Security Incident Response Report

Organization: Acme Financial Services

Report Date: October 16, 2024

Classification: CRITICAL - Data Breach & Unauthorized Access

Prepared by: Metehan Çevik

Executive Summary

On October 15, 2024, Acme Financial Services faced multiple attacks including unauthorized access, injection, and data exfiltration. First of all, this is not a security test, the vendor scheduled to test between October 20 to 25 and they supposed to notify us 2 days before test. Test account id range between 5001-5010. The attacker's IP is that 203.0.113.45. Attacker's ip was in the approved test range, our IP range 203.0.113.0/24, but the attack on October 15 that day before five days the testing days . The attacker also used a real user account and their id is 1523, victims account not in test range. Firstly, an attacker access the account of customer ID 1523, then stole credentials and access other customers' portfolios using an Broken Object Level Authorization (BOLA) vulnerability and this attack affected at least 15 customers they could access 15 different account using this way. Subsequently, the attacker bypassed the Web Application Firewall (WAF) and injected SQL queries, resulting in the exfiltration of approximately 1.14 MB of sensitive financial data.This incident shows us that our system has serious problems like access control issue, input checks, and user authentication. Attacker founded weak points in our application and exploited these weak points. We have already started actions to improve our security and this report explains what happened, how did it happen, what will be done, what are the suggestions, and a list of important steps for proposed solutions.

Section 1: Incident Analysis

Timeline Reconstruction

The incident unfolded across multiple attack phases spanning several hours on October 15, 2024. All timestamps are presented in the original log timezone (UTC-aligned).

Phase 0: Pre-Attack Credential Compromise (Unknown Timestamp)

We coudn't find the how the attacker got user 1523's login informations. We checked the logs but logs didn't show us when or how credentials were taken but at 06:45:10 UTC, attacker has user 1523's login credentials and login with these credentials. There are a few possibilities firstly, maybe they used credentials from their own account, or maybe they got the login information from a different data breach and tried using it on our system and tjey can access the user account. We don't have clear evidence of the initial compromise, we can only guess at how they originally take the credentials.

Phase 1: API Exploitation and Initial Access (06:45 - 06:48 UTC)

At 06:45:10, the attacker log in the application using valid credentials for user 1523. They connected from IP address 203.0.113.45. The login request to /api/v1/login worked successfully - it returned a 200 status code which means okey and our services gave them an authentication token. The attacker made their traffic look normal by using a user agent string that matched our mobile app: "Acme-Mobile-Android/3.2.0".About 80 seconds after logging in, at 06:46:30, the attacker checked the account (portfolio ID 1523). At 06:47:15, the attacker checking other user accounts one by one. They checked account IDs from 1524 to 1538. The requests came in every three seconds, which shows they were using some kind of automated script or tool and bypass the time limits. All 15 attempts to access these accounts worked. Each request got a 200 response, attacker successfully pulled information from victim accounts . They used the same stolen token (jwt_token_1523_stolen) for everything.

Phase 2: Secondary Phishing Campaign (09:00 UTC)

At 09:00:23, the attacker sent phishing emails to our users. The emails came from security@acme-finance.com with the subject "URGENT: Verify Your Account - Action Required. The emails sent every 2 seconds, so the attacker was probably using an automated tool. Six users got the email, and three of them clicked the link. We know it was the same attacker because the emails came from their IP address.

Phase 3: Web Application Attack (09:18 - 09:23 UTC)

At 09:18:30, the attacker logged into the web application using user 1523's stolen credentials from IP 203.0.113.45. After logging in, they went to the dashboard at 09:19:15 and started testing for SQL injection attacks. Between 09:20:30 and 09:22:00, the attacker tried three different SQL injection attacks on the dashboard's search feature. Our application blocked these attempts and returns 403 which means forbidden errors. But at 09:23:45, the attacker found bypass technique. They used payload "ticker=AAPL' /!50000OR/ 1=1--". This SQL injection query pulled 156,789 bytes of database data. The WAF could detect only 3 sql injection query. We must take actions about that, like whitelist and new rules for WAF.

Phase 4: Data Exfiltration (Post 09:24 UTC)

Log analysis shows us that at 09:24, The attacker used the export function at /dashboard/export with CSV format. This request returned 892,341 bytes of data - that's roughly 892 KB of sensitive information. The export feature is a normal dashboard function, so it didn't trigger any alerts.

Attack Vector Identification

Primary Vector: Broken Object Level Authorization (BOLA)

Our application only checks if a valid token exists in the request header. It doesn't check user permission. This BOLA vulnerability lets users view other people's portfolios. They have user credentials for customer 1523 and can request for porfolio. They could request for other customer with using customer 1523's JWT token.

Secondary Vector: SQL Injection via WAF Bypass

The web application's dashboard search functionality can exetcute SQL queries using with user input, this situation creating a SQL injection vulnerability classified as OWASP A03:2021 Injection. Attacker used bypass technique as "ticker=AAPL' /!50000OR/ 1=1--". This SQL injection query pulled 156,789 bytes of database

Tertiary Vector: Phishing Email

Attacker can sent email to our users from an email address outside our domain, and our users are not sufficiently trained against phishing attacks.

