



CASE STUDY: FROM API IDOR TO CLOUD EXFILTRATION, A STEP- BY-STEP PENETRATION TEST BREAKDOWN



Overview



Industry: E-commerce / Retail SaaS

Environment: Cloud-native app, APIs, object storage

Engagement: Blackbox web app penetration test + cloud configuration review

Outcome: Critical vulnerability chain discovered → Mitigated → Verified with follow-up testing

- 01 Executive Summary
- 02 Scope and Environment
- 03 From The Attacker's View
- 04 From the Defender's View
Including Sanitized Technical Findings
- 05 Remediation Roadmap
- 06 Post-Fix Verification and
Lessons Learned
- 07 The Business Summary
Including Developer and Security
Team Checklist
- 08 Take the Next Step

Executive Summary



During a routine penetration test of Velorita Retail's, a *mock e-commerce SaaS company used to simulate real-world environments*, storefront application, and public APIs, our security team uncovered a chain of vulnerabilities that allowed for:

- Enumeration of customer data via an insecure API
- Account takeover using credential stuffing
- Access to sensitive business documents stored in cloud object storage.

The root cause? A blend of common security gaps: insecure direct object references (IDOR), missing authorization checks, over-permissive cloud IAM roles, and optional multi-factor authentication (MFA) for privileged accounts. Together, these issues enabled a simulated attacker to traverse across the app's layers from unauthenticated access to cloud-level data exfiltration.

Thankfully, Velorita Retail's engineering and DevOps teams responded quickly. We partnered with them to fix the vulnerabilities, implement detection logic, and verify that remediations held up under follow-up testing.

Estimated impact: Medium-high, simulated exposure of customer PII (names, emails, shipping addresses), business documents (invoices, CSV exports), and admin credentials.

Status: Resolved, verified, and hardened.

Scope and Environment

Front End	Single-page app with API interactions
API Domain	<code>api.cumulus.example</code>
Auth	Username/password + optional TOTP
Infra	Microservices in cloud containers; object storage for documents/invoices
CI/CD	GitLab pipelines with mixed use of secrets and env vars
Testing Scope	No social engineering, non-destructive methods only

From the Attacker's View: How the Chain Was Built

Here is how the simulated attacker pivoted from public recon to business data exfiltration without ever needing a zero-day exploit.

1

Reconnaissance

The test began with endpoint enumeration. Using tools and custom scripts, we discovered an API pattern:

```
GET /api/customers/{id}
```

The `{id}` was numeric and sequential, a hallmark of potential IDOR vulnerabilities.

3

Credential Stuffing → Account Takeover

Public enumeration revealed common usernames (e.g., `john.smith`, `admin.test`). Using a basic wordlist, we successfully accessed an admin-level account with API and dashboard privileges. Critically, MFA was not enforced for this account

Note: *Credential stuffing was performed with prior approval and non-destructive methods to avoid account lockout or user impact.*

2

IDOR + Lack of Rate Limits

Testing confirmed it: querying sequential IDs returned full customer profiles, including name, email, shipping address, and even partial payment info.

No authorization checks. No rate-limiting. No anomaly detection. A script could iterate through hundreds of profiles silently.

4

Lateral Movement to Cloud Object Storage

From the compromised account, we observed that the app used an internal API to generate pre-signed URLs for accessing files in cloud object storage. However, due to misconfigured cloud roles, the service could generate links for all stored objects, not just those belonging to the authenticated user.

As a result, the attacker was able to enumerate invoice and CSV filenames, generate valid download links, and exfiltrate sensitive business data.

From the Defender's View: Detection, Response & Remediation

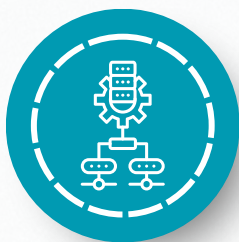


What Triggered Detection

The detection was not immediate, but it worked:

- **SIEM Alerts:** Spike in `GET /api/customers/{id}` traffic from one IP
- **Cloud Monitoring:** Egress spike from object storage
- **Auth Logs:** Multiple failed logins → one successful login for dormant admin user

Together, these signals painted a suspicious picture.



