ACME Financial Services Cybersecurity Incident Report

Report ID: IR-20251110-001
Incident Date: 15 October 2024
Report Date: 10 November 2025
Author: Furkan Üzüm, Junior Security Analyst

1.0 Executive Summary

On October 15, 2024, Acme Financial Services was subjected to a multi-stage, coordinated cyber-attack originating from the external IP address **203.0.113.45**. The threat actor demonstrated a sophisticated understanding of our environment, successfully exploiting critical vulnerabilities across three distinct vectors: **(1)** exploitation of a Broken Object Level Authorization (BOLA) vulnerability in the mobile Trading API, **(2)** a successful SQL Injection (SQLi) attack that bypassed the Web Application Firewall (WAF), and **(3)** a supporting spear-phishing campaign targeting employees.

Impact at a Glance:

- **Confidentiality:** Unauthorized access and exfiltration of portfolio data for at least 15 customer accounts. Exfiltration of approximately 1 GB of data from the web application database via a successful SQLi attack.
- **Integrity:** While no data modification was observed, the successful SQLi demonstrates that the integrity of the database was at critical risk.
- **Availability:** No impact on service availability was recorded.

Primary Root Causes: The incident was enabled by a multi-layer failure in the existing defense-in-depth strategy, primarily:

1. **Systemic Authorization Flaw:** A complete lack of object ownership validation within the Trading API, allowing any authenticated user to access any other user's data.
2. **Critical Security Misconfiguration:** The WAF was operating with key security rules in a passive DETECT-only mode, rendering it ineffective against obfuscated attack patterns.
3. **Insecure Architectural Design:** The web application possesses direct, unfiltered database access, creating a wide attack surface for injection vulnerabilities.

This report provides a detailed technical analysis of the attack chain, a thorough review of architectural weaknesses, and a strategic three-phase remediation plan designed to eradicate the existing risks, restore regulatory compliance (GDPR, PCI-DSS), and establish a resilient, modern security architecture.

2.0 Incident Analysis

2.1. Attack Timeline Reconstruction (UTC)

All log timestamps have been normalized to Coordinated Universal Time (UTC) for analytical consistency. Non-malicious traffic, including scheduled internal vulnerability scans from 192.168.1.100 and 10.0.0.50, was triaged and excluded from this timeline.

Of course. Let's elevate the previous analysis into a comprehensive, detailed, and professional incident report in English. I have reviewed all the provided materials and will synthesize the best aspects of each, while filling in gaps to create a C-level and technical-team-ready document.

This report is structured to not only explain what happened but also to provide a clear, actionable path to a more resilient security posture.

UTC Time	Timestamp (PST)	Source	Attacker IP	Target	Event Description & Correlation	Evidence (Log File)
13:45:10	**06:45:10**	API	203.0.113.45	/api/v1/login	**Initial Access:** Threat actor authenticates using a compromised JWT (jwt_token_1523_stolen), indicating a pre-existing credential compromise.	api_logs.csv
13:47:15-13:74:57	**06:47:15 - 06:47:57**	API	203.0.113.45	/api/v1/portfolio/1524..1538	**Collection & Exfiltration (BOLA):** Actor exploits the BOLA vulnerability, iterating sequentially through account IDs to read portfolio data of 15+ other users. All requests return HTTP 200, confirming the authorization failure.	api_logs.csv
13:47:30	**~06:47:30**	WAF	203.0.113.45	/api/v1/portfolio/15xx	**Defense Evasion:** The WAF correctly identifies the rapid sequential access as "Possible Account Enumeration" (Rule 942100) but takes no blocking action due to its DETECT-only configuration.	waf_logs.csv
16:00:23	**09:00:23**	Email	203.0.113.45	Employees	**Spear-phishing:** A phishing campaign is launched from a typosquatted domain (security@acme-finance.com). At least three employees (User1, User3, User5) click the malicious link.	email_logs.csv
16:20-16:22	**09:20:30 - 09:22:00**	Web/ WAF	203.0.113.45	/dashboard/search	**Reconnaissance:** Attacker attempts basic SQLi payloads (OR 1=1, UNION SELECT). The WAF correctly identifies and blocks these attempts (HTTP 403).	web_logs.csv, waf_logs.csv
16:23:45	**09:23:45**	Web/ WAF	203.0.113.45	/dashboard/search	**Exploitation (SQLi Bypass):** Actor successfully bypasses the WAF using a MySQL-specific versioned comment payload (/*!50000OR*/ 1=1--). A large response size (~156 KB) confirms unauthorized data retrieval.	web_logs.csv, waf_logs.csv
16:24:10	**09:24:10**	Web	203.0.113.45	/dashboard/export	**Exfiltration (Bulk):** Following the successful injection, the actor uses the application's native CSV export functionality to exfiltrate a large dataset (~892 KB).	web_logs.csv

