

# Penetration Test Report: Nova Labs AI Platform

| Metadata               | Details                                                                 |
|------------------------|-------------------------------------------------------------------------|
| Target Organization    | Nova Labs Inc. (Sample Redacted Deliverable)                            |
| Primary Scope          | Web Application (app.novalabs.example), REST API (api.novalabs.example) |
| Assessment Type        | Full Black-Box Application & API Penetration Test                       |
| Testing Window         | 48-Hour Assessment Period                                               |
| Methodology            | OWASP Top 10, ASVS 4.0, WSTG v4.2                                       |
| Assessment Firm        | Zentinel Offensive Security Services (zentinel.dev)                     |
| Lead Security Engineer | Alvin Mugethi, Lead Penetration Tester                                  |
| Classification         | Confidential, Sample Demonstration Report                               |

## 1. Executive Summary

Zentinel was authorized to conduct an external black-box security assessment of Nova Labs Inc. across its production-identical staging perimeter. The primary objective was to identify security vulnerabilities, evaluate cross-tenant data isolation, stress-test authentication and authorization boundaries, and assess third-party integration integrity before public launch.

Testing was conducted using manual, offensive security techniques simulating a determined external adversary. Unlike automated scanners that only detect outdated library versions, this assessment focused on deep business logic flaws, parameter tampering, and access control boundaries.

### Summary of Findings

During the 48-hour testing window, Zentinel identified **5 verified security vulnerabilities**: \* **1 Critical Severity**: OAuth 2.0 account-linking CSRF leading to one-click account takeover. \* **2 High Severity**: Broken object-level authorization (IDOR/BOLA) in project export API, and server-side request forgery (SSRF) in webhook workers. \* **1 Medium Severity**: Unverified Stripe webhook handler allowing subscription status tampering. \* **1 Low Severity**: Exposure of sensitive environment configuration in client-side JavaScript chunks.

All findings have been validated with non-destructive proofs of concept. Exact code-level remediation guidance is provided in Section 4.

Severity Distribution:  
[CRITICAL] ■ (1) - 20%  
[HIGH] ■■ (2) - 40%  
[MEDIUM] ■ (1) - 20%  
[LOW] ■ (1) - 20%

## 2. Scope & Target Perimeter

Testing was strictly confined to the following authorized assets:

| Asset           | Target Identifier                | Description                                                |
|-----------------|----------------------------------|------------------------------------------------------------|
| Web Application | https://app.novalabs.example     | Primary customer dashboard and workspace UI                |
| REST API        | https://api.novalabs.example/v1  | Backend API handling authentication, projects, and billing |
| Webhook Worker  | https://workers.novalabs.example | Asynchronous queue worker processing outbound webhooks     |

**Excluded Assets:** Third-party OAuth identity providers (Google, GitHub), raw cloud host provider infrastructure, and live payment processing gateways.

## 3. Vulnerability Findings Matrix

| Finding ID | Vulnerability Title                                        | Severity | CVSS v3.1 | CWE     | Status   |
|------------|------------------------------------------------------------|----------|-----------|---------|----------|
| ZENT-01    | OAuth 2.0 Account Linking CSRF via Stateless Param         | CRITICAL | 8.8       | CWE-384 | Verified |
| ZENT-02    | Broken Object Level Authorization (IDOR) on Project Export | HIGH     | 7.5       | CWE-639 | Verified |
| ZENT-03    | Server-Side Request Forgery (SSRF) in Webhook Worker       | HIGH     | 7.2       | CWE-918 | Verified |
| ZENT-04    | Missing Cryptographic Signature Check on Stripe Webhook    | MEDIUM   | 5.3       | CWE-347 | Verified |
| ZENT-05    | Client-Side Exposure of Internal Staging Service Keys      | LOW      | 3.7       | CWE-200 | Verified |

## 4. Detailed Vulnerability Dossiers

### ZENT-01: OAuth 2.0 Account-Linking CSRF via Stateless Parameter

- Severity:** CRITICAL (CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:H, Score: 8.8)
- Vulnerability Class:** Session Fixation / Cross-Site Request Forgery (CWE-384)
- Affected Route:** GET /auth/callback/{provider}

#### Vulnerability Description

Nova Labs allows users to link external social and developer accounts (Google, GitHub) to their primary account. When a user initiates the connection flow, the application generates an OAuth redirect to the provider. However, the `state` parameter supplied to the OAuth provider is static and is not cryptographically tied to the user's active session cookie.

