



Offensive Security Assessment

PUBLIC PORTFOLIO EDITION

Web Application Penetration Test Report

A public, sanitized example of application security testing, business-logic analysis and client-ready remediation guidance.

PREPARED BY

Mitch Breton

ASSESSMENT TYPE

Web Application / API Penetration Test

TARGET

Redacted / Representative Environment

VERSION

Public Sanitized Edition 1.0

SANITIZED FOR PUBLIC DISTRIBUTION

Client and lab identities, IP addresses, credentials, exact payloads, account identifiers, canary values, screenshots and certification-specific metadata were removed or generalized. The assessment structure, technical themes, impact analysis and remediation guidance are preserved.

What the assessment showed

EXECUTIVE RISK SUMMARY

Prepared for
Representative Organization

1

CRITICAL

Privilege escalation and administrative access

API impersonation and unauthorized usage

2

HIGH

3

MEDIUM

Direct transaction and revenue integrity risk

Cross-site abuse of authenticated user actions

3

LOW

Testing identified a critical authorization flaw that could allow an unauthenticated user to create a privileged account. Additional weaknesses affected checkout integrity, API authentication, browser-side request protections and MFA design.

The most important theme was trust placed in client-controlled input. Privilege attributes, pricing data and authentication material were accepted with insufficient server-side enforcement, allowing multiple smaller weaknesses to combine into material business impact.

SCOPE & APPROACH

What was tested

01

Authentication & sessions

Registration, login, password changes, session handling and MFA.

02

Authorization & business logic

Role boundaries, administrative access, subscription and checkout flows.

03

API & realtime features

API-key authentication, application endpoints and WebSocket behavior.

METHODOLOGY

How testing was performed

1 Reconnaissance & mapping

Manually mapped application workflows, parameters, client-side behavior and reachable endpoints.

2 Authentication testing

Reviewed registration, login, cookies, token behavior and secondary authentication controls.

3 Authorization testing

Tampered with roles, identifiers and business parameters to validate server-side access control.

4 Business-logic testing

Manipulated checkout and subscription data to determine which values were trusted by the backend.

5 API & realtime testing

Evaluated API credential generation, endpoint authorization and WebSocket origin protections.

6 Client-side security

Reviewed CSRF exposure, clickjacking protections, input validation and error handling.

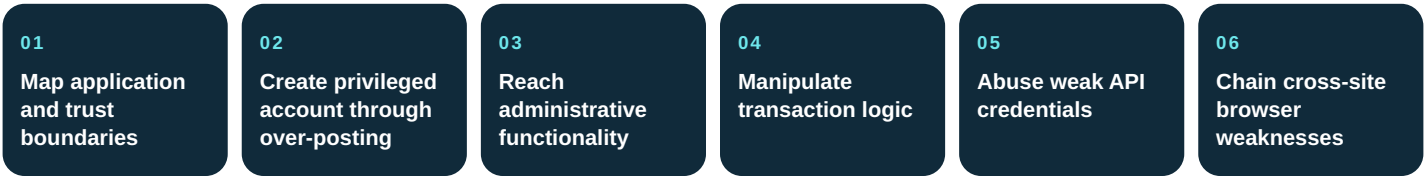
FINDINGS SUMMARY

9 documented security findings

#	Severity	Finding
01	CRITICAL	Registration mass assignment enables privilege escalation Registration / authorization
02	HIGH	Checkout logic trusted client-controlled pricing data Store / checkout
03	HIGH	Predictable API credentials enabled API impersonation API authentication
04	MEDIUM	State-changing actions lacked CSRF protection Subscription / checkout / API key generation
05	MEDIUM	Cross-site WebSocket hijacking permitted user impersonation Realtime / WebSocket
06	LOW	Malformed object input triggered unhandled server errors Authentication input handling
07	MEDIUM	MFA design lacked robust enrollment and brute-force resistance Multi-factor authentication
08	LOW	Password update workflow lacked CSRF protection Account management
09	LOW	Sensitive pages could be framed by external origins Browser hardening

ATTACK PATH

How the weaknesses combined



Technical observations and remediation

FINDING 01

CRITICAL

Registration mass assignment enables privilege escalation

Broken Access Control

Mass Assignment

CVSS 9.8

What we found

The registration workflow accepted client-controlled privilege attributes rather than whitelisting allowed fields. A newly created account could therefore be assigned an administrative role without an approval path.

Why it matters

An unauthenticated attacker could establish privileged application access and reach administrative functionality, creating a direct path to full application compromise.

Recommended action

- Whitelist server-side registration fields and reject unknown privilege-bearing attributes.
- Assign roles and plans exclusively from trusted server-side logic.
- Review other model-binding paths for over-posting or mass-assignment behavior.

Public artifact note: exact payloads, credentials, target identifiers and evidentiary screenshots have been intentionally removed or generalized.

FINDING 02

HIGH

Checkout logic trusted client-controlled pricing data

Business Logic

Integrity

CVSS 8.8

What we found

The checkout workflow accepted item prices, discounts and totals from the client and validated only that the submitted values were internally consistent. Product prices were not authoritatively recalculated from trusted server data.

Why it matters

An authenticated customer could manipulate transaction values and complete unauthorized zero-value or heavily discounted purchases, creating direct financial loss.

Recommended action

- Retrieve product pricing from a trusted data source using product identifiers.
- Calculate line totals, discounts and final totals entirely on the server.
- Alert on impossible or anomalous transactions such as zero-value orders for paid goods.

Public artifact note: exact payloads, credentials, target identifiers and evidentiary screenshots have been intentionally removed or generalized.

