KITNEURIX

OFFENSIVE INTELLIGENCE. EVOLVED.

Security Verification Report

14-Day Full-Scope Proof of Concept (PoC) — Multi-Tenant API & Business-Logic Assessment

CLIENT / ORGANIZATION	Acme Cloud SaaS (Sample Mockup)
AUTHORIZED TARGET SCOPE	staging-api.acme-cloud.example (/v1/*, /v2/*, /graphql)
ASSESSMENT DATE & WINDOW	10 Nov 2026 – 24 Nov 2026 (14-Day Continuous PoC)
SERVICE OFFERED	KITNEURIX 14-Day Full-Scope Proof of Concept ($0 Evaluation)
ISOLATED TEST PRINCIPALS	unauthenticated · tenant_alpha (owner) · tenant_beta (cross-tenant)
PREPARED BY	KITNEURIX Autonomous & Adversarial Verification Engine

CONFIDENTIALITY & AUTHORIZATION NOTICE: This deliverable is classified **TLP:AMBER** for internal engineering, security, and executive review by the authorized client organization. All testing operated strictly within the written scope allowlist under governed rate limits. Session tokens and secrets are masked (`[REDACTED]`). Note: Client name, domain, and findings in this public sample are anonymized mockup data illustrating KITNEURIX PoC deliverable quality.

01 Executive Summary

AUTONOMOUS POC TELEMETRY

1.1 Assessment Overview & Scope

Between 10 November and 24 November 2026, Kitneurix executed a 14-day full-scope Proof of Concept security verification against Acme Cloud SaaS (`staging-api.acme-cloud.example`). The autonomous verification engine evaluated authorization, privilege escalation, and business-logic workflows across live endpoints using three isolated principals: unauthenticated, Tenant Alpha (organization owner), and Tenant Beta (cross-tenant member).

Across 148 endpoints spanning five API domains, Kitneurix generated and replayed 44 targeted security hypotheses. Every candidate anomaly underwent deterministic multi-persona differential verification, automatically refuting 41 benign anomalies and delivering 3 confirmed exploitable vulnerabilities with zero false positives.

01 · SURFACE MAPPED	02 · HYPOTHESES RUN	03 · AUTO-REFUTED	04 · PROVEN EXPLOITS
148	**44**	**41**	**03** VERIFIED
Scoped API Routes	Stateful & Authz Checks	Benign Noise Filtered	Zero False Positives

1.2 Risk Posture & Key Conclusions

> **Overall Risk Rating — Elevated:** One Critical cross-tenant data exposure (`KNX-F-001`, CVSS 9.1) and one High vertical privilege escalation (`KNX-F-002`, CVSS 8.1) were deterministically proven across authenticated tenant boundaries. Engineering has already patched and verified `KNX-F-001` via regression replay; remediation remains open for `KNX-F-002` and `KNX-F-003`.

All three verified findings trace to missing server-side enforcement at trust boundaries: an unscoped global database lookup on workspace exports, client-supplied role acceptance during organization invite redemption, and a non-atomic promo credit redemption check. Addressing these controls systematically eliminates the highest-impact tenant isolation and billing risks across the platform.

1.3 Summary of Findings

ID	VULNERABILITY & ENDPOINT	SEVERITY	STATUS
KNX-F-001	Cross-Tenant Workspace Export Access `GET /v2/workspaces/ws_9481/exports/exp_204`	■ CRITICAL · 9.1	✓ VERIFIED FIXED
KNX-F-002	Client-Supplied Role Override on Invite Redemption `POST /v2/organizations/org_alpha/invites/accept`	▲ HIGH · 8.1	○ OPEN (< 7D)
KNX-F-003	Non-Atomic Promo Credit Redemption Race Condition `POST /v2/billing/credits/redeem`	◆ MEDIUM · 6.5	○ OPEN (< 14D)