An attacker can initiate the OAuth linking flow with their own GitHub account, capture the authorization code callback URL, and entice an authenticated Nova Labs user into clicking a crafted link. Because the backend does not validate the `state` parameter against the session, the victim's workspace is silently linked to the attacker's identity.

## Business Impact

Complete account takeover. The attacker can subsequently click "Sign in with GitHub" to log directly into the victim's workspace, gaining full administrative access to all customer data and proprietary AI models.

## Step-by-Step Reproduction Proof

1. Log in to Nova Labs as User A (Attacker).
2. Initiate the GitHub connection flow at `/auth/connect/github` . Capture the resulting callback URL:  
`text GET /auth/callback/github?code=AUTH_CODE MOCK_12345&state=static_default_state`
3. Do not complete the request. Send the crafted link to User B (Victim).
4. When User B clicks the link while logged into their account, the backend accepts the request without validating session ownership: `bash curl -i -X GET "https://api.novalabs.example/v1/auth/callback/github?code=AUTH_CODE MOCK_12345&state=static_default_state" \ -H "Cookie: session_token=VICTIM_SESSION_TOKEN"`
5. Observe HTTP 200 OK. The victim's account is now bound to the attacker's GitHub profile.

## Code Remediation Guidance

Generate a cryptographically random, unpredictable `state` token on every linking request. Store the hash of this token in an encrypted, `HttpOnly` cookie or the active user session. Reject the callback if the returned state does not match.

```
// Vulnerable Implementation:
app.get('/auth/connect/github', (req, res) => {
  const url = `https://github.com/login/oauth/authorize?
client_id=${CLIENT_ID}&state=default`;
  res.redirect(url);
});

// Secure Implementation:
import crypto from 'crypto';

app.get('/auth/connect/github', (req, res) => {
  const stateToken = crypto.randomBytes(32).toString('hex');
  req.session.oauth_state = stateToken;

  const url = `https://github.com/login/oauth/authorize?
client_id=${CLIENT_ID}&state=${stateToken}`;
  res.redirect(url);
});

app.get('/auth/callback/github', (req, res) => {
  const { code, state } = req.query;
  if (!state || state !== req.session.oauth_state) {
    return res.status(403).json({ error: 'OAuth state parameter mismatch. Request aborted.'
});
  }
  delete req.session.oauth_state;
  // Proceed with token exchange...
});
```

## ZENT-02: Broken Object Level Authorization (IDOR) on Project Export

- **Severity:** HIGH (CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:N, Score: 7.5)
- **Vulnerability Class:** Broken Object Level Authorization (CWE-639)
- **Affected Route:** GET /api/v1/projects/{projectId}/export

### Vulnerability Description

Nova Labs provides a feature allowing workspace members to export project files as structured JSON packages. The export endpoint accepts a `projectId` path parameter. While the endpoint requires a valid Bearer token, it fails to verify whether the requested project belongs to the authenticated user's organization.

### Business Impact

Any authenticated user from Organization A can supply the project ID of Organization B and download proprietary training datasets, customer prompts, and model architectures.

### Step-by-Step Reproduction Proof

1. Authenticate as User A (Tenant A) and acquire a valid API token.
2. Send an export request specifying a target project ID belonging to Tenant B: `bash curl -s -X GET "https://api.novalabs.example/v1/projects/proj_tenantB_9921/export" \ -H "Authorization: Bearer USER_A_TOKEN"`
3. Observe HTTP 200 OK returning full project data, metadata, and configuration belonging to Tenant B.

## Code Remediation Guidance

Enforce tenancy validation at the database query level by always scoping lookups to the authenticated user's organization identifier.

```
# Vulnerable Implementation:
@app.get("/api/v1/projects/{project_id}/export")
async def export_project(project_id: str, user = Depends(get_current_user)):
    project = await db.projects.find_unique(where={"id": project_id})
    if not project:
        raise HTTPException(status_code=404, detail="Project not found")
    return project

# Secure Implementation:
@app.get("/api/v1/projects/{project_id}/export")
async def export_project(project_id: str, user = Depends(get_current_user)):
    project = await db.projects.find_first(
        where={
            "id": project_id,
            "organization_id": user.organization_id # Enforce tenant isolation
        }
    )
    if not project:
        raise HTTPException(status_code=404, detail="Project not found or access denied")
    return project
```

## ZENT-03: Server-Side Request Forgery (SSRF) in Webhook Worker

- **Severity:** HIGH (CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:C/C:H/I:N/A:N, Score: 7.2)
- **Vulnerability Class:** Server-Side Request Forgery (CWE-918)
- **Affected Route:** `POST /api/v1/webhooks/test`

### Vulnerability Description

The Nova Labs workspace settings allow administrators to register outbound webhook URLs for real-time notification dispatch. A "Send Test Webhook" feature dispatches an immediate HTTP POST request to user-supplied URLs.

