

# Section 1: Incident Analysis

## Phishing Attack

**Target :** 6 employees

**Success :** 3 users clicked

It is **unknown how much information was obtained**. This must be investigated, and the relevant emails should be reviewed. The information/details of the users who clicked should be checked.

## Violated Security Standards:

**ISO 27001:2022**

- A.6.8 - Information security awareness training

**MITRE ATT&CK:**

- T1566 - Phishing

2024-10-15 09:00:23	security@acme-finance.com	user1@acme.com	yes
2024-10-15 09:00:25	security@acme-finance.com	user2@acme.com	no
2024-10-15 09:00:27	security@acme-finance.com	user3@acme.com	yes
2024-10-15 09:00:29	security@acme-finance.com	user4@acme.com	no
2024-10-15 09:00:31	security@acme-finance.com	user5@acme.com	yes
2024-10-15 09:00:33	security@acme-finance.com	user6@acme.com	no

## SQL Injection With WAF Bypass

09:20:30	/dashboard/search	ticker=AAPL' OR 1=1--	403	Yes
09:21:15	/dashboard/search	ticker=AAPL'; DROP TABLE users--	403	Yes
09:22:00	/dashboard/search	ticker=AAPL' UNION SELECT * FROM users--	403	Yes
09:23:45	/dashboard/search	ticker=AAPL' /*!50000OR*/ 1=1--	200	No

The initial **three attempts** by the testing team were successfully blocked and detected in the WAF logs. However, the final attack, an OR-based injection, bypassed the WAF without being blocked. This success was due to insufficient or weak **WAF** rules. The attacker exploited a **WAF bypass** technique (specifically, using a comment method) to pass the payload targeting the **ticker** parameter on the **/deashboard/search** endpoint.

The payloads that using in logs are specialized below :

#### **Detected and Blocked :**

- ticker=AAPL' OR 1=1-

Detected as a **OR Attack**.

- ticker=AAPL'; DROP TABLE users--

Detected as a **Drop Table Attack**

- ticker=AAPL' UNION SELECT \* FROM users-

Detected as a **UNION Attack**

#### **Detected but not Blocked :**

- ticker= AAPL' /\*!50000OR\*/ 1=1--

Detected as a SQL attack but **not blocked** because **obfuscation** this way WAF bypassed and the attack was successful without being blocked.

### **Violated Security Standards:**

#### **ISO 27001:2022 :**

- **A.8.3** : Information Access Restriction, The best way to protect information security is with access control and information access restrictions.

#### **OWASP :**

- **A03:2021-Injection**

### **IDOR Attack in API**

**2024-10-15 14:45:10 UTC:** Initial login activity detected.

**2024-10-15 14:46:30 UTC:** User 1523 successfully authenticated and obtained a valid **JWT** token.

**2024-10-15 14:47:15 UTC to 14:47:57 UTC:** Immediately following authentication, the user initiated an automated attack. The logs show requests to the **/api/v1/portfolio/{account\_id}** endpoint occurring at a regular 3-second interval.

**Observation:** The logs indicate that information belonging to **multiple users** was accessed, not just the attacker's own data.

This activity appears to be a direct exploitation of a known weakness documented in the **api\_docs.pdf** file, which states:

"Authorization checks validate token but may not verify account ownership."

### **Violated Security Standards:**

#### **ISO 27001:2022:**

- A.8.3 - Information access restriction

#### **OWASP API:**

- API1:2023 Broken Object Level Authorization

While analyzing log timestamps, identified that **Api\_logs** utilize **Pacific Standard Time (PST)**, departing from the **UTC** standard used in other logs. This finding was verified against the **security\_test\_schedule.pdf** file, which lists '**Test1**' beginning at **01:30 AM PST**. The **Api\_logs** entries align with this PST schedule

PST	UTC	user_id	Endpoint		acc_id
06:45:10	14:45:10	1523	/api/v1/login	POST	
06:46:30	14:46:30	1523	/api/v1/portfolio/1523	GET	1523
06:47:15	14:47:15	1523	/api/v1/portfolio/1524	GET	1524
...	...	...	...	...	...
06:47:54	14:47:54	1523	/api/v1/portfolio/1537	GET	1537
06:47:57	14:47:57	1523	/api/v1/portfolio/1538	GET	1538

## Security Testing

**Log:** api\_logs.csv line 2-6

**IP :** 192.168.1.100

**Response :** 401 (failed)

Analysis confirms that these initial requests represent an unauthenticated reconnaissance phase. The 401 Unauthorized responses correctly indicate that the API rejected these attempts because they were made without **JWT token**. This demonstrates that the system's authentication controls are functioning correctly by denying access to unauthenticated users. The attack shown in these logs was the result of a planned penetration test conducted by the pentest team, as indicated in the **security\_test\_schedule.pdf** file.

## Log injection

**Log Entry:** 2024-10-15 09:30:00,1523,/dashboard/home,**200"**,200,8934,203

The fourth field, intended to record the **HTTP Status Code**, contains the value **200"**. This injected quote character ("") appears to be a deliberate attempt to break the log's structured format

## Sec\_team Security Testing

api\_logs.csv içerisindeki 7-11 lineleri ve arasında sec\_team security\_test\_schedule.pdf dosyasında belirtildiği gibi test yapmıştır.