02 Adversarial Verification Findings

DETERMINISTIC HTTP PROOF

2.1 Cross-Tenant Workspace Export Access

■ CRITICAL · CVSS 9.1 ✓ VERIFIED FIXED

FINDING ID	**KNX-F-001**	STANDARD	OWASP API1:2023 · CWE-639
ENDPOINT	`GET /v2/workspaces/ws_9481/exports/exp_204`	TARGET SLA	Immediate (< 48 hours)
COMPONENT	`src/routes/workspaces/exports.ts`	CVSS VECTOR	`AV:N/AC:L/PR:L/UI:N/S:C/C:H/I:L/A:L`

Overview & Business Impact

An authenticated user in an unrelated organization (Tenant Beta) can retrieve pre-signed workspace export URLs belonging to Tenant Alpha simply by supplying a target workspace and export identifier. This breaks strict multi-tenant data isolation and exposes customer invoices, billing records, and audit logs across organization boundaries, creating direct SOC 2 CC6.1, GDPR Art. 32, and LGPD compliance exposure.

Technical Root Cause & Reproduction

The export controller queries `db.exports.findUnique({ where: { id: exportId } })` using only the primary export key and never verifies that the parent workspace belongs to `req.auth.orgId`.

01 Authenticate as a standard user in Tenant Beta (`org_beta`) and obtain a valid bearer JWT.

02 Replay `GET /v2/workspaces/ws_9481/exports/exp_204` referencing Tenant Alpha's workspace and export IDs.

03 Observe HTTP `200 OK` returning `org_alpha` metadata and a valid pre-signed export URL.

Proof of Concept

Replaying the export lookup with Tenant Beta's session returns identical `org_alpha` payload data to the legitimate owner:

□ REPLAY TRACE // DETERMINISTIC MULTI-PERSONA PROOF CONFIDENCE: 1.00 · VERIFIED

```
# Request (Authenticated as Tenant Beta)
curl -sS -i "https://staging-api.acme-cloud.example/v2/workspaces/ws_9481/exports/exp_204" \
  -H "Authorization: Bearer [REDACTED_TENANT_BETA_JWT]" \
  -H "Accept: application/json"

# Observed Response (Tenant Beta) → HTTP 200 OK [Unauthorized Cross-Tenant Leak]
{"exportId":"exp_204","orgId":"org_alpha","signedExportUrl":"[REDACTED]"}

# Baseline Comparison: Unauthenticated → HTTP 401 | Owner (Tenant Alpha) → HTTP 200 OK
```

Remediation

Scope every export lookup to both `workspaceId` and the caller's authenticated `orgId`. **Retest Verified (22 Nov 2026):** Following deployment of the tenant-scoped ORM filter, Tenant Beta now receives HTTP `404`.

□ SYNTHESIZED REMEDIATION DIFF // SRC/ROUTES/WORKSPACES/EXPORTS.TS CI REGRESSION READY

```
- const exportRecord = await db.exports.findUnique({ where: { id: exportId } });
+ const exportRecord = await db.exports.findFirst({
+   where: { id: exportId, workspaceId, workspace: { orgId: req.auth.orgId } },
+ });
+ if (!exportRecord) return res.status(404).json({ error: "not_found" });
```

2.2 Client-Supplied Role Override on Invite Redemption

▲ HIGH · CVSS 8.1 ○ OPEN

FINDING ID	KNX-F-002	STANDARD	OWASP API3:2023 · CWE-269
ENDPOINT	POST /v2/organizations/org_alpha/invites/accept	TARGET SLA	High (< 7 days)
COMPONENT	src/routes/organizations/invites.ts	CVSS VECTOR	AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:N

Overview & Business Impact

A low-privilege user invited to an organization with a read-only `viewer` role can supply an arbitrary `role` property during invitation redemption and self-promote to `billing_admin`. This enables vertical privilege escalation inside customer organizations, exposing payment methods, invoices, and API key management to unauthorized collaborators.

Technical Root Cause & Reproduction

The invite handler destructures `role` directly from `req.body` and falls back to `invite.defaultRole` only when undefined, trusting client-supplied privilege attributes at membership creation.