The background worker executing this request failed to filter internal IP addresses or prevent redirect following. By supplying internal IP addresses or cloud metadata endpoints, an attacker can coerce the worker into scanning internal networks and querying cloud metadata services.

### Business Impact

Internal network reconnaissance and potential cloud instance credential extraction. In containerized cloud environments, querying `169.254.169.254` can leak temporary IAM session credentials.

### Step-by-Step Reproduction Proof

1. Authenticate to the Nova Labs portal.
2. Submit a test webhook payload pointing to an internal loopback address: 

```
bash curl -X POST "https://api.novalabs.example/v1/webhooks/test" \
-H "Authorization: Bearer USER_TOKEN" \
-H "Content-Type: application/json" \
-d '{"target_url": "http://127.0.0.1:8080/health"}'
```

3. Observe HTTP 200 response with worker execution response headers confirming internal network reachability.

### Code Remediation Guidance

Implement pre-flight DNS resolution, validate that the target IP address does not fall within private or reserved IP blocks (RFC 1918, RFC 3927), and disable automatic redirect following on outbound HTTP clients.

```
import dns from 'dns/promises';
import ipaddr from 'ipaddr.js';

async function validateWebhookUrl(userUrl: string): Promise<boolean> {
  const parsed = new URL(userUrl);
  if (parsed.protocol !== 'https:' && parsed.protocol !== 'http:') {
    return false;
  }

  const addresses = await dns.resolve4(parsed.hostname);
  for (const addr of addresses) {
    const ip = ipaddr.parse(addr);
    // Disallow private, loopback, link-local (metadata), and broadcast ranges
    if (ip.range() !== 'unicast') {
      throw new Error('Forbidden: Webhook destination resolves to private/internal network.');
    }
  }
  return true;
}
```

---

## ZENT-04: Missing Cryptographic Signature Check on Stripe Webhook

- **Severity:** MEDIUM (CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:H/A:N, Score: 5.3)
- **Vulnerability Class:** Cryptographic Signature Verification Failure (CWE-347)
- **Affected Route:** POST /api/webhooks/stripe

### Vulnerability Description

Nova Labs processes subscription upgrades via Stripe webhooks. The endpoint parses the incoming request body directly as JSON without validating the `Stripe-Signature` header against the configured webhook secret.

### Business Impact

An unauthenticated attacker can send a forged `checkout.session.completed` event containing arbitrary customer IDs, instantly upgrading any account to the Enterprise tier without financial settlement.

### Code Remediation Guidance

Enforce raw body signature verification before processing webhook events:

```
// Secure Implementation:
app.post('/api/webhooks/stripe', express.raw({ type: 'application/json' })), (req, res) => {
  const sig = req.headers['stripe-signature'];
  let event;
  try {
    event = stripe.webhooks.constructEvent(req.body, sig,
process.env.STRIPE_WEBHOOK_SECRET);
  } catch (err) {
    return res.status(400).send(`Webhook Signature Error: ${err.message}`);
  }

  // Safely handle validated event...
  res.json({ received: true });
});
```

---

## ZENT-05: Client-Side Exposure of Internal Staging Service Keys

- **Severity:** LOW (CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N, Score: 3.7)
- **Vulnerability Class:** Information Disclosure (CWE-200)
- **Affected Component:** `main-chunk.48fa.js`

### Vulnerability Description

An internal API key used for development telemetry was prefixed with a public environment variable in the frontend build configuration, causing it to be compiled into client-side JavaScript assets.

### Code Remediation Guidance

Review all frontend environment variable prefixes. Ensure backend service secrets are strictly accessible on the server runtime and never prefixed with `NEXT_PUBLIC_` or `VITE_`.

---

## 5. Security Attestation & Conclusion

Nova Labs Inc. demonstrated a proactive security posture by commissioning this third-party assessment prior to public release. The vulnerabilities identified in this report are typical of high-velocity software engineering and do not indicate systemic architectural weakness.

Following remediation of findings ZENT-01 through ZENT-04, Zentinel conducted a targeted retest and confirmed that all identified high and critical risks were effectively mitigated.

### Official Assessment Attestation

- **Target:** Nova Labs Platform (Production-Identical Staging)
- **Assessment Result:** Remediated and Verified Clean
- **Attestation Date:** September 06, 2026
- **Signed:** Alvin Mugethi, Lead Penetration Tester, Zentinel (zentinel.dev)