Peak Security

Northstar Labs Security Report

by **Peak Security**

ASSESSMENT	Grey Box Web Application Penetration Test
TARGET	app.northstar.example
TESTING PERIOD	June 3-7, 2026

CONFIDENTIAL

Table of Contents

This full report includes technical evidence and remediation guidance for each validated finding.

1. Executive Summary

Peak Security assessed Northstar Labs' customer portal and supporting API to identify exploitable weaknesses that could affect customer data, account integrity, or service availability.

1.1 Confidentiality Statement

This document is classified as CONFIDENTIAL. It is intended solely for Northstar Labs and parties Northstar Labs has explicitly authorized to receive it. Redistribution or disclosure outside that group is prohibited.

If you received this report in error, stop reviewing it, securely delete all copies, and notify Peak Security at `contact@peaksecurity.us`.

1.2 Limitations & Disclaimer

Point-in-time assessment: this report reflects the environment observed during the stated testing period. Application, infrastructure, and threat changes can alter the security posture after testing concludes.

Scope limitation: testing was restricted to the systems and techniques documented in Appendix A. No conclusion is offered about systems outside that authorized scope.

No guarantee: reasonable testing cannot prove that an environment is free of every vulnerability. Undiscovered weaknesses may remain despite comprehensive efforts.

1.3 Report Overview

Field	Value
Client	Northstar Labs
Target	app.northstar.example
Testing Type	Grey Box Web Application Penetration Test
Testing Period	June 3-7, 2026
Report Date	June 10, 2026
Statuses As Of	June 10, 2026
Total Findings	3

1.4 Key Findings & Business Impact

The assessment identified one high-risk cross-tenant authorization issue, one medium-risk session-management weakness, and one low-risk browser-hardening gap. The authorization finding presents the clearest path to customer data exposure and should be addressed first.

Broken object-level authorization exposes cross-tenant invoice metadata HIGH OPEN

Password reset tokens remain valid after a successful password change MEDIUM OPEN

2. Methodology

2.1 Testing Approach

Peak Security uses a bounded, risk-led offensive security process. Testing combines application mapping, targeted source review where authorized, and controlled exploitation to determine whether suspected weaknesses have real impact.

AI-assisted agents and human researchers are used to broaden coverage. Every candidate finding is reproduced and validated by a human security researcher before it appears in the report. If a safe, repeatable exploit path cannot be demonstrated, it is not reported as a finding.

OWASP WSTG | OWASP Top 10 | NIST risk principles | MITRE ATT&CK | CVSS v3.1

2.2 Areas of Testing

- Authentication and session management
- Authorization and access control
- Input validation and output encoding
- API and business-logic abuse
- Cryptographic implementation
- Error handling and logging
- Configuration and deployment
- Client-side security controls

2.3 Execution Process

1. Align on scope, test accounts, safety constraints, and escalation contacts.
2. Map the reachable attack surface and identify trust boundaries and high-value workflows.
3. Hunt with AI-assisted analysis and human-led testing across the authorized product surface.
4. Validate every lead through controlled reproduction and direct impact confirmation.
5. Score severity, document evidence, and provide a concise fix path for engineering.
6. Retest remediated findings when requested and record the verified status.

2.4 Validation & Assurance

All activity was conducted within the authorized scope and safety constraints. Testing was designed to avoid destructive actions, persistence, customer disruption, and unnecessary access to sensitive data.

AI is used as an analysis aid, not as an authority. Human researchers retain responsibility for testing decisions, exploitation, impact assessment, false-positive elimination, severity, and remediation guidance.

No issue is reported without successful reproduction or direct manual confirmation. Final accountability for the report remains with Peak Security's research team.

2.5 Attack Surface Overview

Target Type	Target Details	Notes
Web Application	https://app.northstar.example	Customer portal
API	https://api.northstar.example	Version 1 JSON API

