$ intruso.ai

Penetration Test Report

Prepared for Acme Demo

Client:	Acme Demo
Report date:	2026-04-24
Test period:	2026-04-24 to 2026-04-24
Version:	1.0
Prepared by:	Intruso
Target:	https://app.acme.example

Document Control

VERSION	DATE	CHANGES
1.0	2026-04-24	Initial report

Table of Contents

1. Executive Summary
2. Scope and Methodology
3. Risk Rating Methodology
4. Findings Summary
5. Detailed Findings
6. Attack Narrative
7. Letter of Attestation
8. Appendices

1. Executive Summary

We identified 1 critical finding (full exposure of the user dataset due to missing RLS), 2 high (reflected CORS with credentials and a cross-tenant IDOR on profiles), and 3 lower-severity issues. Combining the three worst findings lets an attacker enumerate the user base anonymously and drain other tenants' profiles with a single valid account — fixing RLS on public.users and public.profiles is the immediate priority.

Overall risk assessment: CRITICAL

Key statistics

METRIC	COUNT
Critical findings	1
High findings	2
Medium findings	1
Low findings	1
Informational	1

Total unique findings	6

Top business risks identified

1. The entire user PII dataset (12,400 rows) is anonymously extractable via REST without RLS, constituting a GDPR Art. 32/33 breach with notification obligations to the regulator and data subjects.
2. Any authenticated account can enumerate profiles in other tenants — a single free signup is enough to drain the entire platform's customer base.
3. An authenticated account endpoint is read cross-origin with credentials, exposing email, plan, and internal IDs to any site visited by a user with an active session.

2. Scope and Methodology

2.1 Scope

Target systems	https://app.acme.example/* https://*.acme.example/api/*
Out-of-scope	https://app.acme.example/admin/*
Test type	SaaS platform
Perspective	Grey-box (test credentials provided)
Environment	Production

2.2 Rules of engagement

Testing window	2026-04-24, business hours (UTC)
Notification	Critical findings reported immediately to security@acme.example via the engagement Slack channel.
Data handling	No real customer data exfiltrated. Proof-of-access via count probe + 1 masked record per finding.
Network routing	Traffic originated from Intruso's authorized infrastructure. No proxy chaining.
Restrictions	Destructive opt-ins: write, rate_limit. No credential brute-force. No actions against production data outside the agent-created sentinel rows.

2.3 Methodology

Testing followed the intruso-pentest methodology v1.1.0, executed across the five phases below.

PHASE	ACTIVITIES
1. Reconnaissance	Subdomain enumeration, technology fingerprinting, OpenAPI/spec review
2. API surface mapping	Endpoint mapping, RPC discovery, storage bucket and table enumeration
3. Anonymous access testing	Probing every endpoint anonymously for data exposure and write access
4. Authentication testing	Credential strength, session handling, JWT/OAuth flow assessment

5. Authenticated exploitation	IDOR, privilege escalation, cross-tenant access, and chained exploits

2.4 Tools used

TOOL	PURPOSE
Chrome DevTools MCP	HTTP interception, evaluate_script, network and WebSocket inspection
WebFetch	NVD + CISA KEV lookups for detected technologies
Maildrop	Disposable mailbox for signup verification
Supabase REST	Tables / RPC / Storage enumeration via OpenAPI
JWT.io tooling	Token forge / replay

3. Risk Rating Methodology

Findings are rated using CVSS 4.0 (with CVSS 3.1 fallback) base scores aligned to the qualitative severity levels below.

SEVERITY	CVSS	DESCRIPTION
CRITICAL	9.0 – 10.0	Immediate exploitation risk with severe business impact. Requires emergency remediation.
HIGH	7.0 – 8.9	Significant risk of exploitation with material business impact. Remediate within 7 days.
MEDIUM	4.0 – 6.9	Moderate risk requiring attacker effort or chaining. Remediate within 30 days.
LOW	0.1 – 3.9	Minor risk with limited direct impact. Remediate within 90 days.
INFO	0.0	Observation or best-practice recommendation. No direct security impact.

