DIGITAL FORENSICS WRITEUP

Author: Nikhil Birla
Date: 16 October 2025
Lab: Reachkart
Difficulty Level: Medium
Evidence: network-traffic.pcap, access.logs

As an aspiring digital forensics analyst, this represents my learning journey. I welcome any feedback, corrections, or insights from experienced professionals to help me grow in this field. Some methodologies may reflect my current learning level, and I'm continuously working to improve my investigative techniques.

1. What was the vulnerable endpoint that allowed the attacker to leak files?

Answer - [redacted]

By analyzing the network traffic generated by the application to understand its structure and identify potential attack vectors.

Filtering: To focus on client-initiated requests, I applied the Wireshark filter:

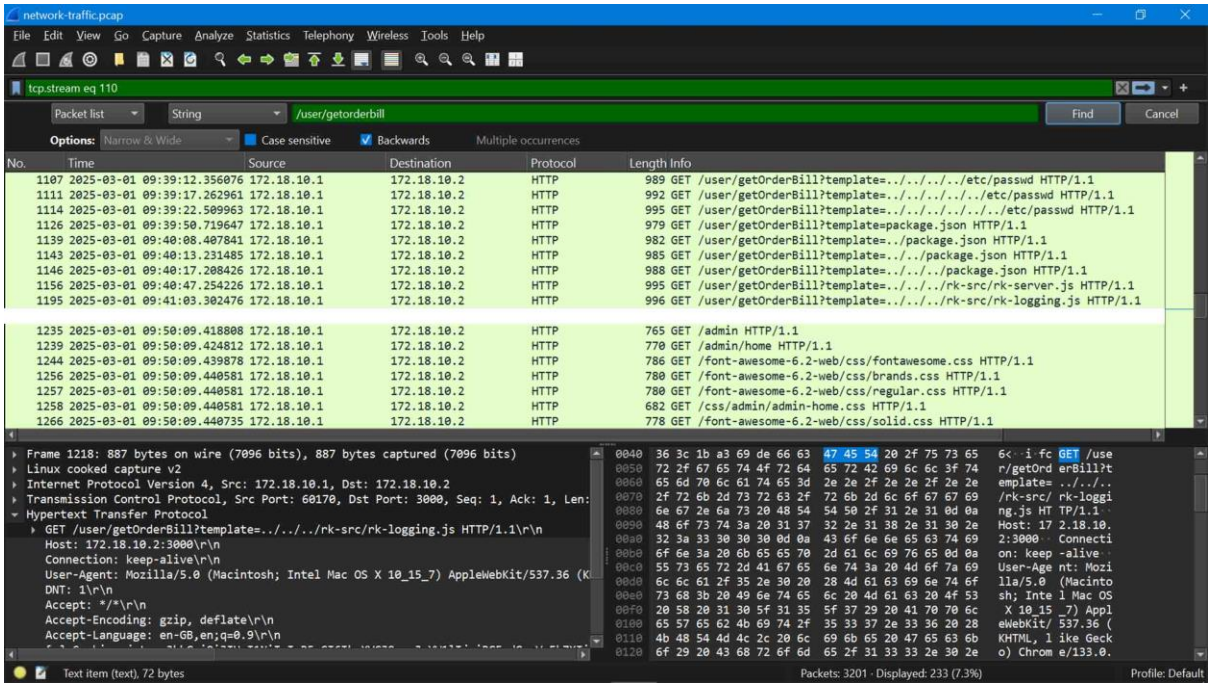
http.request.method == "GET".

I systematically reviewed the filtered requests, working from the bottom of the list—a habit that often reveals less common and potentially overlooked endpoints.

The Finding:

This process quickly revealed a suspicious GET request:

[redacted]



The request immediately stood out for one critical reason: [redacted].

The template parameter was accepting a file path.

The use of [redacted] indicated that the application was likely not sanitizing user input and was directly incorporating the parameter's value into a file system path.

This pattern is a textbook signature of a [redacted], where an attacker can read arbitrary files from the server's file system.

2. When was the first successful exploitation of the vulnerable endpoint by the attacker (time in UTC)?

Answer - [redacted]

During the forensic analysis of the network capture, I initially identified what appeared to be the first exploitation attempt at [redacted]. However, upon deeper investigation, this timestamp represented only the initial probe rather than a successful compromise.