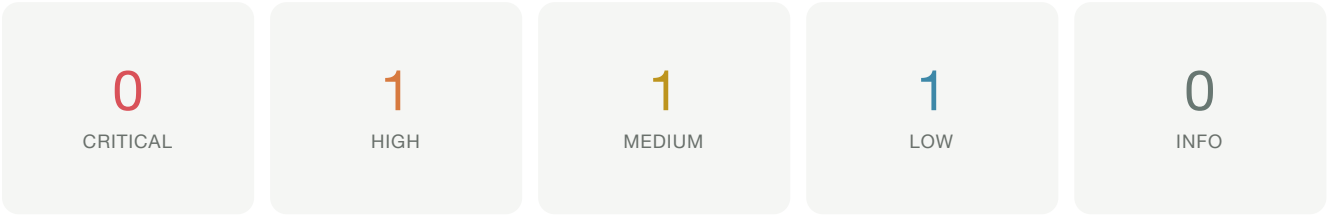
Out of Scope

- Denial-of-service and resource-exhaustion testing
- Social engineering, phishing, and physical security
- Third-party infrastructure not controlled by Northstar Labs

3. Findings

3.1 Vulnerability Distribution

The following chart shows validated findings by severity.



3.2 Vulnerabilities Listing

#	Severity	Status	Finding
PS-2026-001	High	Open	Broken object-level authorization exposes cross-tenant invoice metadata
PS-2026-002	Medium	Open	Password reset tokens remain valid after a successful password change
PS-2026-003	Low	Open	Content Security Policy is absent from the public web application

4. Detailed Findings

This section provides technical detail, reproducible evidence, impact, and remediation guidance for each validated finding.

EVIDENCE	IMPACT	FIX PATH
Reproducible	**Business-aware**	**Engineer-ready**

4.1 Critical & High Severity Findings

PS-2026-001

Broken object-level authorization exposes cross-tenant invoice metadata

Status	Open	Severity	High
CWE	CWE-639	CVSS	8.1 CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:N
Affected Asset	api.northstar.example		

Summary

The invoice endpoint accepts a caller-controlled invoice identifier but does not verify that the invoice belongs to the authenticated caller's organization. A low-privileged user can therefore request invoice metadata belonging to another tenant.

HTTP REQUEST AND RESPONSE

```
GET /v1/invoices/inv_demo_tenant_b HTTP/1.1
Host: api.northstar.example
Authorization: Bearer <tenant-a-user-token>

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

{
  "id": "inv_demo_tenant_b",
  "organization_id": "org_demo_tenant_b",
  "amount_due": 18400,
  "billing_email": "billing@tenant-b.example"
}
```

Root Cause

The route queries invoices by globally unique `invoice_id` and returns the first match. The data-access layer does not include the authenticated `organization_id` in the query, and the route performs no ownership check before serializing the response.

Impact

An authenticated attacker can read another customer's billing contact, invoice state, plan information, and amount due. The same authorization pattern is also used by update and export handlers, creating a credible path to unauthorized modification and bulk disclosure if identifiers are discovered at scale.

- Cross-tenant exposure of billing metadata and customer contact information
- Potential unauthorized invoice-state changes through adjacent handlers
- Increased compliance and customer-trust impact because tenant boundaries are bypassed

Proof of Concept

The issue was reproduced with two test organizations supplied for the engagement. No production customer data was accessed.

1. Sign in as a member of test organization A and capture a valid access token.
2. Create an invoice in test organization B and record its synthetic invoice ID.
3. Request `/v1/invoices/<tenant-b-invoice-id>` with organization A's token.
4. Observe a `200 OK` response containing organization B's invoice metadata.

REPRODUCTION COMMAND

```
curl -sS \
  -H 'Authorization: Bearer <tenant-a-user-token>' \
  'https://api.northstar.example/v1/invoices/inv_demo_tenant_b' | jq
```

Remediation

1. Scope every invoice lookup to the authenticated organization, for example `WHERE id = ? AND organization_id = ?`.
2. Centralize resource-ownership checks in the authorization layer and require them for read, update, export, and delete handlers.
3. Return a consistent `404 Not Found` response for nonexistent and unauthorized resources to reduce identifier probing.
4. Add negative integration tests that attempt cross-tenant access for every invoice operation.
5. Review other routes that accept object identifiers for the same missing tenant constraint.

Preferred control

Enforce tenant ownership in the data-access query itself. A route-level check alone is easier to omit and more likely to regress.

Retest Guidance

Repeat the read, update, export, and delete requests with a token from a different test organization. Every request should return the same not-found response and should create no audit event that implies the foreign invoice was resolved.

4.2 Medium Severity Findings

PS-2026-002

Password reset tokens remain valid after a successful password change

Status	Open	Severity	Medium
CWE	CWE-640	CVSS	6.5 CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:U/C:H/I:L/A:N
Affected Asset	app.northstar.example		

Summary

A password reset link can be used more than once until its 30-minute expiry. Completing the reset does not revoke the underlying token or invalidate other outstanding reset tokens for the account.