4. Findings Summary

#	TITLE	SEVERITY	CVSS	CWE
1	Anonymous SELECT on the users table exposes 12,400 PII records	CRITICAL	8.7	CWE-862
2	CORS reflects arbitrary Origin with credentials allowed	HIGH	7.3	CWE-942
3	Cross-tenant IDOR in GET /rest/v1/profiles via the org_id filter	HIGH	7.7	CWE-639
4	Password policy accepts 6 characters and dictionary words	MEDIUM	5.1	CWE-521
5	Server header exposes exact nginx version	LOW	2.3	CWE-200
6	PostHog project key visible in the client bundle	INFO	0.0	CWE-200

5. Detailed Findings

5.1 — Anonymous SELECT on the users table exposes 12,400 PII records

Severity	CRITICAL
CVSS 4.0 score	8.7
CVSS vector	CVSS:4.0/AV:N/AC:L/AT:N/PR:N/UI:N/VC:H/VI:N/VA:N/SC:N/SI:N/SA:N
CWE	CWE-862
OWASP Top 10 2021	A01 — Broken Access Control
Affected asset	GET /rest/v1/users
Auth context	anonymous
Confidence	Confirmed
Status	Open

Description

The public /rest/v1/users endpoint accepts unauthenticated GET requests and returns rows containing email, full name, phone, and role. Row-Level Security policies are either missing or evaluate to TRUE for the anon role. A count probe returned a content-range header showing 12,400 total records.

Exploitability

Skill zero. Any browser or curl with the anon key (visible in the JS bundle) works. No rate limiting observed below 50 qps.

Evidence

```
# Reproduce
curl -s 'https://a1b2c3d4.supabase.co/rest/v1/users?select=id,email,full_name,phone,role&lim-
it=5' -H 'apikey: {ANON_KEY}' -H 'Authorization: Bearer {ANON_KEY}'
```

```
# Request
GET https://a1b2c3d4.supabase.co/rest/v1/users?select=id,email,full_name,phone,role&limit=5
apikey: {ANON_KEY}
Authorization: Bearer {ANON_KEY}
```

```
# Response
HTTP 200 · application/json
content-type: application/json; charset=utf-8
content-range: 0-4/12400

[{"id":"2a...","email":"j***@example.com","full_name":"J*** S***","phone":"+351 9** ***
123","role":"user"}, ...]
```

```
# The count header proves the row volume without exfiltrating the dataset.
```

Business impact

The entire user PII dataset can be exfiltrated with a single anonymous request pattern. Sufficient for credential-stuffing seed data, targeted phishing, and a regulatory breach (GDPR Art. 32/33).

Remediation

Short: Enable RLS on public.users and add a policy that limits SELECT to auth.uid() = id.

Run `ALTER TABLE public.users ENABLE ROW LEVEL SECURITY;` and create the policy:
```sql
CREATE POLICY users_self_read ON public.users
FOR SELECT USING (auth.uid() = id);
```

If staff need broader read access, create a second policy gated on a JWT role claim. Audit every other table in the `public` schema with the same pattern — a single missing policy on a table with foreign-key relationships leaks the same data.

References

- OWASP A01:2021 — Broken Access Control
- CWE-862
- https://supabase.com/docs/guides/auth/row-level-security
- https://owasp.org/Top10/A01_2021-Broken_Access_Control/

5.2 — CORS reflects arbitrary Origin with credentials allowed

Severity	HIGH
CVSS 4.0 score	7.3
CVSS vector	CVSS:4.0/AV:N/AC:L/AT:N/PR:N/UI:A/VC:H/VI:L/VA:N/SC:N/SI:N/SA:N
CWE	CWE-942
OWASP Top 10 2021	A05 — Security Misconfiguration
Affected asset	GET/POST /api/account
Auth context	authenticated_user
Confidence	Confirmed
Status	Open

Description

The /api/account endpoint reflects any Origin header into Access-Control-Allow-Origin and also emits Access-Control-Allow-Credentials: true. This lets any malicious site read authenticated user data via a browser-side fetch.

Exploitability

Requires victim interaction (visiting the attacker's page) with an active session cookie. No additional controls.

Evidence

```
# Reproduce
curl -s 'https://app.acme.example/api/account' -H 'Origin: https://evil.example.com' -i
```

```
# Request
GET https://app.acme.example/api/account
Origin: https://evil.example.com
```

```
# Response
HTTP 200
access-control-allow-origin: https://evil.example.com
access-control-allow-credentials: true

{"id":1234,"email":"user@acme.example","plan":"pro"}
```

Business impact

An authenticated user who visits an attacker-controlled page leaks their account JSON, including email, plan, and internal IDs.