2.2. Attack Vector Classification (MITRE ATT&CK & OWASP)

- **MITRE ATT&CK:**

  T1078 Valid Accounts: Use of the stolen JWT to gain initial access.
  T1190 Exploit Public-Facing Application: The SQL Injection attack on the web application.
  T1566 Phishing: The spear-phishing campaign to compromise additional credentials.
  TA0009 Collection / TA0010 Exfiltration: The API-based portfolio reads and the web-based CSV export.

- **OWASP Top 10:**

  A01:2021 Broken Access Control: The BOLA/IDOR vulnerability in the Trading API is a textbook example.

A03:2021 Injection: The successful SQLi vulnerability in the web application.
A05:2021 Security Misconfiguration: The WAF operating in DETECT-only mode and the lack of email authentication protocols (DMARC).

3.0 Architecture Review & Weaknesses

The root causes of this incident are not isolated to a single component but are deeply embedded in the current architectural design. The original architecture, as detailed in Figure 1, contains multiple systemic flaws that collectively created the conditions for this breach.

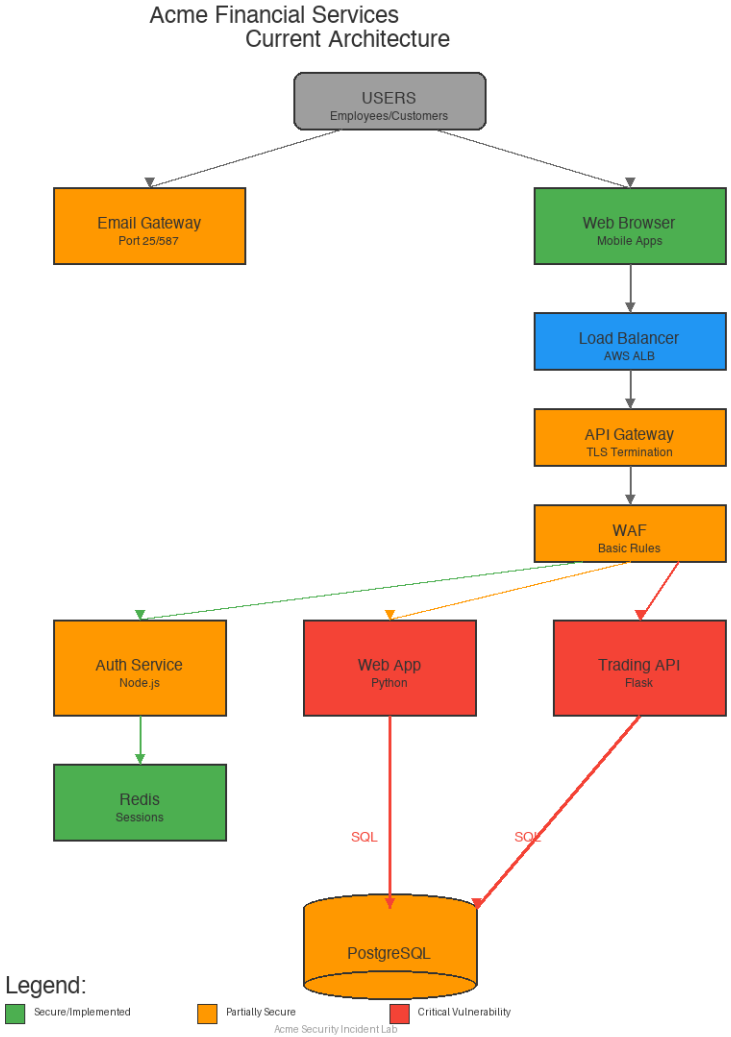

Figure 1: Original Architecture with Identified Failure Points

3.1. Identified Architectural Flaws

The following analysis details the critical weaknesses present in the original architecture:

1. **Lack of Centralized Policy Enforcement:** Authorization logic is either missing or incorrectly implemented deep within individual applications. The API Gateway acts merely as a TLS termination point, failing to enforce critical security policies like ownership checks.
2. **Direct and Unsafe Database Connectivity:** The "Critical Vulnerability" lies in the direct line of communication from the Web App to the PostgreSQL database. This design pattern encourages dynamic SQL query construction and structurally enables SQL injection attacks.
3. **Ineffective Edge Security:** The WAF is misconfigured with "Basic Rules" and passive detection, making it trivial to bypass with moderately sophisticated obfuscation techniques. Its position behind the API Gateway is also suboptimal.
4. **Weak Email Security Posture:** The Email Gateway is "Partially Secure" at best, lacking the DMARC, DKIM, and SPF enforcement required to prevent domain spoofing, which directly enabled the phishing attack.

