Acme Financial Services - Incident Report

Prepared by: Cihan Kan
Incident Date: 15/10/2024

Section 1: Incident Analysis

Acme Financial Services was the target of a multi-vector attack with three main components on October 15, 2024. The attack started with a phishing campaign, then exploited an API vulnerability known as Insecure Direct Object References(IDOR), and finally launched a SQL Injection (SQLi) attack against the Web application. The most important discovery is the API's lack of authorization control, which permitted illegal access to customer portfolios and gave the attacker the means to get past the Web Application Firewall (WAF) and steal large amounts of data.

During the investigation, an insider thread was detected, and it was observed that meeting notes and user data were downloaded.

1.1 Timelines

1.Attacker Timeline (UTC)

Time (UTC)	Event	Evidence File
15/10/2024 09:00	The attacker sent phishing emails to employees using a spoofed address (security@acme-finance.com)	email_logs.csv (lines 3–8)
15/10/2024 09:00	User1, User3, User5 clicked the phishing link. The JWT token belonging to user with ID 1523 was stolen.	email_logs.csv (lines 3,5,7)
15/10/2024 14:45	The attacker logged into the system using the stolen JWT token.	api_logs.csv (line 19)
15/10/2024 14:47	The attacker exploited a vulnerability in the API Gateway and enumerated the portfolios of 15 users(1524-1538). The WAF detected this activity but took no action.	api_logs.csv (lines 20–35), waf_logs.csv (lines 9–11)
15/10/2024 17:20	The attacker attempted SQL Injection (OR 1=1--), which was blocked by the WAF	web_logs.csv (line 10), waf_logs.csv (line 2)
15/10/2024 17:21	The attacker attempted SQL Injection (DROP TABLE users--) which was blocked by the WAF.	web_logs.csv (line 11), waf_logs.csv (line 3)
15/10/2024 17:22	The attacker attempted SQL Injection (UNION SELECT * FROM users--) which was blocked by the WAF.	web_logs.csv (line 12), waf_logs.csv (line 4)
15/10/2024 17:23	The attacker used a WAF evasion payload (/*!50000OR*/ 1=1--) and successfully received a valid response. The WAF detected the attack but took no action.	web_logs.csv (line 13), waf_logs.csv (line 5)
15/10/2024 17:24	The attacker downloaded 892,341 bytes of data in CSV format.	web_logs.csv (line 14)

2. Scheduled Security Testing Timeline(UTC)

Time (UTC)	Event	Evidence File
15/10/2024 09:30	Weekly automated scan.	api_logs.csv (lines 2-6) security_test_schedule.pdf
15/10/2024 09:45	Sec_team users(5001-5010) testing auth.	api_logs.csv (lines 7-11) security_test_schedule.pdf

3. Insider Threat Timeline(UTC)

Time (UTC)	Event	Evidence File
15/10/2024 08:55	Admin exporting meeting notes external mail.	email_logs.csv (lines 2)
15/10/2024 16:55	admin_5678 started exporting data.	web_logs.csv(line 2)
15/10/2024 16:56	admin_5678 downloaded 245890 bytes of data in CSV format.	web_logs.csv(line 3)

1.3 Attack vector identification

The analysis shows the attacker used multiple attack vectors:

1. Spearphishing Link

Acme.com is the main domain of Acme Financial Services. To conduct the phishing campaign, the attacker sent malicious emails from security@acme-finance.com to six employees, impersonating a similar-looking domain, acme-finance.com. The phishing emails were created by the attacker with the intention of obtaining users' JWT tokens. Three employees' tokens were successfully acquired.

2. Insecure Direct Object References (IDOR)

The attacker gained access to other users' accounts by simply increasing the id value in the URL after logging into the system with the stolen token that belonged to the user with ID 1523. The attacker viewed the portfolios of fourteen distinct users(1524-1538) by taking advantage of this vulnerability. The API documentation note that reads, "Authorization checks validate token but may not verify account ownership," is in line with this behavior.

