SECURITY ASSESSMENT REPORT · SAMPLE

ExampleCo Invoicing — Web Application & API Penetration Test

> ExampleCo, its application and all findings in this document are fictional and for illustration only.

Prepared by	AP Ventures Security
Assessment type	Grey-box web application & REST API test
Version	1.0 — Initial report
Classification	Confidential — client use only

Contents

1. Executive summary
2. Scope and approach
3. Findings summary
4. Detailed findings
5. Remediation roadmap
6. Severity ratings and notes

1 Executive summary

ExampleCo engaged AP Ventures Security to assess the security of its invoicing web application and supporting REST API. Testing combined AI-assisted discovery with manual testing as three user roles. All findings below were manually reproduced and validated.

OVERALL RISK

High

The most serious issues allow an unauthenticated attacker to impersonate any user (APV-01) and allow customers to access each other's invoices (APV-03). Together they put all customer data at risk. Both are straightforward to fix and should be addressed before further growth in customer numbers.

1	2	3	2	1
Critical	High	Medium	Low	Informational

Key recommendations

1. Fix token signature verification immediately and rotate signing keys (APV-01).
2. Introduce a central, mandatory ownership check for every object-level request (APV-03, APV-05).
3. Treat all user-supplied content as untrusted in internal tools as well as customer-facing pages (APV-02).
4. Add rate limiting and stronger session controls to authentication flows (APV-04, APV-07).

2 Scope and approach

In scope

- `https://app.example.com` — customer web application
- `https://api.example.com/v1` — REST API (OpenAPI spec provided)
- Roles: organisation owner, standard member, support administrator
- Staging environment with test data only

Out of scope

- Denial-of-service and load testing
- Social engineering and physical security
- Third-party payment provider infrastructure
- Mobile application binaries

Approach

Testing followed our six-phase methodology and used the OWASP Web Security Testing Guide and OWASP API Security Top 10 (2023) as references. Automated and AI-assisted tools were used for application mapping, endpoint discovery and

initial triage. Every reported finding was manually reproduced; 14 automated candidates that could not be reproduced or had no security impact were discarded.

Limitations

Testing was time-boxed and performed against a staging environment. Results reflect the application's state during testing; absence of a finding does not guarantee absence of vulnerabilities.

3 Findings summary

ID	Finding	Severity	CVSS 3.1	Status
APV-01	JWT signature not verified — authentication bypass	CRITICAL	9.1	Open
APV-02	Stored cross-site scripting in invoice notes (admin dashboard)	HIGH	8.7	Open
APV-03	Cross-tenant invoice access (IDOR)	HIGH	7.1	Open
APV-04	No rate limiting on login and password reset	MEDIUM	5.3	Open
APV-05	Mass assignment of subscription plan	MEDIUM	4.3	Open
APV-06	Excessive data exposure in team endpoint	MEDIUM	4.3	Open
APV-07	Session tokens remain valid after logout	LOW	3.7	Open
APV-08	Missing Content-Security-Policy header	LOW	3.1	Open
APV-09	Outdated front-end library (not exploitable)	INFO	–	Open

4 Detailed findings

Three findings are shown in full below. In a real report every finding is documented to this standard.

CRITICAL APV-01

JWT signature not verified — authentication bypass

CVSS 3.1	`9.1 — CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:N`
References	CWE-347 · OWASP API2:2023 Broken Authentication
Affected	`All authenticated endpoints under https://api.example.com/v1/`
Validation	✓ Manually reproduced

Description

The API accepts JSON Web Tokens whose header specifies the "none" algorithm and an empty signature. The token payload, including the user ID and role, is trusted without cryptographic verification.

Business impact

An unauthenticated attacker can forge a token for any user, including administrators, and gain full access to that account's data and functions across all tenants.

Steps to reproduce

1. Log in as any user and capture the issued access token.
2. Base64url-decode the header and change "alg" to "none".
3. Modify the payload: set "sub" to the target user ID and "role" to "admin".
4. Re-encode header and payload and remove the signature, keeping the trailing dot.
5. Send a request with the forged token — the API responds as the target administrator.

Evidence

```
GET /v1/admin/users HTTP/1.1
Host: api.example.com
Authorization: Bearer eyJhbGciOiJub25lIiwidHlwIjoiSldUIn0.eyJzdWIiOiJ1c3
JfMDAwMSIsInJvbGUiOiJhZG1pbiJ9.

HTTP/1.1 200 OK
Content-Type: application/json

{"users":[{"id":"usr_0001","email":"admin@exampleco.test","role":"admi
n"}, …]}
```

Remediation

- Verify every token signature server-side using a fixed, allow-listed algorithm (e.g. RS256 or ES256); never read the algorithm from the token header.
- Use a maintained JWT library and configure it to reject "none" and unexpected algorithms explicitly.
- Rotate signing keys and invalidate existing sessions after the fix is deployed.
- Add an automated test that sends unsigned and wrongly-signed tokens and expects HTTP 401.

HIGH APV-02

Stored cross-site scripting in invoice notes (admin dashboard)

CVSS 3.1	`8.7 — CVSS:3.1/AV:N/AC:L/PR:L/UI:R/S:C/C:H/I:H/A:N`
References	CWE-79 · OWASP Injection
Affected	`Invoice "notes" field — rendered in https://app.example.com/admin/invoices`
Validation	✓ Manually reproduced

Description

Text entered in an invoice's notes field is stored without sanitisation and rendered as HTML in the internal admin dashboard. The admin view uses a

component that inserts raw HTML.

Business impact

A customer can plant JavaScript that executes in a support administrator's browser when they view the invoice, allowing actions to be performed with administrator privileges.