01 Obtain a valid standard `viewer` invitation token for `org_alpha`.

02 Submit `POST /v2/organizations/org_alpha/invites/accept` with `{"inviteToken":"...","role":"billing_admin"}`.

03 Confirm the elevated `billing_admin` role assignment via `GET /v2/me`.

Proof of Concept

Adding the unvalidated `role` field to the request body overrides the server-assigned `viewer` role:

REPLAY TRACE // DETERMINISTIC MULTI-PERSONA PROOF — CONFIDENCE: 1.00 · VERIFIED

```
# Request (Tampered Invite Acceptance)
curl -sS -i -X POST "https://staging-api.acme-cloud.example/v2/organizations/org_alpha/invites/accept"
\
  -H "Authorization: Bearer [REDACTED_INVITED_USER_JWT]" \
  -H "Content-Type: application/json" \
  -d '{"inviteToken":"[REDACTED_TOKEN]","role":"billing_admin"}'

# Observed Response (Tampered) → HTTP 200 OK [Privilege Escalation Confirmed]
{"orgId":"org_alpha","assignedRole":"billing_admin"}

# Baseline Comparison: Unauthenticated → HTTP 401 | Control (No role field) → assignedRole: "viewer"
```

Remediation

Enforce strict DTO schema validation that strips unknown properties and bind membership roles exclusively from the server-side invitation record. **Status:** Open; awaiting patch deployment for automated regression replay.

SYNTHESIZED REMEDIATION DIFF // SRC/ROUTES/ORGANIZATIONS/INVITES.TS — CI REGRESSION READY

```
- const { inviteToken, role } = req.body;
- await memberships.create({ userId: req.auth.userId, orgId, role: role ?? invite.defaultRole });
+ const { inviteToken } = acceptInviteSchema.parse(req.body); // strict schema strips extra keys
+ await memberships.create({ userId: req.auth.userId, orgId, role: invite.defaultRole });
```

2.3 Non-Atomic Promo Credit Redemption Race Condition

◆ MEDIUM · CVSS 6.5 ○ OPEN

FINDING ID	**KNX-F-003**	STANDARD	OWASP API6:2023 · CWE-362
ENDPOINT	`POST /v2/billing/credits/redeem`	TARGET SLA	Medium (< 14 days)
COMPONENT	`src/services/billing/credits.ts`	CVSS VECTOR	`AV:N/AC:H/PR:L/UI:N/S:U/C:N/I:H/A:N`

Overview & Business Impact

Single-use promotional credit codes can be redeemed multiple times by dispatching synchronized parallel HTTP/2 requests, causing direct financial leakage through repeated account credit inflation.

Technical Root Cause & Reproduction

The billing service exhibits a time-of-check to time-of-use (TOCTOU) race window: it reads `redeemedByOrgId` in one query and writes the redemption update in a separate non-transactional query, allowing concurrent requests to pass validation before state is persisted.

01 Hold an authenticated Tenant Alpha session and an unused single-use promo code (`$50` value).

02 Dispatch 5 synchronized `POST /v2/billing/credits/redeem` requests concurrently over HTTP/2.

03 Observe 3 of 5 requests returning HTTP `200 OK` and crediting `$150` instead of `$50`.

Proof of Concept

Sequential replay is properly blocked with HTTP `409 Conflict`, whereas a 5-request concurrent burst succeeds 3 times:

REPLAY TRACE // CONCURRENT BURST TELEMETRY CONFIDENCE: 1.00 · VERIFIED

```
# Request (Synchronized 5x Concurrent Burst as Tenant Alpha)
curl -sS -i -X POST "https://staging-api.acme-cloud.example/v2/billing/credits/redeem" \
  -H "Authorization: Bearer [REDACTED_TENANT_ALPHA_JWT]" \
  -H "Content-Type: application/json" \
  -d '{"promoCode":"LAUNCH-TEST-CREDITS"}'

# Observed Concurrent Burst → 3/5 HTTP 200 OK [Race Condition Confirmed]
{"appliedCreditsUsd":50,"newBalanceUsd":150}

# Baseline Comparison: Sequential Replay → HTTP 409 {"error":"promo_already_redeemed"}
```

Remediation

Replace the read-then-write pattern with a single atomic conditional update inside a database transaction, paired with a unique constraint on `(promo_id, org_id)`. **Status:** Open; scheduled for billing sprint.