3. SQL Injection

The attacker attempted the following SQL injection payloads. The first three payloads were blocked by the WAF, but the last payload bypassed the WAF and enabled a successful data extraction resulting in the download of 892,341 bytes.

```
OR 1=1
DROP TABLE users--
UNION SELECT * FROM users
/!50000OR/ 1=1--
```

1.4 Attack classification (MITRE ATT&CK, OWASP)

1. MITRE ATT&CK

ID	Name	Description
T1566	Spearphishing Link	Adversaries may send spearphishing emails with a malicious link in an attempt to gain access to victim systems.
T1539	Steal Web Session Cookie	An adversary may steal web application or service session cookies and use them to gain access to web applications or Internet services as an authenticated user without needing credentials.

2. OWASP

ID	Name	Description
A01:2025	Broken Access Control	Access control enforces policy such that users cannot act outside of their intended permissions. Failures typically lead to unauthorized information disclosure, modification, or destruction of all data or performing a business function outside the user's limits.
A05:2025	Injection	Injection flaws occur when untrusted data is sent to an interpreter as part of a command or query. The attacker’s hostile data can trick the interpreter into executing unintended commands or accessing data without proper authorization.
A02:2025	Security Misconfiguration	Security Misconfiguration is the most commonly seen issue. This typically is a result of unsecured default configurations

1.5. Root Cause Analysis

1. Human Security Weakness:
Employees were not sufficiently trained or alerted regarding phishing attacks. The fact that three users clicked suspicious links demonstrates a lack of security awareness.

2. Weakness Email Security:
The email filtering system failed to detect the phishing emails, allowing the malicious messages to reach employee inboxes.

3. Lack of Protection Against JWT Token Theft:
JWT tokens were not bound to IP address, device ID, or geolocation. This allowed stolen tokens to be reused without detection.

4. API Authorization Flaw:
The API validates tokens but does not verify whether the requested account actually belongs to the authenticated user, enabling unauthorized access.

5. Insufficient Parameter Filtering in the Web Application:
Successful SQL injection attempts indicate that the application lacks proper input validation and sanitization mechanisms.

6. Missing or Insufficient WAF Rules:
Although the WAF detected suspicious SQL patterns, it did not take any blocking action, allowing one of the payloads to bypass the protection mechanisms.

1.6 Impact Assessment

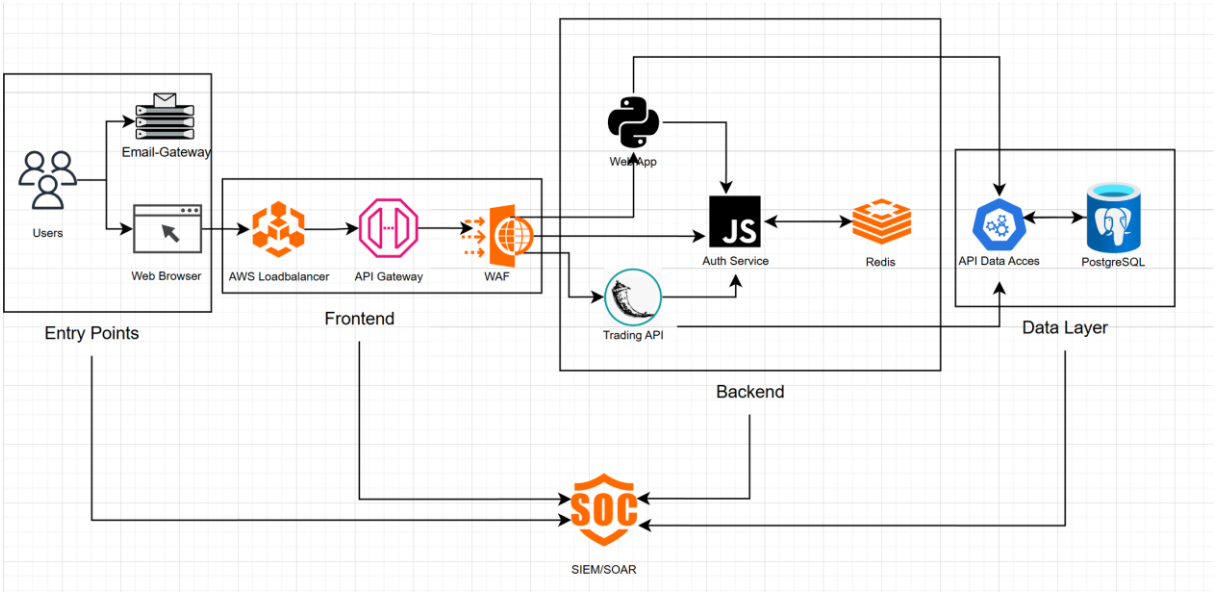
Category	Impact
Data Leak	Exposure of 15 user accounts + CSV export (892 KB)
Financial Risk	Potential fraud and reputational damage
Compliance Violation	Possible GDPR and PCI-DSS implications