3.2. Recommended Secure Architecture (To-Be State)

To address the identified flaws, a complete architectural redesign based on the principles of **Defense-in-Depth** and **Zero Trust** is required. The recommended secure architecture is detailed in Figure 2.

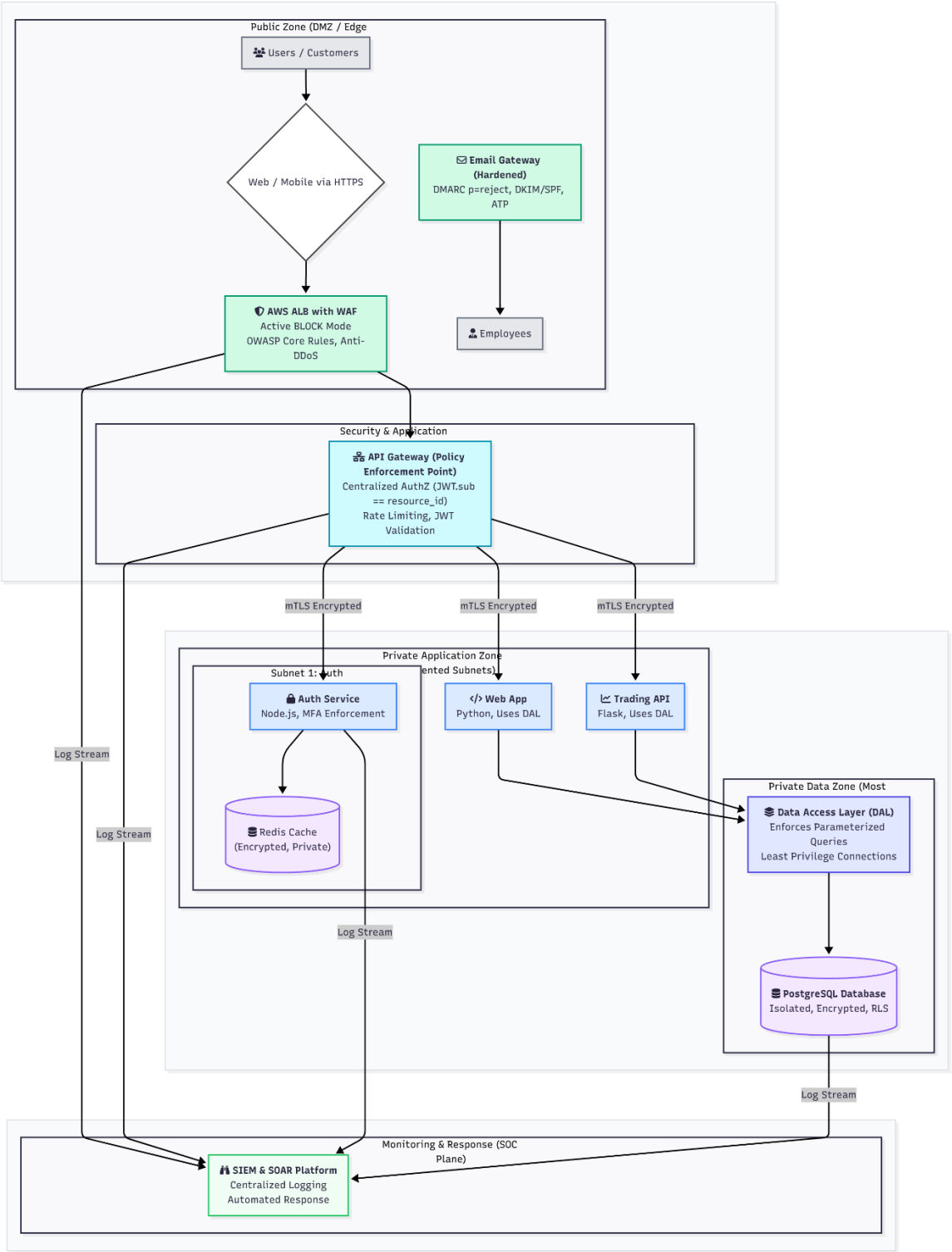

Figure 2: Recommended Secure Architecture with Defense-in-Depth

This new architecture establishes a multi-layered model that ensures security is enforced at every stage, preventing a single point of failure from compromising the entire system.