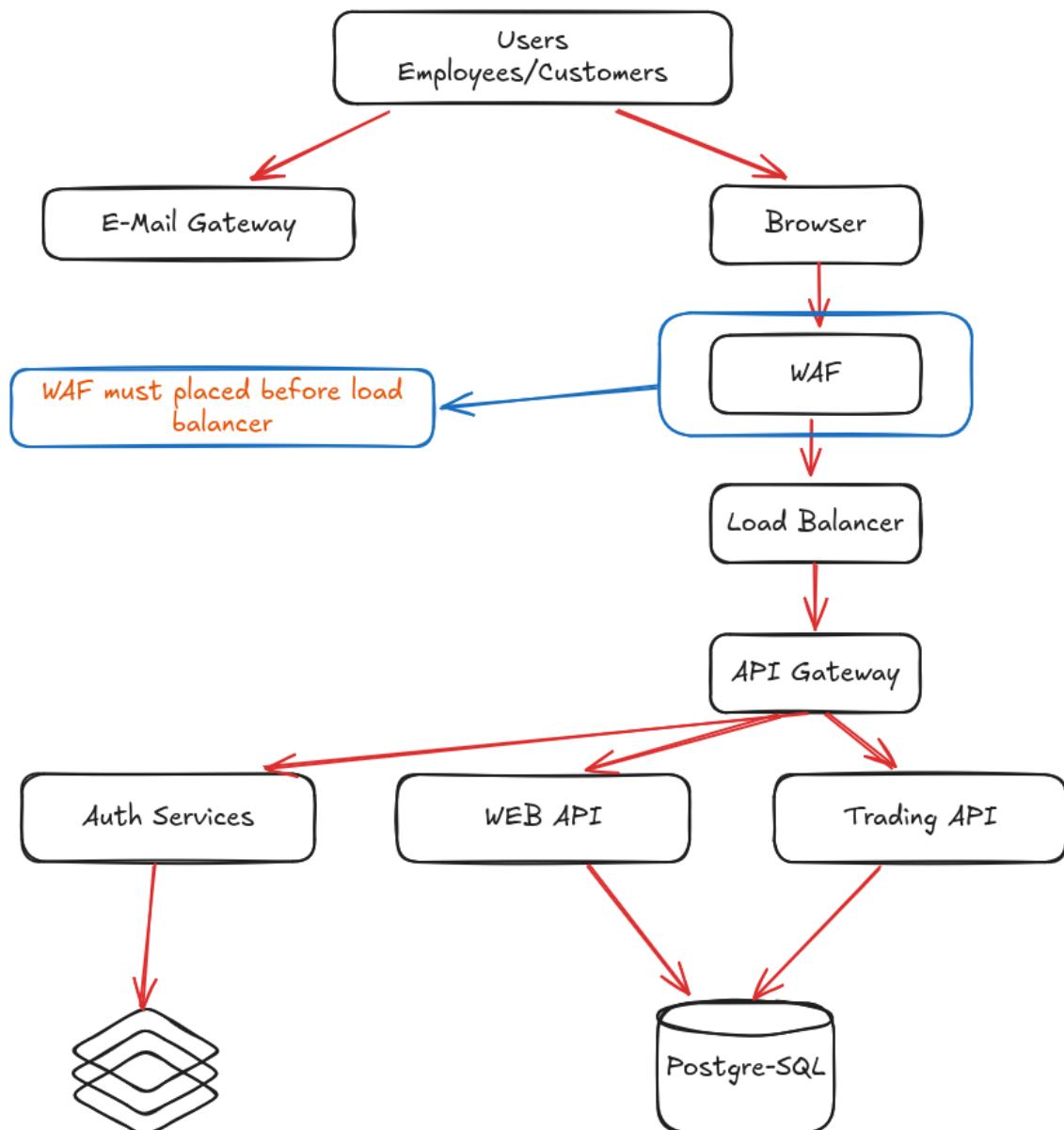
IP: 10.0.0.50

Documented in: security\_test\_schedule.pdf

## Section 2 : Architecture Review

Upon reviewing the architecture, the following significant issues were identified, and the proposed architecture is shown below. WAF security detects and filters out threats which could degrade, compromise, or expose online applications to denial-of-service (DoS) attacks. WAF security examines HTTP traffic before it reaches the application server. They also protect against unauthorized transfer of data from the server. In most application architectures, the WAF is best positioned behind the load balancing tier to maximize utilization, performance, reliability and visibility. WAFs are an L7 proxy-based security service and can be deployed anywhere in the data path. However, we recommend positioning WAFs closest to the application they are protecting behind the load balancing tier to optimize your architecture for utilization, performance, reliability, and visibility.

- The WAF should be the first component to handle requests in the system. Therefore, it should be placed **before** the Load Balancer.



# Section 3: Response & Remediation

## Immediate Actions :

- Investigations must be initiated regarding the operations performed by the internal **admin user**.
- WAF rules must be corrected to eliminate the **bypass vulnerabilities**.

## Short-term fixes :

- The SQLI vulnerability found in the ticker parameter on the /dashboard/search path of the web server must be remediated. User inputs should be checked using the **Whitelist technique**.
- Since the operations performed on the database could not be viewed, **Database query logs** and **logs for the authentication service** must be maintained.

## Long-term improvements :

- The WAF's position in the architectural structure must be changed so that it is the first component to handle requests coming from users.
- Users must receive **Security Awareness training** to ensure they are more resilient/sensitive to phishing attacks.