Section 2: Architecture Review

2.1 Current architecture weaknesses

Component	Weakness
API Gateway	Rate limit values are too high, reducing protection against enumeration and brute-force attempts.
WAF	Rules are insufficient and fail to block certain SQL injection payloads.
Email Gateway	Lacks robust filtering and sandbox-based malware analysis.
Web App (Python)	Contains critical security vulnerabilities.
Trading API (Flask)	Contains critical security vulnerabilities.
Auth Service (Node.js)	Tokens are not validated against IP address, device ID, or geolocation, enabling token theft attacks.

2.2. Improved security architecture diagram



2.3 Recommended security controls

1. Identity and Access Management (IAM)

- Enforce additional verification when abnormal location, device, or user behavior is detected.
- Include an account_id claim within JWT tokens and perform ownership validation on every API request.

2. Application Security

- Implement strict input validation to sanitize and validate all user-supplied data.
- Deploy advanced WAF rule sets for improved protection against injection and evasive payloads.
- Provide developers with OWASP Top 10 and Secure SDLC training programs.

3. API Security

- Enforce strict resource ownership checks to ensure users access only their own data.
- Reduce rate limit thresholds to mitigate enumeration and brute-force attempts

4. Email Security

- Scan all email links through a security inspection layer before user access.
- Analyze all attachments in an isolated sandbox environment.

5. Security Monitoring and Response

- Integrate SIEM with SOAR to automate detection, triage, and incident response workflows.

Section 3: Response & Remediation

3.1 Immediate Actions (0–24 Hours)

- Block the IP address 203.0.113.45.
- Update WAF rule sets to actively block Suspicious SQL Pattern, Rapid Sequential Access, and Possible Account Enumeration alerts.
- Revoke and reset the JWT tokens and passwords of employees User1, User3, and User5.
- Restrict and notify all users between IDs 1523–1538, as well as any users whose data was included in the attacker's 892,341-byte CSV download.
- Submit a GDPR data breach notification due to the unauthorized exposure of personal financial data.

3.2 Short-Term Actions (1–2 Weeks)

- Implement comprehensive input validation across the entire application.
- Adjust and lower rate limit thresholds to mitigate enumeration attacks.
- Enforce account ownership validation within all API calls.
- Deploy an advanced WAF with updated rule sets.
- Implement a modern email security solution (link inspection, sandboxing, malware detection).

3.3 Long-Term Improvements (1–3 Months)

- Conduct a black-box penetration test to validate the effectiveness of the immediate and short-term fixes.
- DLP solution should be used to protect against insider threats.
- Transition to a new, secure architecture aligned with zero-trust principles.
- Provide secure coding training to development teams.
- Provide cybersecurity awareness training to all employees.

3.4 Compliance Considerations

1. GDPR (General Data Protection Regulation)

- Unauthorized access to personal financial data constitutes a GDPR violation.
- Impacted data subjects must be notified.

2. PCI-DSS (Payment Card Industry Data Security Standard)

- Protection against SQL Injection is mandatory.
- Multi-Factor Authentication (MFA) is required for critical systems.

3. SOC 2 Type II

- Monitoring controls for system operations were insufficient.
- Logical and physical access controls failed to prevent unauthorized access.