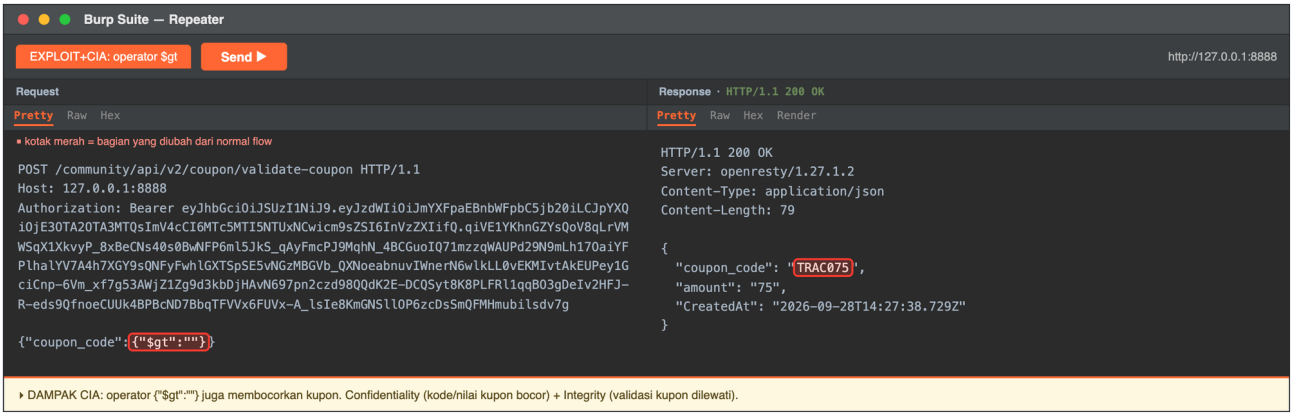


Figure 47 - NoSQL Injection on Coupon Validation (API8) - http://127.0.0.1:8888/community/api/v2/coupon/validate-coupon (Coupon Validation) (Grey-box) - Impact: $gt also leaks coupon, injection confirmed

3.3.2 Security Misconfiguration (API8) – Coupon Validation with Normal String Input

RESULT	CVSS V3.1 BASE SCORE	EXPLOITED	WEAKNESS TESTED
PASS	—	No	CWE-943

Standard mapping	**API8:2023** Security Misconfiguration (OWASP API Security Top 10 2023)
Affected feature / URL	http://127.0.0.1:8888/community/api/v2/coupon/validate-coupon (Coupon Validation)
Testing approach	Grey-box
Result	**PASS** — Enforce string type and reject operators/objects (see the related FAIL finding).

DESCRIPTION

Coupon validation was tested with ordinary string input (both a valid and an unknown code), without NoSQL operators. A valid code was accepted and an unknown code was rejected as expected.

EVIDENCE

Affected URL: http://127.0.0.1:8888/community/api/v2/coupon/validate-coupon (Coupon Validation)

a. The team submitted a valid string coupon code (TRAC075), which returned HTTP 200 (normal flow).

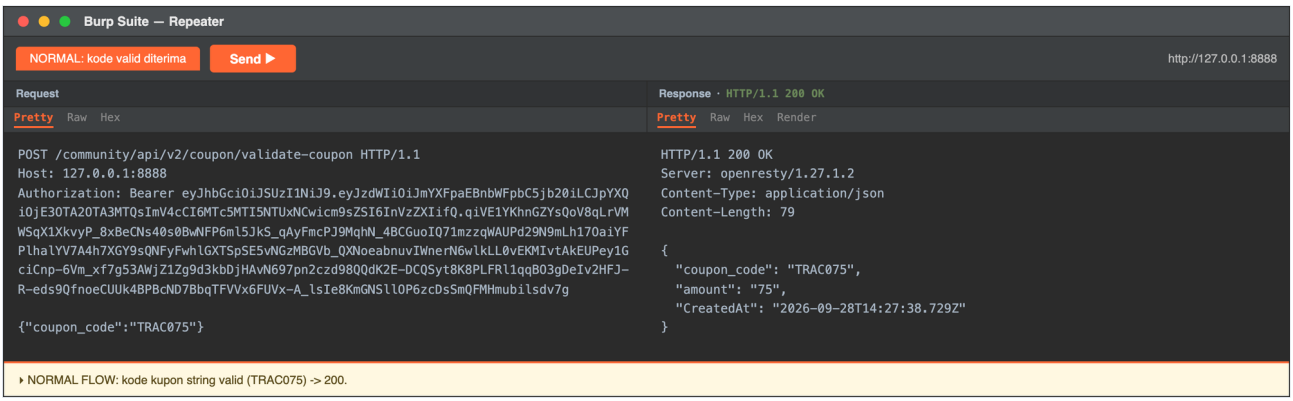


Figure 48 - Coupon Validation with Normal String Input (API8) - http://127.0.0.1:8888/community/api/v2/coupon/validate-coupon (Coupon Validation) (Grey-box) - Normal flow: valid coupon code accepted

b. The team submitted an unknown string code (INVALID_RANDOM_XYZ), which was rejected, confirming normal input validation works.