Attack Classification

OWASP Top 10 Classification

The incident demonstrates multiple OWASP Top 10 2021 vulnerabilities:

API1:2023 Broken Object Level Authorization (BOLA) is explained in the OWASP handbook as follows. Object level authorization is an access control mechanism that is usually implemented at the code level to validate that a user can only access the objects that they should have permissions to access.

A03:2021 Injection is explained in the OWASP handbook as follows. An application is vulnerable to attack when: User-supplied data is not validated, filtered, or sanitized by the application and Dynamic queries or non-parameterized calls without context-aware escaping are used directly in the interpreter.

A09:2021 Security Loggining and Monitoring Failures is explained in the OWASP handbook as follows.This category is to help detect, escalate, and respond to active breaches. Without logging and monitoring, breaches cannot be detected. Insufficient logging, detection, monitoring, and active response occurs any time:Auditable events, such as logins, failed logins, and high-value transactions, are not logged.Warnings and errors generate no, inadequate, or unclear log messages.Logs of applications and APIs are not monitored for suspicious activity.

Root Cause Analysis

The main problem is missing authorization checks in our API. The system only verified a token is valid but don't check user's permission to access the requested data. This let the attacker use one user's token to view 15 other accounts. The dashboard had SQL injection vulnerabilities, dashboard can execute user input directly into SQL queries. Our WAF blocked basic attacks but missed the obfuscated payload. We didn’t have security code reviews during development. Our monitoring systems failed to alert us during the attack. The WAF was configured poorly - some rules were in detect mode instead of block mode.

Impact Assessment

The attacker accessed 16 user accounts (IDs 1523-1538) and stole approximately 1.14 MB of sensitive financial data. Stolen data include portfolio information and personal financial records. The data was extracted in two stages - 156 KB through SQL injection and 892 KB through the export function. Attacker send phishing mail for 6 users and 3 of them clicked the malicious links, potentially exposing this 3 users’credentials too.We faced costs for incident response and security upgrades. The breach could lead to customer compensation claims and legal fees. We must report to regulatory authorities and inform all affected customers.

Section 2: Architecture Review

Current Architecture Weaknesses

First of all, There isn’t additional verification step when user log in, like a OTP verification. API request only check whether there is a token or not, don’t check user permission. WAF rules are incorrectly configured and not enough, WAF can be bypassed and time limiting is not enough. Monitoring is not enough, it should more controlled automatically and Manual control and blocking should be implemented against attacks. Mail servers are incorrectly configured, it lets that user which outside the domain can send mail and it cound’t check mail content clearly.

Improved Security Architecture

We need to add Multi-Factor Authentication (MFA) for all user logins. Users need to enter a code from their phone or email after enter their password. We'll also implement account lockout after 5 failed login attempts and require strong passwords with regular updates. API will checks the user permission not only is JWT valid. We will add rate limiting for request and Any suspicious patterns like rapid sequential account access will automatically black list the token for 3600 seconds, this makes the attacker's token useless when we catch them, and it stays blocked until the token naturally expires. We will improve WAF configuration like add new rules will catch obfuscated SQL injection attempts like the one used in this attack, rate limits and implementing automatic IP blocking. All SQL queries will use parameterized statements and validates input. We will deploy SIEM solution that monitors all logs in real-time. It will automatically alert security teams when it sees suspicious activity. We will configure DMARC, SPF, and DKIM to prevent email spoofing, Only authorized mail servers can send emails from our domain. We're also adding content filtering to scan for phishing attempts and malicious links.

Recommended Security Controls

We need API authorization checks that validate user permissions. Rate limiting will be 40 requests per minute with JWT blacklisting. All SQL queries must use parameterized statements with input validation. MFA is required for all accounts with account lockout after 5 failed attempts. SIEM provide real-time monitoring with 90-day log retention and automated alerts.

Defense-in-Depth Strategy

Multiple security layers will protect our systems. Enhanced WAF with DDoS protection forms the perimeter. TLS 1.3 encryption and network segmentation secure communications. Application layer validates inputs and uses parameterized queries. Database encryption protects data at rest. MFA and least privilege control access. Real-time monitoring detects and blocks threats automatically.

Immediate Actions (0-24 Hours)

We blocked the attacker's IP, reset compromised credentials, and disabled the stolen token. Export function is disabled and WAF is in blocking mode.

Short-Term Fixes (1-2 Weeks)

Adding API authorization checks and parameterized SQL queries. Implementing rate limiting and token blacklisting. Configuring DMARC, SPF, DKIM for email. Updating WAF rules and adding MFA. Testing all fixes.

Long-Term Improvements (1-3 Months)

We will add security checks to all code before release and following DevSecOps practices. SIEM system will monitor everything and respond automatically for threats. We will launch security tests every quarter. Monthly training for developers and quarterly phishing tests will keep everyone alert.

Compliance Considerations

We must tell authorities within 72 hours and notify all affected customers about the breach. We're recording everything we did to respond and offering identity protection services to victims. We're following NIST and ISO 27001 security standards. Independent auditors will check our compliance every year.