To accurately determine the timeline of exploitation, I followed the complete HTTP stream for the [redacted] endpoint, which revealed a sequence of three distinct attempts:

Attempt 1 - [redacted]

[redacted]

Response: 500 Internal Server Error

Status: ❌ Failed - Incorrect [redacted] depth

Attempt 2 - [redacted]

[redacted]

Response: 500 Internal Server Error

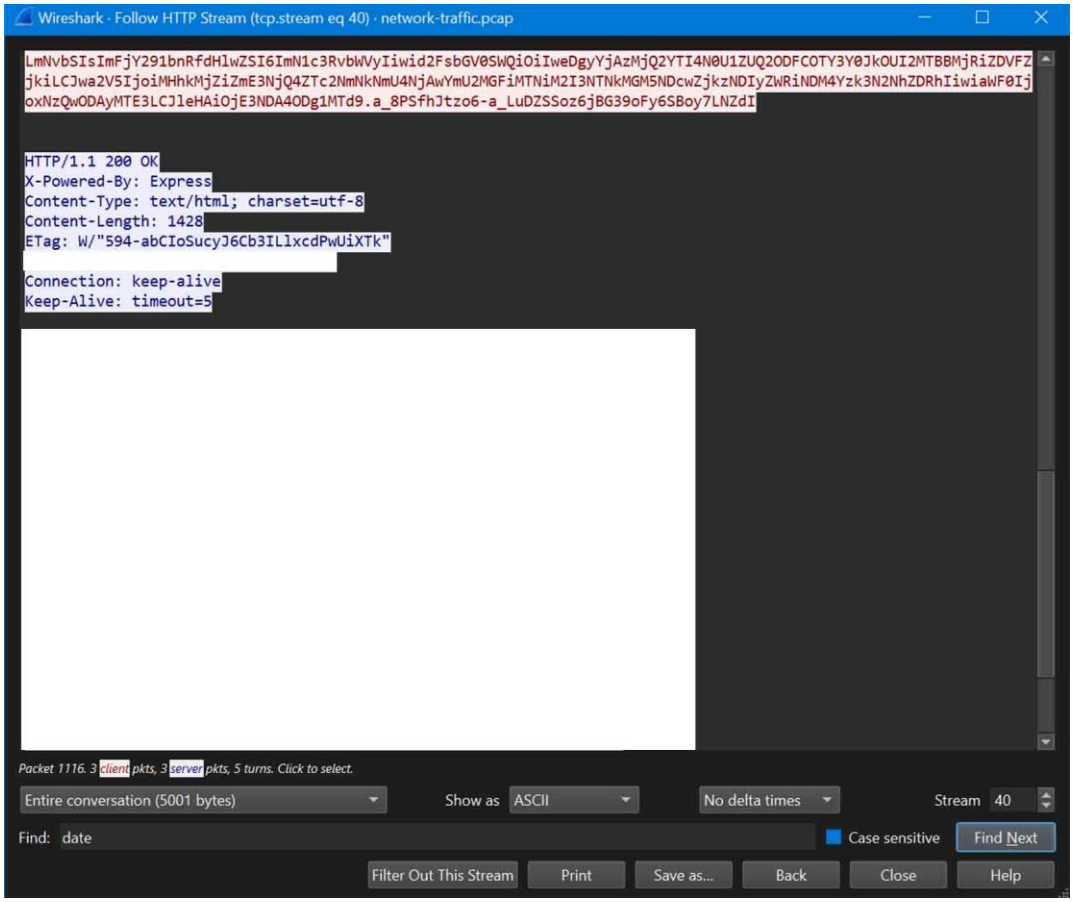
Status: ❌ Failed - Still incorrect depth

Attempt 3 - [redacted]

[redacted]

Response: 200 OK with full [redacted] file content

Status: ✅ First Successful Exploitation



3. Which version of Express is currently being used on the server?

Answer - [redacted]