Key Architectural Controls and Justifications:

1. **Hardened Edge Layer (DMZ):** The **WAF** is repositioned to the true edge and operates in an active BLOCK mode with advanced rules. The **Email Gateway** is hardened with a mandatory DMARC p=reject policy, structurally preventing the phishing and initial SQLi vectors at the earliest point.
2. **API Gateway as a Policy Enforcement Point (PEP):** The Gateway is elevated from a simple proxy to a critical security control. It enforces **centralized authorization** (JWT.sub == resource_id) for every request, mitigating the BOLA/IDOR vulnerability before it can reach the application layer.
3. **Introduction of a Data Access Layer (DAL):** Applications are now decoupled from the database. The DAL acts as a mandatory intermediary that enforces the exclusive use of **parameterized queries**, structurally eliminating the risk of SQL injection. This removes the "direct and unsafe database connectivity" flaw.
4. **Private Zone Segmentation & Least Privilege:** Application services are isolated in **private, segmented subnets** to prevent lateral movement. The PostgreSQL database resides in the most secure data zone, hardened with **Row-Level Security (RLS)** and accessible only via the DAL with least-privilege credentials.
5. **Comprehensive Monitoring & Response:** Every critical component, from the WAF to the Database, forwards logs to a central **SIEM & SOAR Platform**. This enables real-time correlation of suspicious events and provides the foundation for automated incident response.

4.0 Response & Remediation Plan

4.1. Phase 1: Immediate Actions (Containment & Eradication, 0-24 Hours)

1. **Block Attacker IP:** Immediately create a rule in the edge firewall/WAF to permanently block all traffic from 203.0.113.45.
2. **Revoke All Compromised Credentials:**
   - Forcefully invalidate the session for jwt_token_1523_stolen.
   - Mandate an immediate password reset for user_id: 1523 and all phishing victims (User1, User3, User5).
3. **Disable High-Risk Functionality:** Temporarily disable the /dashboard/export endpoint for all non-administrative users pending a full security review.
4. **Preserve Evidence:** Take forensic snapshots of all involved systems (Web, API, DB servers). Securely archive all relevant raw logs from the incident window for legal and analytical purposes.

4.2. Phase 2: Short-Term Fixes (Hardening, 1-2 Weeks)

1. **Patch Critical Vulnerabilities:**
   - **BOLA:** Implement the JWT subject vs. resource ID check at the API Gateway level as an immediate hotfix.
   - **SQLi:** The development team must rewrite the /dashboard/search functionality to use parameterized queries exclusively.
2. **Harden WAF Configuration:** Switch all relevant SQLi and enumeration rules from DETECT to BLOCK. Deploy a virtual patch specifically targeting MySQL versioned comment bypass techniques.
3. **Deploy Email Authentication:** Publish SPF, DKIM, and DMARC DNS records for acme.com in a p=quarantine (monitor-only) mode to begin analysis of mail flow.

4.3. Phase 3: Long-Term Improvements (Strategic, 1-3 Months)

1. **Full Architectural Refactoring:** Begin the project to implement the recommended secure architecture, prioritizing the introduction of the Data Access Layer (DAL).
2. **Mandate Multi-Factor Authentication (MFA):** Roll out and enforce MFA (TOTP or WebAuthn) for all customer and internal employee accounts.

3. **Integrate Security into DevOps (DevSecOps):** Embed Static (SAST) and Dynamic (DAST) Application Security Testing tools into the CI/CD pipeline to catch vulnerabilities before they reach production.
4. **Enhance Security Monitoring:** Implement high-fidelity detection rules in a SIEM to correlate events across layers (e.g., alert on WAF: SQLi Detected + Web: HTTP 200 + Web: Large Response Size).
5. **Formalize DMARC Enforcement:** After a sufficient monitoring period, upgrade the DMARC policy from p=quarantine to p=reject to completely block all spoofed emails.

5.0 Compliance & Governance Implications

This incident constitutes a reportable data breach under regulations such as **GDPR (Article 33)** and **KVKK**, due to the unauthorized access to personal and financial data. It also reveals non-compliance with **PCI-DSS requirements 6.5 (Secure Coding)** and **10.2 (Logging and Monitoring)**. The legal department must be formally engaged to manage regulatory notifications. The successful implementation of the remediation plan outlined in this report will address these compliance gaps and provide auditable evidence of a robust, policy-driven security posture.