SYNTHESIZED REMEDIATION DIFF // SRC/SERVICES/BILLING/CREDITS.TS CI REGRESSION READY

```
- const promo = await db.promos.findUnique({ where: { code } });
- if (promo.redeemedByOrgId) throw new ConflictError();
- await db.promos.update({ where: { code }, data: { redeemedByOrgId: orgId } });
+ const updated = await db.promos.updateMany({
+   where: { code, redeemedByOrgId: null },
+   data: { redeemedByOrgId: orgId, redeemedAt: new Date() },
+ });
+ if (updated.count === 0) throw new ConflictError("promo_already_redeemed");
```

03 Remediation & Telemetry

CONTINUOUS REGRESSION CLOSURE

3.1 Prioritized Remediation Plan

PRIO	FINDING	OWNER	RECOMMENDED ENGINEERING ACTION	STATUS
P0	KNX-F-001	API Platform	Enforce `workspace.orgId = req.auth.orgId` filter in export lookup and add cross-tenant CI regression test.	✓ VERIFIED FIXED
P1	KNX-F-002	Identity / RBAC	Apply strict Zod schema stripping unknown fields; bind membership role exclusively from the server invite record.	○ IN PROGRESS
P2	KNX-F-003	Billing Service	Replace read-then-write promo check with atomic conditional `updateMany` and unique database index.	○ SCHEDULED

3.2 Strategic Security Recommendations

ARCH-01 **Enforce a Tenant-Scoped Data Access Layer:** Wrap ORM queries in a context-bound repository helper requiring `orgId` on every read and write so individual route handlers cannot accidentally omit tenant filters.

ARCH-02 **Apply Strict Input Allowlisting at API Boundaries:** Reject or strip unexpected JSON properties (such as `role`, `orgId`, `planTier`, or `status`) via strict DTO validation before requests reach service logic.

ARCH-03 **Use Atomic Conditional Writes for Financial State:** Eliminate read-then-write race windows on billing credits, seat allocations, and payouts using single-statement conditional updates inside ACID transactions.

3.3 Scope Coverage & Verification Summary

API DOMAIN	ENDPOINTS	HYPOTHESES	CONFIRMED
Auth & Session Lifecycle	19	6	0
Organizations & RBAC	31	11	1
Workspaces & Exports	42	13	1
Billing & Credits	26	8	1
GraphQL & Metadata	30	6	0
Total Verified Scope	**148**	**44**	**3**

All automated requests remained strictly inside `staging-api.acme-cloud.example` under governed rate limits and non-destructive HTTP policies. Forty-one benign anomalies (such as identical unauthenticated `200` parity on public status routes, GraphQL soft-403 responses, and unexposed UUID v4 assets) were automatically refuted prior to triage. Once fixes for `KNX-F-002` and `KNX-F-003` are merged, trigger an automated regression replay to transition all items to Verified Fixed.

KITNEURIX

OFFENSIVE INTELLIGENCE. EVOLVED.

Verification Sign-Off & Next Steps

Continuous release-delta verification engineered for zero false positives.

INCLUDED POC RETEST VERIFICATION	Once patches for KNX-F-002 and KNX-F-003 are merged to staging, trigger an automated regression replay to transition both items to VERIFIED FIXED.
CONTINUOUS COVERAGE OPTION	Sign during the 14-day PoC window to lock in flat-rate $400/month continuous verification across every release without annual lock-in.
GOVERNANCE ATTESTATION	All automated verification remained strictly inside staging-api.acme-cloud.example with destructive HTTP methods blocked and tokens masked ([REDACTED]).
CONTACT & PORTAL	https://kitneurix.com · https://kitneurix.com/sample-report/

DOCUMENT END // KNX-RPT-2026-014: Thank you for evaluating KITNEURIX. Keep this report within authorized engineering and security channels under **TLP:AMBER**.