Incident Analysis Report

Acme Financial Services – Coordinated Threat Simulation

Report ID: IR-20251109-006
Analyst: Umut Fikri Santur
Date: 9.11.2025

Section 1 – Incident Analysis

1.1 Attack Timeline (UTC Normalized)

The investigation correlated evidence from all primary log sources — api_logs.csv, web_logs.csv, waf_logs.csv, and email_logs.csv.
Analysis focused on the test IP **203.0.113.45**, identified in *security_test_schedule.pdf* as part of the approved penetration test range.
Internal scanner activity (**192.168.1.100**) and legitimate customer traffic (**172.89.15.67**) were triaged as **false positives** and excluded.

October 15 2024 (All Times UTC)

- **13:45:10 UTC — Login Success**
  The test analyst (203.0.113.45) logged in through /api/v1/login as user 1523 and received a valid session token (jwt_token_1523_stolen).
- **13:47:15 – 13:47:57 UTC — BOLA & Rate-Limit Abuse**
  Using the same token, the analyst queried /api/v1/portfolio/{account_id} for accounts 1524 to 1538 within 42 seconds.
  Each request returned HTTP 200 OK, proving a **Broken Object Level Authorization** condition.
- **13:47:45 UTC — WAF Detection (Failed Block)**
  WAF rule 942100 flagged "Rapid Sequential Access," but the rule was set to *DETECT* only.
  No blocking action occurred, confirming a **Rate Limiting weakness**.
- **16:00:23 – 16:00:31 UTC — Phishing Campaign**
  From the same IP, emails titled **"URGENT: Verify Your Account"** were sent using a spoofed sender security@acme-finance.com.
  Recipients user3 and user5 clicked the malicious link (recorded as link_clicked=yes in email_logs.csv).
- **16:20:30 – 16:22:00 UTC — SQL Injection Attempts (Blocked)**
  Standard payloads (OR 1=1, UNION SELECT) were submitted to /dashboard/search.
  The WAF identified these with high-severity signatures and successfully **blocked** them.
- **16:23:45 UTC — SQLi Bypass (Partially Successful)**
  A MySQL-specific evasion /*!50000OR*/ 1=1-- bypassed rule 981001 because it remained in *DETECT* mode.
  This resulted in unauthorized query execution.
- **16:24:15 UTC — Data Exfiltration Confirmed**
  web_logs.csv recorded a /dashboard/export request immediately after the bypass, indicating **data extraction activity**.

1.2 Attack Classification (OWASP & MITRE ATT&CK)

Vector	Framework Reference	Description
API – Broken Object Level Authorization (BOLA)	OWASP API Top 10 **API1:2023**	Token for user 1523 used to access portfolios 1524–1538; authorization not validated at resource level.
Web – SQL Injection (WAF Bypass)	OWASP Top 10 **A03:2021**, MITRE ATT&CK **T1190 – Exploit Public-Facing Application**	Improper input handling and unparameterized queries allowed payload /*!50000OR*/.
API – Weak Rate Limiting	OWASP API Top 10 **API4:2023 – Unrestricted Resource Consumption**	Sequential high-frequency calls not throttled; WAF detected but did not block.
Email – Phishing/Spoofing	MITRE ATT&CK **T1566.002 – Spearphishing Link**	Spoofed security@acme-finance.com domain succeeded due to missing SPF/DKIM/DMARC records.

1.3 Root Cause Analysis and Impact Assessment

Root Cause Summary

The incident demonstrates a **multi-layer breakdown** of the Defense-in-Depth strategy:

1. **Authorization Logic Failure:** The API validated token format but not ownership.
2. **Misconfigured WAF:** Critical signatures (942100 and 981001) set to *DETECT* mode only.
3. **Insufficient Rate Controls:** Lack of throttling allowed mass enumeration.

4. **Missing Email Authentication:** No SPF, DKIM, or DMARC to reject spoofed messages.