Remediation

Short: Replace origin reflection with an explicit allow-list; never combine wildcards/reflection with credentials.

In the CORS middleware, replace the reflection logic with a hardcoded allow-list of trusted origins. If the application must support multiple tenant subdomains, validate the origin against a pattern anchored to your own registered domain and only echo on a match — never echo arbitrary input.

References

- OWASP A05:2021 — Security Misconfiguration
- CWE-942
- https://portswigger.net/web-security/cors
- https://owasp.org/www-community/attacks/CORS_OriginHeaderScrutiny

5.3 — Cross-tenant IDOR in GET /rest/v1/profiles via the org_id filter

Severity	HIGH
CVSS 4.0 score	7.7
CVSS vector	CVSS:4.0/AV:N/AC:L/AT:N/PR:L/UI:N/VC:H/VI:N/VA:N/SC:N/SI:N/SA:N
CWE	CWE-639
OWASP Top 10 2021	A01 — Broken Access Control
Affected asset	GET /rest/v1/profiles?org_id=eq.{other}
Auth context	cross_tenant
Confidence	Confirmed
Status	Open

Description

Authenticated users can issue cross-tenant queries against /rest/v1/profiles by overriding the org_id filter parameter. The RLS policies on the profiles table do not restrict SELECT to the requester's org_id claim.

Exploitability

Low skill. A regular user account is sufficient. Only an HTTP client and the user's own JWT are required.

Evidence

```
# Reproduce
curl -s 'https://a1b2c3d4.supabase.co/rest/v1/profiles?org_id=eq.org_2&se-
lect=id,email,full_name,role' -H 'apikey: {ANON_KEY}' -H 'Authorization: Bearer {REDACTED_TO-
KEN}'
```

```
# Request
GET https://a1b2c3d4.supabase.co/rest/v1/profiles?org_id=eq.org_2&se-
lect=id,email,full_name,role
apikey: {ANON_KEY}
Authorization: Bearer {REDACTED_TOKEN}
```

```
# Response
HTTP 200 · application/json
content-type: application/json

[{"id":"u_***","email":"j***@example.com","full_name":"J*** S***","role":"admin"}, ...]
```

Cross-tenant read confirmed by signing up two accounts in different orgs and reading each other's profiles.

Business impact

Any authenticated user can read PII (email, full name, role) from users in other tenants. With a single valid signup, the entire customer base is enumerable.

Remediation

Short: Apply RLS on profiles that filters by (auth.jwt() ->> 'org_id')::uuid = org_id.

Run:
```sql
ALTER TABLE public.profiles ENABLE ROW LEVEL SECURITY;
CREATE POLICY profiles_same_org ON public.profiles
FOR SELECT USING ((auth.jwt() ->> 'org_id')::uuid = org_id);
```

Replicate the same pattern across every table with a foreign key to org_id. Audit with a query that lists tables without policies on the SELECT command.

References

- OWASP A01:2021 — Broken Access Control
- CWE-639
- https://supabase.com/docs/guides/auth/row-level-security
- https://owasp.org/www-community/attacks/Forced_browsing

5.4 — Password policy accepts 6 characters and dictionary words

Severity	MEDIUM
CVSS 4.0 score	5.1
CVSS vector	CVSS:4.0/AV:N/AC:L/AT:P/PR:N/UI:N/VC:L/VI:L/VA:N/SC:N/SI:N/SA:N
CWE	CWE-521
OWASP Top 10 2021	A07 — Identification & Authentication
Affected asset	POST /auth/v1/signup
Auth context	anonymous
Confidence	Confirmed
Status	Open

Description

Signup accepts `123456` and `password` as valid passwords. The minimum enforced length is 6 characters.

Exploitability

Trivial as soon as one weak-password account exists — no lockout, no MFA prompt.

Evidence

```
# Reproduce
curl -s 'https://a1b2c3d4.supabase.co/auth/v1/signup' -H 'apikey: {ANON_KEY}' -H 'Content-Type: application/json' -d '{"email":"probe@maildrop.cc","password":"123456"}'
```

```
# Request
POST https://a1b2c3d4.supabase.co/auth/v1/signup
apikey: {ANON_KEY}
Content-Type: application/json

{"email":"probe@maildrop.cc","password":"123456"}
```

```
# Response
HTTP 200
content-type: application/json

{"user":{"id":"..."},"session":null}
```

Business impact

Enables credential stuffing and brute-forcing of weak passwords against the user base.