Containment Actions

Within the first 4 hours:

1. Rate-limited the offending IP via WAF
2. Revoked all active pre-signed URLs
3. Reset suspicious user credentials
4. Enabled elevated API logging
5. Launched full IR and internal stakeholder comms



Root Causes

Area	Vulnerability	Description
API	IDOR	No auth checks on resource access
Auth	Weak credentials and no MFA	Admins could log in with weak passwords
Cloud	Overbroad IAM	App role could list and access unrelated files
API Abuse	No throttling	Attackers could enumerate without blocking
Visibility	Logging gaps	No alerts on pre-signed URL issuance or downloads



Finding 1 — IDOR on `/api/customers/{id}`

Risk: High

Fix: Authorization middleware + GUIDs + WAF
rate limiting



Finding 2 — Optional MFA for privileged accounts

Risk: High

Fix: Enforce MFA, strengthen password policy, and device fingerprinting



Finding 3 — Over-permissive cloud role

Risk: High

Fix: Least-privilege IAM; prefix restrictions on object storage access



Finding 4 — No API rate-limiting or anomaly detection

Risk: Medium

Fix: Throttling, IP reputation scoring, and account
lockout logic



Finding 5 — Missing logging/alerting

Risk: Medium

Fix: Enable full access logging + alerts for excessive downloads



Remediation Roadmap



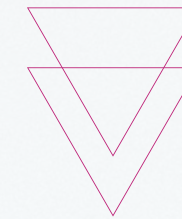
Immediate (0-48 hours)

- Enforce MFA for all privileged accounts
- Rate-limit high-risk API endpoints
- Validate pre-signed URL access at the server side



Short Term (1-2 weeks)

- Replace numeric IDs with GUIDs
- Enforce password strength and lockouts
- Add SIEM alerts for enumeration + data downloads



Medium-Term (2-8 Weeks)

- Apply least-privilege IAM to service roles
- Add automated tests to catch sensitive field leaks
- Conduct a purple-team exercise to validate coverage

Post-Fix Verification

We re-tested the entire chain. Here is what we found:

Test	Result
IDOR attempts	Blocked with 403 or 404
Credential stuffing	Triggered alerts & lockouts
Cloud role overreach	Access denied outside scoped paths
Pre-signed abuse	Logs + alerting worked as intended
Detection time	From >24h → <15 min
Containment time	From days → ~1 hour



Lessons Learned

APIs are real attack surfaces

Treat them like front doors, not side doors.

Defense-in-depth is non-negotiable.

MFA, least privilege, logging, they all matter.

Prevention starts in design.

Do not patch after launch, build securely from day one.

Telemetry is your tripwire.

Alerting to abnormal behavior shortens response windows.

Simulated attacks reveal real blind spots.

Do not wait for an incident to test your IR plan.

The Business Summary



Velorita Retail's proactive approach to security helped prevent a simulated breach from becoming a real incident.

By identifying a chain of exploitable vulnerabilities from API IDORs to over-permissive cloud roles and remediating them swiftly, the company dramatically reduced its attack surface and improved its detection and response maturity.

Developer and Security Team Checklist

- 01 Enforce auth checks on all APIs
- 02 Avoid numeric IDs, use GUIDs
- 03 Throttle and monitor API access
- 04 Require MFA for all admin users
- 05 Apply least-privilege IAM to cloud roles
- 06 Log all object storage access
- 07 Monitor pre-signed URL usage
- 08 Add CI tests for sensitive API responses
- 09 Regularly run pentests and purple-team simulations

Want this kind of review?

We simulate real-world adversaries to uncover vulnerabilities before attackers do and partner with your teams to fix them fast. Do not wait, contact us today.



Phone

(816) 299-4123



Email

sales@depthsecurity.com



Website

www.depthsecurity.com



Location

2131 Washington St, Kansas City,
MO 64108