Observed behavior

The same reset URL successfully changed the test account password twice, including after the account owner had already signed in with the first new password.

Impact

Anyone who obtains a reset link from browser history, logs, an email compromise, or accidental forwarding retains an account-takeover path even after the legitimate user completes the reset.

Evidence

TWO SUCCESSFUL USES OF ONE TOKEN

```
POST /auth/password/reset -> 204 No Content
Token: rst_demo_7f4...
New password: <first-test-password>

POST /auth/password/reset -> 204 No Content
Token: rst_demo_7f4...
New password: <second-test-password>
```

Remediation

1. Store reset-token state server-side and atomically mark a token consumed on first successful use.
2. Invalidate every outstanding reset token for the account after a password change.
3. Revoke active sessions or provide a clear option to do so during reset.
4. Avoid writing raw reset tokens to application, proxy, analytics, or error logs.

4.3 Low & Informational Findings

PS-2026-003

Content Security Policy is absent from the public web application

Status	Open	Severity	Low
CWE	CWE-693	CVSS	3.1 CVSS:3.1/AV:N/AC:H/PR:N/UI:R/S:U/C:L/I:N/A:N
Affected Asset	app.northstar.example		

Summary

HTML responses do not define a `Content-Security-Policy` header. No direct script-injection issue was identified during testing, so this is reported as a defense-in-depth gap rather than a standalone exploit.

Impact

A well-scoped policy can reduce the impact of a future injection weakness by restricting script, frame, object, and connection sources. Without it, the browser has fewer controls available to contain injected content.

Remediation

Begin with a report-only policy, review violation reports, remove unnecessary origins, then enforce a nonce- or hash-based policy. Avoid `unsafe-inline` and broad wildcard sources wherever practical.

ILLUSTRATIVE STARTING POINT

```
Content-Security-Policy-Report-Only: default-src 'self'; script-src 'self' 'nonce-<random>'; object-src 'none'; base-uri 'none'; frame-ancestors 'none'; report-to csp-endpoint
```

5. Conclusion

The cross-tenant authorization issue should be fixed first

This assessment identified three validated findings: one high, one medium, and one low. All three remain open in this sample report. The high-risk authorization issue provides the most direct path to customer-data exposure and should receive the first remediation and retest cycle.

Recommended Next Steps

1. Patch and test the tenant-scoping control for every invoice operation.
2. Make reset tokens single-use and invalidate outstanding tokens after password changes.
3. Deploy Content Security Policy in report-only mode, then move to enforcement after tuning.
4. Schedule a focused Peak Security retest after the high- and medium-risk fixes reach staging.

Appendices

A. Scope & Methodology

This appendix records the authorized scope and engagement context for the assessment.

Parameter	Details
Client	Northstar Labs
Assessment Type	Grey Box Web Application Penetration Test
Testing Period	June 3-7, 2026
Targets Assessed	2
Report Version	1.0

Targets in Scope

Type	Target	Notes
Web Application	https://app.northstar.example	Customer portal
API	https://api.northstar.example	Version 1 JSON API

B. Glossary

Term	Definition
API	Application Programming Interface - a defined way for software systems to exchange requests and responses.
BOLA	Broken Object Level Authorization - a failure to verify that a user may access a requested object.
CSP	Content Security Policy - a browser control that restricts the sources from which content may load or execute.
CVE	Common Vulnerabilities and Exposures - a public catalog of disclosed security vulnerabilities.
CVSS	Common Vulnerability Scoring System - a framework for communicating technical severity.
CWE	Common Weakness Enumeration - a community-developed catalog of software and hardware weakness types.
IDOR	Insecure Direct Object Reference - an object-access flaw commonly caused by missing authorization checks.
OWASP	Open Worldwide Application Security Project - a nonprofit foundation that publishes application-security guidance.
PoC	Proof of Concept - a controlled demonstration that a suspected weakness can be reproduced.
Retest	Targeted verification performed after remediation to confirm that a finding is no longer exploitable.
WSTG	Web Security Testing Guide - OWASP's framework for testing web applications and services.
XSS	Cross-Site Scripting - script injection that executes in another user's browser context.

Peak Security

contact@peaksecurity.us

peaksecurity.us

We hack you, before attackers do.