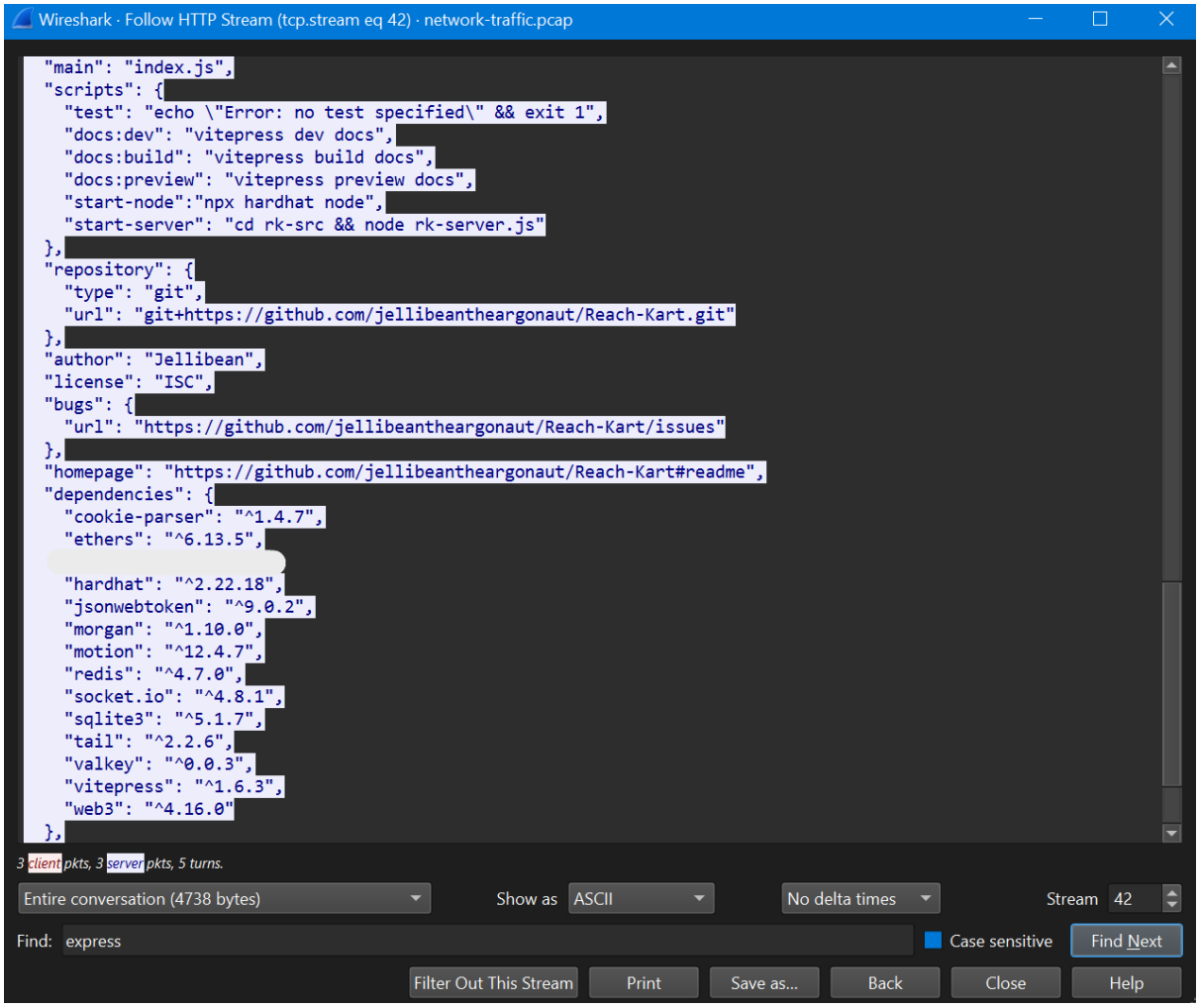
Express.js is a web application framework for Node.js that is commonly used to build web servers and APIs. Identifying the exact version is crucial for vulnerability assessment, as different versions may contain known security issues and CVEs.

Following the successful [redacted] exploitation, I analyzed subsequent HTTP traffic to identify framework information. Express.js often exposes version details in HTTP response headers, particularly in the X-Powered-By header.

Finding

During traffic analysis, I identified a successful HTTP response that contained the framework version information:

The server is running Express.js version [redacted]. This information is valuable for further vulnerability research, as it allows targeted searching for version-specific exploits and known security issues in this release.



4. Which Ethereum compatible development smart contract network is running on the server? (Format: name@version)

Answer - [redacted]

Following the successful [redacted], I analyzed the subsequent attack progression to understand the attacker's reconnaissance methodology. Rather than employing broad network scanning, I focused on the logical next steps an attacker would take after gaining initial file read capabilities.