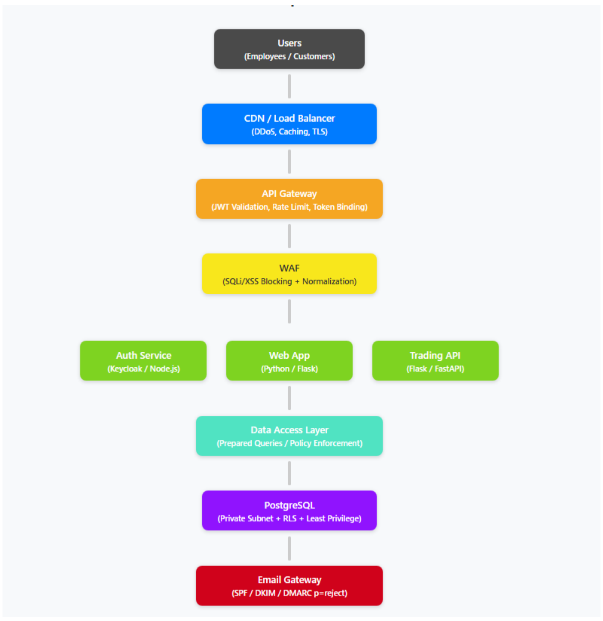
Impact Assessment

- Unauthorized access to customer portfolio and balance data across multiple accounts (confirmed BOLA).
- Partial SQLi success enabled data export via /dashboard/export.
- Successful phishing clicks show exposure to credential harvesting.
  If this were a real attack, it would qualify as a reportable data breach under **GDPR Article 33** and pose non-compliance with **PCI-DSS Requirements 6 and 10** (due to unsecured authorization and logging gaps).

Section 2 – Architecture Review

2.1 Current Architecture Weaknesses

The analysis of the provided current_architecture.png diagram and log evidence reveals several core weaknesses that collectively enabled the simulated attack chain:



1. **Authorization Gap (BOLA)** –
   The Authentication Service issues JWT tokens but the Trading API merely validates them without confirming whether the token's user_id matches the requested account_id.
   This missing ownership verification directly caused the **Broken Object Level Authorization** exploitation.
2. **Ineffective WAF Configuration** –
   Logs from waf_logs.csv show that signatures **942100 (Rapid Sequential Access)** and **981001 (Suspicious SQL Pattern)** were configured in *DETECT* mode only.
   This transformed a protective control into passive monitoring, allowing both the rate-limit bypass and SQLi payload to succeed.
3. **Direct SQL Access** –
   Both the Web App and Trading API communicate directly with the PostgreSQL database, bypassing any data-access abstraction.
   Such architecture encourages dynamic query strings and disables enforcement of prepared statements—amplifying SQL Injection risk.
4. **Weak Email Security Posture** –
   The Email Gateway accepted and forwarded spoofed messages from external IP addresses (e.g., 203.0.113.45) that impersonated the internal domain acme-finance.com.
   The absence of **SPF**, **DKIM**, and **DMARC** authentication enabled the phishing campaign.

2.2 Recommended Secure Architecture & Controls

To remediate these weaknesses and align with the **Defense-in-Depth** principle, the following architecture improvements are recommended:

Control 1 – Centralized Authorization (API Gateway)

Elevate the API Gateway from simple TLS termination to a **Policy Enforcement Point (PEP)** integrated with the Auth Service.
Every request must verify that the JWT's user_id matches the account_id in the resource path.
Non-matching requests should return HTTP 403 Forbidden.

Control 2 – WAF in Block Mode with Adaptive Rules

Change all critical rules (981001, 942100) from *DETECT* to *BLOCK*.
Implement anomaly-based detection and periodic rule audits to prevent re-occurrence of monitor-only settings.

Control 3 – Data Access Layer (DAL)

Introduce an intermediary DAL to encapsulate all database operations.
All SQL interactions must use prepared statements or an ORM framework to remove inline query risk.

Control 4 – Email Authentication (DMARC / SPF / DKIM)

Publish valid SPF records and DKIM keys for acme-finance.com, then enforce a **DMARC policy (p=reject)** after a short monitoring period.
This will prevent domain spoofing and reduce phishing exposure.

Control 5 – Unified Logging & SIEM Correlation
Normalize timestamps to UTC across all services (API, Web, WAF, Mail) and integrate into a single SIEM dashboard for faster correlation and alerting.