Steps to reproduce

1. As a standard user, create an invoice with a harmless proof-of-concept payload in the notes field.
2. As an administrator, open the invoice in the admin dashboard.
3. Observe that the payload executes in the administrator's session (proof of concept only displayed a marker — no data was accessed).

Evidence

```
PATCH /v1/invoices/9104 HTTP/1.1
Host: api.example.com
Content-Type: application/json

{"notes":"<img src=x onerror=\"document.title='APV-XSS-POC'\">"}

→ Admin dashboard tab title changes to "APV-XSS-POC" when the invoice is viewed.
```

Remediation

- Render user content as text, not HTML. Remove the raw-HTML rendering in the admin invoice component.
- If rich text is required, sanitise on output with a maintained allow-list sanitiser (e.g. DOMPurify).
- Deploy a Content-Security-Policy that blocks inline script execution (see APV-08).

HIGH APV-03

Cross-tenant invoice access (IDOR)

CVSS 3.1 7.1 — CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:L/A:N

References	CWE-639 · OWASP API1:2023 BOLA
Affected	`GET / PATCH https://api.example.com/v1/invoices/{invoiceId}`
Validation	✓ Manually reproduced

Description

The invoices endpoint returns and updates any invoice by its numeric ID without checking that the invoice belongs to the requesting user's organisation. Invoice IDs are sequential, so valid IDs are easy to guess.

Business impact

Any authenticated user can read other customers' invoices — including names, addresses and amounts — and change their status (for example, marking an unpaid invoice as paid).

Steps to reproduce

1. Create two organisations, A and B, each with one user (accounts provided for testing).
2. As user B, note the ID of an invoice belonging to organisation A (e.g. 8812).
3. As user B, request GET /v1/invoices/8812 — the API returns organisation A's invoice.
4. As user B, send PATCH /v1/invoices/8812 with {"status":"paid"} — the change is accepted.

Evidence

```
GET /v1/invoices/8812 HTTP/1.1
Host: api.example.com
Authorization: Bearer <token for user B, organisation B>

HTTP/1.1 200 OK
Content-Type: application/json

{"id":8812,"organisationId":"org_A","customer":"Jane Doe","total":"1,24
0.00","status":"unpaid"}
```

Remediation

- Enforce ownership in the data access layer: scope every query by the caller's organisation ID (e.g. WHERE id = ? AND organisation_id = ?).
- Return 404 for objects outside the caller's organisation to avoid confirming their existence.

- Apply the same check to every endpoint that accepts an object identifier, including bulk and export endpoints.
- Consider non-sequential identifiers (UUIDs) as defence in depth — not as a replacement for authorization checks.

Other findings (summary)

MEDIUM APV-04 CVSS 5.3

No rate limiting on login and password reset

500 consecutive login attempts against one account were accepted without throttling, lockout or CAPTCHA, enabling password guessing and credential stuffing.

Fix: Apply per-account and per-IP rate limits with progressive delays; alert on spikes; support MFA.

CWE-307 · OWASP API4:2023

MEDIUM APV-05 CVSS 4.3

Mass assignment of subscription plan

Adding "plan":"enterprise" to PUT /v1/organisation was accepted, enabling paid features without payment.

Fix: Bind request bodies to explicit allow-listed DTOs; manage plan changes only through the billing service.

CWE-915 · OWASP API3:2023

MEDIUM APV-06 CVSS 4.3

Excessive data exposure in team endpoint

GET /v1/team returns every member's last login IP address and MFA status to all users, although the UI displays only names.

Fix: Return only the fields each role needs; define response schemas per role instead of serialising models directly.

CWE-213 · OWASP API3:2023

LOW APV-07 CVSS 3.7

Session tokens remain valid after logout

Access tokens continued to work for their full 24-hour lifetime after the user logged out.

Fix: Shorten access token lifetime and maintain a server-side revocation list or session store checked on each request.

CWE-613

LOW APV-08 CVSS 3.1

Missing Content-Security-Policy header

The web application sends no CSP, removing an important layer of defence against XSS (see APV-02).

Fix: Deploy a restrictive CSP (start in report-only mode), avoiding unsafe-inline for scripts.

CWE-1021 · OWASP Security Misconfiguration

INFO APV-09 CVSS —

Outdated front-end library (not exploitable)

An outdated date-picker library has a published advisory. The vulnerable function is not used, so no exploit path was found.

Fix: Upgrade during normal maintenance and add automated dependency scanning to CI.

Dependency hygiene

5 Remediation roadmap

Immediately

- APV-01 — verify JWT signatures, rotate keys
- APV-03 — enforce organisation ownership checks

Within 30 days

- APV-02 — remove raw HTML rendering
- APV-04 — rate limiting
- APV-05, APV-06 — request and response allow-lists

Next maintenance cycle

- APV-07 — session revocation
- APV-08 — Content-Security-Policy
- APV-09 — dependency upgrades

Once fixes are deployed, a retest verifies each finding and this report is reissued with an updated status (Resolved, Partially resolved or Open).

6 Severity ratings and notes

Severity	CVSS 3.1 range	Meaning
CRITICAL	9.0 - 10.0	Easily exploitable, severe impact. Fix immediately.
HIGH	7.0 - 8.9	Significant impact on data or users. Fix as a priority.
MEDIUM	4.0 - 6.9	Meaningful risk, often requiring conditions. Plan a fix.
LOW	0.1 - 3.9	Limited impact or defence-in-depth issue.
INFO	–	Observation or best practice; no direct exploit found.

CVSS base scores are adjusted in narrative where business context changes real-world risk. This sample is illustrative; it does not describe any real organisation or system.