

Name: S.Dhanish Ahamed

Reg No: 225BCE0001

Incident Response and Traffic Anomaly Analysis:

Based on the logs provided, this is a SQL injection attack leading to data exfiltration

Incident Analysis Report:

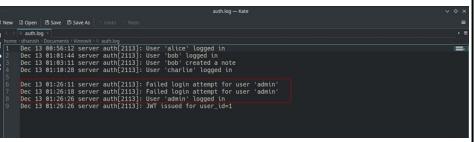
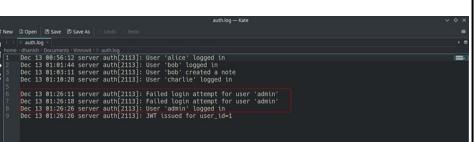
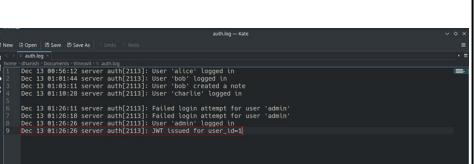
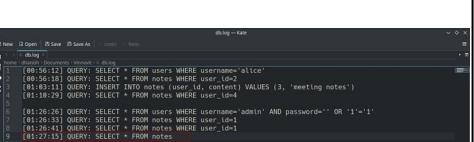
1) Executive Summary:

On Dec 13 between 01:26 AM and 01:27 AM, an external attacker (203.0.113.45) successfully compromised the application using SQL injection. The attacker bypassed authentication to gain administrative access and executed a database query that dumped the entire notes table. Approximately 190 KB of sensitive data was exfiltrated.

2) Technical Findings:

- Attack Vector: SQL injection via the /login POST parameter.
- Vulnerability: The application constructs SQL queries by concatenating user input without sanitization.
- Compromised accounts: The admin account was taken over via the auth bypass
- Data leak:
 - Initially, the application fetches notes for specific users
 - At 01:27:15, a query is executed without a WHERE clause: SELECT * FROM notes
 - This correlates with the 190kB outbound traffic spike in network_traffic.txt, confirming the database dump was sent to the attacker.
- Secondary attack:
 - The process_snapshot.txt shows a curl command running as root. This indicates the web service or a triggered script has root privileges.

3) Attack Timeline Reconstruction

Timestamp	Activity	Screenshots
01:26:11	Attacker attempts login as 'admin' (Failed).	
01:26:18	Second failed login attempt.	
01:26:26	SQL Injection Executed. Attacker sends payload ' OR '1='1 '. Auth bypass successful.	
01:26:26	JWT Token issued for User ID 1 (Admin).	
01:26:49	Attacker hits /debug endpoint. This likely triggered the root-level curl process seen in snapshots.	
01:27:15	Attacker requests /notes . The backend executes SELECT * FROM notes (dump all).	
01:27:15	190 KB data transfer to 203.0.113.45 completes.	

4) Mitigation Steps:

- Add the IP address to the firewall/security group deny list immediately
- Invalidate all active JWT tokens

- Temporarily disable the /debug endpoint at the web server level
- Rewrite the SQL logic to use parametrized queries