Remediation

Short: Enforce a 12-character minimum, reject dictionary words, and validate against the HIBP Pwned Passwords API.

In Supabase Auth settings (or the custom signup handler), enforce:
- length >= 12
- reject the top-10k common passwords
- check against the HIBP k-anonymity API (https://haveibeenpwned.com/Passwords)
Combine with account lockout after 5 failures per 15 minutes and offer MFA.

References

- OWASP A07:2021 — Identification & Authentication
- CWE-521
- https://pages.nist.gov/800-63-3/sp800-63b.html#sec5
- https://haveibeenpwned.com/API/v3#PwnedPasswords

5.5 — Server header exposes exact nginx version

Severity	LOW
CVSS 4.0 score	2.3
CVSS vector	CVSS:4.0/AV:N/AC:L/AT:N/PR:N/UI:N/VC:L/VI:N/VA:N/SC:N/SI:N/SA:N
CWE	CWE-200
OWASP Top 10 2021	A05 — Security Misconfiguration
Affected asset	GET /
Auth context	anonymous
Confidence	Confirmed
Status	Open

Description

The response header is `Server: nginx/1.21.3`, revealing the exact nginx version and making it easier to target known CVEs.

Exploitability

Informational; speeds up other attacks.

Evidence

```
# Reproduce
curl -s -I 'https://app.acme.example/'
```

```
# Request
HEAD https://app.acme.example/
```

```
# Response
HTTP 200
server: nginx/1.21.3
```

Business impact

Attackers can fingerprint the server and cross-reference published nginx CVEs without further reconnaissance.

Remediation

Short: Set `server_tokens off;` in nginx.conf to suppress the version.

In the http{} block of nginx.conf add `server_tokens off;`. Optionally use the ngx_headers_more module to fully replace the Server header. Confirm with `curl -I` that only `Server: nginx` (or nothing) is returned.

References

- OWASP A05:2021 — Security Misconfiguration
- CWE-200
- https://nginx.org/en/docs/http/ngx_http_core_module.html#server_tokens

5.6 — PostHog project key visible in the client bundle

Severity	INFO
CVSS 4.0 score	0.0
CVSS vector	CVSS:4.0/AV:N/AC:L/AT:N/PR:N/UI:N/VC:N/VI:N/VA:N/SC:N/SI:N/SA:N
CWE	CWE-200
OWASP Top 10 2021	A05 — Security Misconfiguration
Affected asset	GET /_next/static/chunks/main-*.js
Auth context	anonymous
Confidence	Informational
Status	Open

Description

The PostHog public project key `phc_...` appears in the client JS bundle. This is by design for client-side analytics, but is recorded for completeness.

Exploitability

Trivial but low value.

Evidence

```
# Reproduce
curl -s 'https://app.acme.example/_next/static/chunks/main-abc123.js' | grep -o
'phc_[A-Za-z0-9]*' | head -1
```

```
# Request
GET https://app.acme.example/_next/static/chunks/main-abc123.js
```

```
# Response
HTTP 200
content-type: application/javascript

...posthog.init("phc_aBc123...")...
```

Business impact

An attacker can send forged events to the customer's PostHog project, polluting analytics. No data exfiltration risk.

Remediation

Short: Accepted by design. Optional: rotate the PostHog key periodically and configure Authorized Domains.

The PostHog client-side key is intentionally public. Mitigate event spoofing by enabling 'Authorized Domains' in the PostHog project settings and rotate the key if relevant spoofing is observed.

References

- OWASP A05:2021 — Security Misconfiguration
- CWE-200
- https://posthog.com/docs/libraries/js#authorized-urls

6. Attack Narrative

Anonymous enumeration ' free signup ' cross-tenant exfiltration

An attacker combines the anonymous SELECT on public.users with the cross-tenant IDOR on /profiles to enumerate the platform's ENTIRE user base and harvest admin emails from other tenants — using only a free signup account.