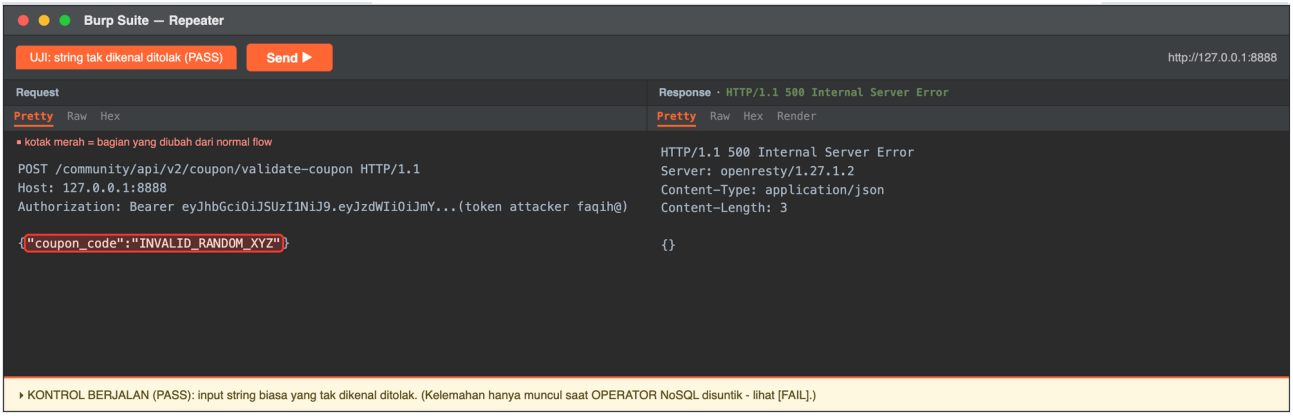


Figure 49 - Coupon Validation with Normal String Input (API8) - http://127.0.0.1:8888/community/api/v2/coupon/validate-coupon (Coupon Validation) (Grey-box) - Control: unknown string code rejected

4 Summary of Recommendations

Based on our team's examination, several vulnerabilities were found in the crAPI application that can be exploited to expose sensitive data and take over user accounts. We recommend the following actions, ordered by severity.

Broken Authentication (API2) – JWT Signature Not Verified — Authentication Bypass — Critical

1. Verify the JWT signature with the server's RS256 public key on every request.
2. Explicitly reject the alg:none algorithm and enforce algorithm allow-listing (RS256 only).
3. Use a mature JWT library with mandatory verification and validate exp, iss, and aud claims.

Server Side Request Forgery (API7) – Server-Side Request Forgery (SSRF) via Contact Mechanic → Internal MailHog → Account Takeover — Critical

1. Do not accept arbitrary URLs from the client; use an allow-list of legitimate mechanic endpoints.
2. Block internal schemes, IP ranges (loopback, RFC1918) and internal service hostnames (e.g., mailhog, crapi-identity).
3. Do not reflect raw upstream responses to the client.
4. Network-segment internal services (MailHog, actuator) so URL-consuming services cannot reach them.

Broken Function Level Authorization (API5) – Unauthenticated Access to Order Details — High

1. Require authentication on the order endpoint and verify that the order belongs to the caller.
2. Replace sequential order IDs with non-guessable UUIDs.
3. Enforce centralized authorization across all workshop endpoints.

Broken Authentication (API2) – Bypassable Brute-Force Protection on Password Reset OTP — High

1. Enforce a persistent per-account lockout that is not reset by requesting a new OTP.
2. Rate-limit and add CAPTCHA/anti-automation to the forget-password endpoint.
3. Use OTPs of at least 6 digits with short expiry and cumulative failure invalidation.

Security Misconfiguration (API8) – NoSQL Injection on Coupon Validation — High

1. Coerce coupon_code to a string type before querying; reject objects and operators.
2. Use parameterized/sanitized queries and enable input schema validation.
3. Apply the same input hardening to other MongoDB-backed queries.

Broken Function Level Authorization (API5) – Broken Function Level Authorization — Management Endpoint Exposes All Users — Medium

1. Enforce role-based authorization (verify role=admin server-side) on all management/administrative endpoints.
2. Do not rely on the absence of a UI link as access control.
3. Minimize the data returned.

Broken Object Level Authorization (API1) – Broken Object Level Authorization (IDOR) on Mechanic Reports	Medium

1. Verify the requested report_id is associated with the calling user/mechanic.
2. Use UUIDs instead of sequential identifiers.
3. Return only the minimum necessary fields.

Broken Object Level Authorization (API1) – Broken Object Level Authorization (IDOR) on Vehicle Location	Medium

1. Verify that the requested vehicle belongs to the user in the token before returning its location.
2. Implement centralized object-level authorization checks.
3. Avoid exposing owner PII in the location response.

Unrestricted Resource Consumption (API4) – Unrestricted Resource Consumption — No Login Rate Limiting	Medium

1. Enforce per-account/per-IP rate limiting and progressive account lockout.
2. Add CAPTCHA/anti-automation and anomaly detection.
3. Limit concurrent requests per client.

Unsafe Consumption of APIs (API10) – Unsafe Consumption of APIs — Unvalidated Upstream Response in Contact Mechanic	Medium

1. Validate and sanitize upstream responses (schema, content-type) before processing/reflecting.
2. Restrict upstream destinations to a trusted allow-list (see also the SSRF finding).
3. Never reflect raw upstream responses to the client.

Improper Inventory Management (API9) – Improper Inventory Management — Deprecated OTP Endpoint (v2) Still Live	Medium

1. Retire/disable deprecated API versions (v2) no longer in use.
2. Maintain an API version inventory and document live endpoints.
3. Apply uniform security controls across all live versions.

5 Conclusion

OVERALL RISK RATING

Critical

Based on our team's assessment, the crAPI application is considered **insecure**: broken authentication (JWT signature not verified) enables full account takeover, and multiple access-control, SSRF, injection, resource-consumption, inventory and unsafe-consumption issues expose customer data. Remediation of the exploited findings is required before production use.

6 Attachments

Attachment	Description