Security Incident Report

Prepared by: **Demir Ata Albuz**

Date: November 9, 2025

Severity: **CRITICAL**

Security Incident Report: Demir_Albuz_Report.pdf
Date of Incident: October 15, 2024 Report Date: November 9, 2025 Severity: CRITICAL Prepared by: Demir ata Albuz

Section 1: Incident Analysis (Page 1-2)

1.1. Incident Timeline Reconstruction (UTC Normalised)
The incident was a multi-stage attack that happened on October 15, 2024. The attacker used the IP address 203.0.113.45 and the compromised credentials of user_id 1523 for all actions.

The attack started at 06:45:10 UTC, when the attacker logged into the mobile API (/api/v1/login). Just one minute later, at 06:46:30, the attacker tested their access by checking their own portfolio (/portfolio/1523). Between 06:47:15 and 06:47:57, the attacker launched a fast, automated attack, stealing data from 15 other user accounts (from 1524 to 1538) in only 42 seconds.

After a break, at 09:00:23 UTC, the attacker started a phishing campaign. They sent emails with the subject "URGENT: Verify Your Account" to multiple users, including user1, user3, and user5. This was a "backup plan" to get more passwords .

While the security team was likely distracted, at 09:18:30 UTC, the attacker logged into the web application. At 09:23:45 UTC, they used a special SQL Injection payload (/*!50000OR*/ 1=1--) to bypass the WAF. Immediately after this, at 09:24:10 UTC, the attacker exported the database using the /dashboard/export function.

1.2. Attack Vector and Classification (MITRE ATT&CK; & OWASP)
The attacker used three main vectors, which map to the OWASP Top 10 and MITRE ATT&CK; frameworks.

Vector 1: Broken Access Control (IDOR): The API attack at 06:45 was a classic OWASP A01:2021 - Broken Access Control. The attacker used a valid token for one user (1523) to access the data of other users (1524, 1525). In MITRE ATT&CK;, this is T1078 (Valid Accounts).

Vector 2: Phishing: The 09:00 email campaign was MITRE T1566 (Phishing), used for T1078 (Valid Accounts) (to get more credentials).

Vector 3: SQL Injection (WAF Bypass): The 09:23 web attack was OWASP A03:2021 - Injection. The attacker used a special comment-based payload (`/!.../) to bypass the WAF, which is MITRE T1595 (Active Scanning) and T1190 (Exploit Public-Facing Application).

1.3. Root Cause Analysis
The root cause of this incident is not just a technical failure, but a procedural and insider-knowledge failure.

The security_test_schedule.pdf shows that the attacker's IP (203.0.113.45) was on an "approved" list for a test. This test was planned for October 20. The attacker knew this secret IP and attacked 5 days early (October 15).

This proves the attacker was an insider threat. They were either an employee at "CyberSec Partners" or (more likely) an Acme employee (like user_id 1523) who had access to both their own account and the secret test plan. The attacker used the approved IP to hide their attack, hoping it would look like a "test."

1.4. Impact Assessment
The impact is CRITICAL. The attacker successfully stole two types of sensitive data:

Customer Portfolio Data: All financial holdings, names, and account values for at least 15 users were stolen via the IDOR attack.

Full Database: The SQL Injection attack at 09:24 likely exported the entire database, which could include all user data, passwords, and trading history.

This is a major data breach of sensitive financial information.

Section 2: Architecture Review (Page 3-4)

2.1. Existing Architecture Weaknesses
The current_architecture.png shows a system with a WAF, but the attack proved it had critical weaknesses:

Weak Authorization at the API: The API had no check to see if a user owned the data they were asking for . The api_docs.pdf even noted this weakness ("...may not verify account ownership").

Weak WAF Rules: The WAF was not configured to block advanced bypass techniques like /*!...*/ comments. It only blocked simple, known attack strings.

Weak Rate Limiting: The API did not stop the attacker from making 15 requests for 15 different accounts in just 42 seconds. The api_docs.pdf also noted this ("Rate limiting may not be strictly enforced").

No "Zero Trust" for Partners: The system had a rule to "Do NOT block" traffic from the test IP. This created a "trusted" hole that the attacker used.

2.2. Recommended Security Controls and Defense-in-Depth
I cannot create a new diagram, but I can describe the new "Defense-in-Depth" strategy. We must add security at every layer, not just at the front (WAF).

At the Application Code (Layer 1): This is the most important fix.

Fix IDOR: The API code must be changed. It must check if the user_id from the token matches the account_id in the URL request . If they do not match, it must return a 403 Forbidden error.

Fix SQLi: All database commands must use Parameterized Queries (Safe Commands). This stops SQL Injection at the code level, so we do not have to trust the WAF.

At the API Gateway (Layer 2):

Enforce Rate Limits: We must apply strict speed limits (like 60 requests/minute). If a user goes over this limit, they should be automatically blocked (429 Too Many Requests).

At the Network (Layer 3):

Smarter WAF: The WAF rules must be updated to block advanced bypass methods.

Smarter Monitoring: We must remove the "Do NOT block test traffic" rule. Instead, we should create a new alert (see Section 3.3).

Section 3: Response & Remediation (Page 5)

3.1. Immediate Actions (0-24 Hours)
The first step is to stop the bleeding. The IP address 203.0.113.45 must be permanently blocked on all firewalls. The user_id 1523 account must be immediately frozen, and all their active sessions ended. The passwords for all users who clicked the phishing link (user1, user3, user5) must also be reset. Finally, an internal investigation must begin, focused on user 1523 and anyone with access to the security_test_schedule.pdf. The vulnerable pages (/api/v1/portfolio/ and /dashboard/export) should be taken offline until they are fixed...

3.2. Short-Term Fixes (1-2 Weeks)
The development teams must immediately apply the code fixes described in Section 2.2. First, fix the IDOR weakness by adding the ownership check to the API . Second, fix the SQLi weakness by changing all database commands to use Parameterized Queries. Third, apply the strict rate limits at the API Gateway. These are the most critical technical fixes.

3.3. Long-Term Improvements (1-3 Months)
To prevent this from happening again, we must improve our long-term security. First, Multi-Factor Authentication (MFA) must become mandatory for all employees and all customers. This would have made it much harder to use the stolen 1523 password. Second, we must create a new "CRITICAL" alarm rule: if any traffic comes from a test IP range before the official test start date, the

security team must be alerted immediately. Finally, all staff must get new phishing training.

3.4. Compliance Considerations

This incident is a major data breach of financial information (Portfolio data). This means the company may be in violation of data privacy laws like GDPR (if EU customers are affected) or PCI-DSS (Payment Card Industry Data Security Standard). We must immediately notify the legal and compliance teams to begin the correct notification process for our customers.