> **[Step 1: Reconnaissance]**
>
> Downloading the Supabase OpenAPI exposes 47 tables and the full schema of public.users and public.profiles, including the email and role columns.

v

> **[Step 2: Anonymous PII pull] (finding-001)**
>
> GET /rest/v1/users with the anon key returns 12,400 rows; the content-range header confirms the full table volume. No RLS on the read path.

v

> **[Step 3: Free signup with weak password] (finding-003)**
>
> Signup accepts '123456'; email verification handled via Maildrop in seconds. The new account is now authenticated.

v

> **[Step 4: Cross-tenant profile drain] (finding-006)**
>
> Authenticated GET to /rest/v1/profiles?org_id=eq.org_2 returns rows from a tenant the account does NOT belong to. Confirmed enumeration of admin accounts in other orgs.

7. Letter of Attestation

This letter attests that Intruso performed an authorized penetration test against Acme Demo (https://app.acme.example) during the period 2026-04-24 to 2026-04-24, following the intruso-pentest methodology v1.1.0.

The assessment identified 6 unique findings: 1 critical, 2 high, 1 medium, 1 low, and 1 informational. Chained attack paths were validated and are documented in section 6. All test artefacts were removed at engagement close, and no production data was exfiltrated.

This letter is intended for sharing with auditors, customers, partners, and insurance carriers as evidence that an independent assessment was performed. Detailed vulnerability data is not included in this letter and is provided in the full report.

Intruso
2026-04-24

8. Appendices

Appendix A — Raw tool output

FILE	DESCRIPTION
[redacted]/recon-headers.json	Full security-header probe response
[redacted]/openapi-public.json	Supabase OpenAPI spec dump (47 tables, 12 RPCs)
[redacted]/cors-probe-matrix.txt	CORS probe matrix (3 origins × 2 methods)
[redacted]/profile-idor-evidence.json	Cross-tenant response on /profiles?org_id=eq.org_2
[redacted]/cleanup-ledger.json	Reversion + removed-account ledger

Appendix B — Testing timeline

TIME	PHASE	ACTION	RESULT
10:00	Recon	Subdomain enumeration + port scan	3 subdomains, 12 open ports
10:03	Recon	Technology fingerprinting	Next.js 15.1, Supabase 2.45, nginx 1.21.3, PostHog
10:05	Recon	Server header exposes exact nginx version	Finding 4 confirmed
10:06	Recon	Security header audit	CSP missing, HSTS no preload, X-Frame-Options DENY
10:08	Recon	Source map + bundle scan	PostHog key visible (Finding 5, INFO)
10:10	API	Download Supabase OpenAPI spec	47 tables, 12 RPC functions mapped
10:14	API	Identify dangerous RPC (seed_admin)	Exposed but requires service-role
10:17	API	Enumerate Storage buckets	2 buckets discovered: public, uploads

10:22	Anon	Anonymous SELECT probe on public.users	Finding 1 — 12,400 PII records exposed
10:28	Anon	CORS probe with arbitrary Origin	Finding 2 — reflection + Allow-Credentials true
10:33	Anon	Attempt anonymous Realtime subscription	Blocked (403)
10:38	Auth	Minimum-password probes	Finding 3 — accepts 6 chars, dictionary words
10:42	Auth	Signup via Maildrop + email verification	Account created and tokens stored in memory
10:46	Auth	JWT attacks (alg:none, HS256 forged, expired replay)	All rejected
10:55	Exploit	Cross-tenant isolation test on /profiles	Finding 6 — cross-tenant read confirmed
11:00	Exploit	Mass-assignment probe (is_admin)	Blocked by schema
11:03	Exploit	Revert sentinel writes + clean up account	Verified by re-read

Appendix C — Glossary

TERM	DEFINITION
BOLA	Broken Object Level Authorization — accessing objects belonging to other users via direct reference
CVSS	Common Vulnerability Scoring System
CWE	Common Weakness Enumeration
IDOR	Insecure Direct Object Reference — accessing resources by manipulating identifiers
MFA	Multi-Factor Authentication
OWASP	Open Web Application Security Project — non-profit publishing secure-development guidance
PII	Personally Identifiable Information
PoC	Proof of Concept — demonstration that a vulnerability is exploitable
RLS	Row-Level Security — Postgres feature restricting which rows a role can read/write
RPC	Remote Procedure Call — server-side function exposed for client invocation
SQLi	SQL Injection
SSRF	Server-Side Request Forgery
XSS	Cross-Site Scripting
WSTG	OWASP Web Security Testing Guide