FINDING 03

HIGH

Predictable API credentials enabled API impersonation

Broken Authentication

API Security

CVSS 7.5

What we found

Generated API credentials contained predictable structure and insufficient entropy. Their format substantially reduced the effective search space for guessing or brute-force attacks.

Why it matters

An attacker could impersonate other API consumers, consume another user's quota and access data protected by the same authentication mechanism.

Recommended action

- Generate API keys from cryptographically secure random values with sufficient entropy.
- Avoid embedding dates, account type markers or other predictable material in key values.
- Add rate limiting, anomaly detection, key rotation and revocation.

Public artifact note: exact payloads, credentials, target identifiers and evidentiary screenshots have been intentionally removed or generalized.

FINDING 04

MEDIUM

State-changing actions lacked CSRF protection

CSRF

Session Security

CVSS 6.5

What we found

Multiple sensitive JSON POST actions relied on ambient session cookies but did not require an anti-CSRF token. Cookie policy also did not provide sufficient cross-site request protection.

Why it matters

A logged-in user could be induced to trigger unwanted state changes from a malicious origin, including subscription, checkout or credential-generation actions.

Recommended action

- Require CSRF tokens on state-changing requests.
- Use appropriate SameSite cookie settings.
- Require step-up verification for higher-risk account or financial actions.

Public artifact note: exact payloads, credentials, target identifiers and evidentiary screenshots have been intentionally removed or generalized.

FINDING 05

MEDIUM

Cross-site WebSocket hijacking permitted user impersonation

CSWSH

Origin Validation

CVSS 5.4

What we found

The realtime endpoint authenticated using existing cookies but did not sufficiently validate the request origin or require an additional anti-CSRF-style binding token.

Why it matters

A malicious website could open a WebSocket in the victim's browser and submit realtime actions in the victim's authenticated context.

Recommended action

- Validate the Origin header during the WebSocket handshake.
- Bind realtime connections to a per-session anti-CSRF token or equivalent challenge.
- Avoid relying on cookies alone for cross-origin WebSocket authentication.

Public artifact note: exact payloads, credentials, target identifiers and evidentiary screenshots have been intentionally removed or generalized.

FINDING 06

LOW

Malformed object input triggered unhandled server errors

Input Validation

Availability

What we found

Authentication-related parameters expected strings but accepted object-shaped input far enough into backend logic to generate unhandled 500 responses.

Why it matters

The behavior indicates insufficient type validation and creates an application-level availability and error-handling weakness, although no authentication bypass was demonstrated.

Recommended action

- Strictly validate parameter types and reject invalid structures with 4xx responses.
- Use safe query construction APIs and prevent direct operator injection.
- Handle malformed input without exposing server errors.

Public artifact note: exact payloads, credentials, target identifiers and evidentiary screenshots have been intentionally removed or generalized.

FINDING 07

MEDIUM

MFA design lacked robust enrollment and brute-force resistance

Authentication

Rate Limiting

CVSS 6.0

What we found

The secondary factor used a short PIN-style mechanism without a clear enrollment/change workflow or observable lockout controls for repeated failed attempts.

Why it matters

The protection provided by MFA was substantially weakened because the second factor could be subjected to repeated guessing attempts.

Recommended action

- Adopt a mature factor such as TOTP, phishing-resistant hardware tokens or equivalent.
- Provide controlled enrollment and recovery workflows.
- Apply throttling, lockout and alerting for repeated failures.

Public artifact note: exact payloads, credentials, target identifiers and evidentiary screenshots have been intentionally removed or generalized.

FINDING 08

LOW

Password update workflow lacked CSRF protection

CSRF

Defense in Depth

What we found

The password-change endpoint relied on cookies and did not enforce anti-CSRF validation. Requiring the current password reduced practical exploitability but did not eliminate the design gap.

Why it matters

The issue is primarily defense-in-depth because an attacker would still need the victim's current password, but the endpoint should not accept cross-site state-changing requests.

Recommended action

- Add standard CSRF protection to the password update action.
- Retain current-password verification as an additional safeguard.

Public artifact note: exact payloads, credentials, target identifiers and evidentiary screenshots have been intentionally removed or generalized.

Sensitive pages could be framed by external origins

Clickjacking

CSP

Headers

What we found

Sensitive application pages did not consistently return X-Frame-Options or an equivalent Content-Security-Policy frame-ancestors directive.

Why it matters

An attacker could embed authenticated pages in a malicious interface and attempt UI-redressing attacks against subscription, checkout or authentication-related actions.

Recommended action

- Set a restrictive Content-Security-Policy frame-ancestors directive globally.
- Use X-Frame-Options where legacy compatibility is required.
- Relax framing only for explicitly approved integration paths.

Public artifact note: exact payloads, credentials, target identifiers and evidentiary screenshots have been intentionally removed or generalized.

What should be dealt with first

1 Fix privilege assignment first

Remove mass assignment and ensure all role decisions are made by trusted server-side code.

2 Rebuild transaction trust boundaries

Ignore client-provided price, discount and total values; calculate every financial value server-side.

3 Replace predictable authentication material

Issue high-entropy API credentials with revocation, rotation and abuse detection.

4 Harden browser/session controls

Deploy CSRF defenses, SameSite cookies, origin validation and anti-clickjacking policy.

5 Strengthen MFA and input handling

Adopt mature MFA, throttling and strict request-type validation.

About this artifact

This is a reconstructed public-facing version of a technical assessment. It is intentionally safer to publish than a black-box overlay redaction: removed details are not present in the PDF's underlying text or images. The original confidential source reports remain unchanged.