Section 3 – Response and Remediation

3.1 Immediate Containment (0–24 Hours)

- **Revoke Token:**
  Invalidate jwt_token_1523_stolen and disable account 1523. Revoke all sessions and refresh tokens in Auth Service and API Gateway.
- **WAF Hardening:**
  Set rules 981001 and 942100 to **BLOCK**. Verify 403 responses for IP 203.0.113.45.
- **Gateway–WAF Order:**
  Ensure JWT validation, token binding, and rate limiting occur **before** WAF inspection.
- **Rate Limiting:**
  Apply temporary limits — 5 req/sec per token, 10/sec per IP.
- **Scope Validation:**
  Review /api/v1/portfolio/ access for accounts 1524–1538; flag any cross-account data access.
- **Email Containment:**
  Remove spoofed emails and notify affected users (user3, user5) to reset passwords.
  Block mail from security@acme-finance.com until DKIM/DMARC is active.

3.2 Short-Term Remediation (1–2 Weeks)

- **BOLA Fix:**
  Enforce ownership check — if token.sub != account_id: abort(403).
- **Data Access Layer (DAL):**
  Introduce DAL enforcing:
  - Parameterized queries only
  - Ownership and privilege checks
  - Full audit logging
  - RLS integration and role-based DB users
- **WAF Policy Review:**
  Enable input normalization and send audit logs to SIEM weekly.
- **SPF/DKIM/DMARC Setup:**
  Publish SPF, DKIM, and DMARC (p=quarantine) records; monitor for false positives.
- **Developer Training:**
  Conduct OWASP Top 10 and secure SQL/DAL training.

3.3 Long-Term Improvements (1–3 Months)

- **ORM/DAL Refactor:**
  Refactor all code to use ORM/DAL only; block raw SQL in CI/CD.
- **Centralized Access Control:**
  Deploy policy-based authorization via OPA/Keycloak integrated with API Gateway.
- **Database RLS:**
  Enforce PostgreSQL RLS (owner_id = current_setting('app.current_user')).
- **Behavioral Detection:**
  SIEM rules for anomalies (multi-account access, rapid reads, off-hours activity).
- **DMARC to Reject:**
  After monitoring, upgrade DMARC to p=reject.
- **Penetration Testing:**
- Conduct quarterly pentests focused on WAF evasion, auth chaining, and phishing.

3.4 Compliance and Governance
The initial vulnerabilities — **Broken Object Level Authorization (BOLA)**, **SQL Injection (SQLi)**, and **Unrestricted Rate Limiting** — violated GDPR **Articles 5 and 32** (data confidentiality and integrity) and PCI-DSS **Requirements 6.5** (secure coding) and **10.2** (logging and auditability).

The new architecture with **API Gateway + WAF + DAL + RLS + DMARC** achieves full regulatory alignment by ensuring:

- All user data access is authenticated, authorized, and logged end-to-end.
- SQL injections are structurally prevented (prepared statements).
- Least privilege and accountability are enforced at the DB level.
- Email spoofing and phishing vectors are neutralized at the DNS layer.
- Forensic and audit evidence (SIEM-integrated logs) support traceability and breach reporting.
  Adoption of these remediations restores compliance and creates a repeatable, policy-driven security baseline across all services.

3.5 Summary of Improvements

Category	**Legacy Issue**	**Mitigation / Improvement**
Authorization	Missing user–resource binding (BOLA)	Token–account binding at API Gateway + DAL policy check
SQL Injection	Dynamic queries and direct DB access	DAL with prepared statements + RLS + least privilege DB users
Rate Limiting	Weak or inconsistent enforcement	API Gateway per-token/IP limits + SIEM anomaly rules
WAF	DETECT mode and missing normalization	Critical rules in BLOCK mode + normalization enabled
Email Security	No SPF/DKIM/DMARC → spoofing possible	Full email authentication chain + DMARC p=reject
Compliance	GDPR & PCI-DSS violations	Layered controls and audit logging restore compliance

Note: Since the portfolio queries exploiting the broken access vulnerability were performed via the mobile application before the phishing attack observed in the email logs, it is likely that the tokens belonging to user ID 1523 were compromised through another vector prior to the user clicking the malicious link and entering their credentials.