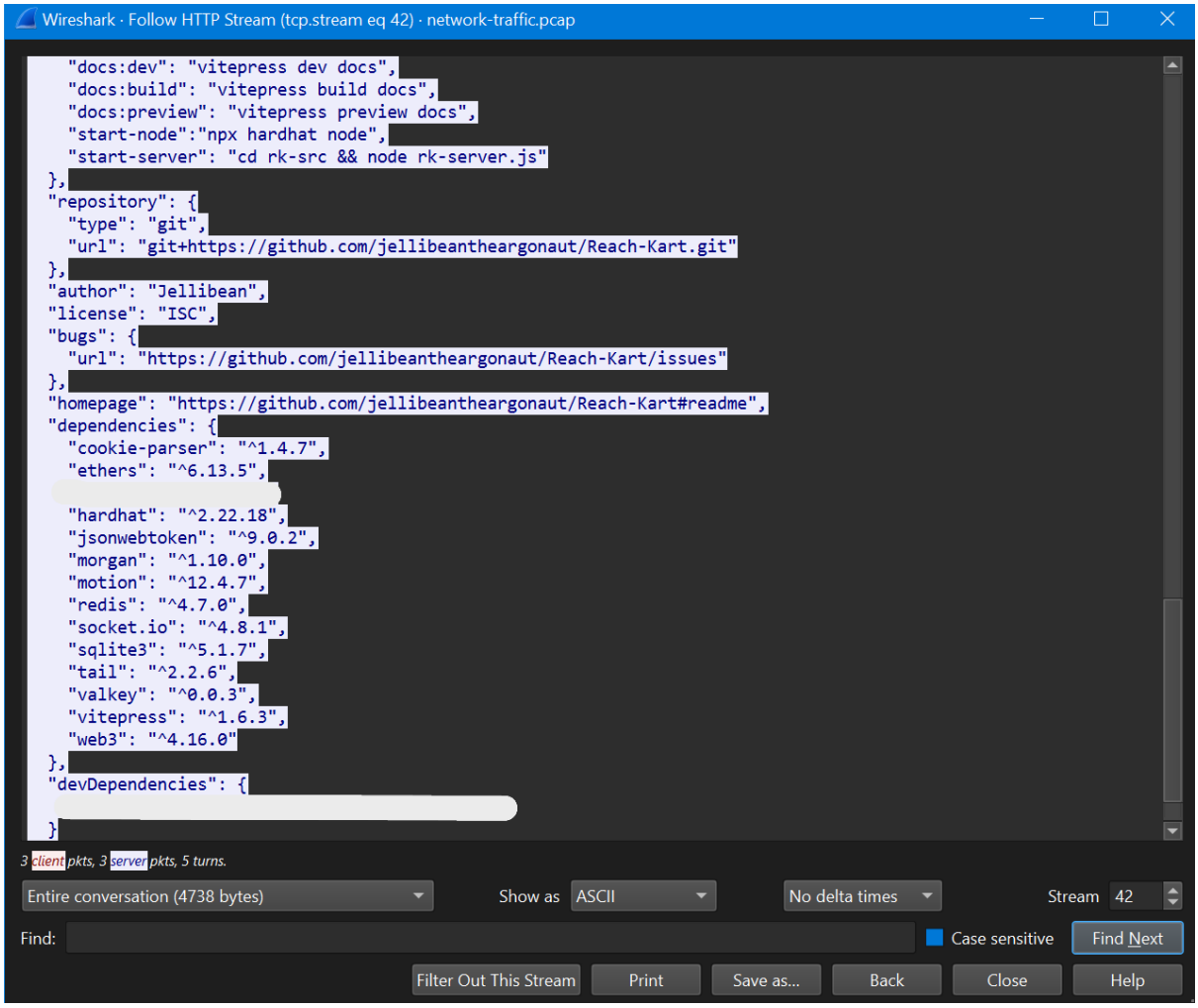
The attack sequence revealed a systematic approach:

Initial [redacted] validation via [redacted] read

Immediate technology stack enumeration via [redacted] extraction

The rapid transition to reading the application's dependency manifest demonstrated the attacker's understanding of modern web application structure and their focus on identifying the technology stack for further exploitation.

The [redacted] file retrieved via LFI revealed key blockchain development dependencies:



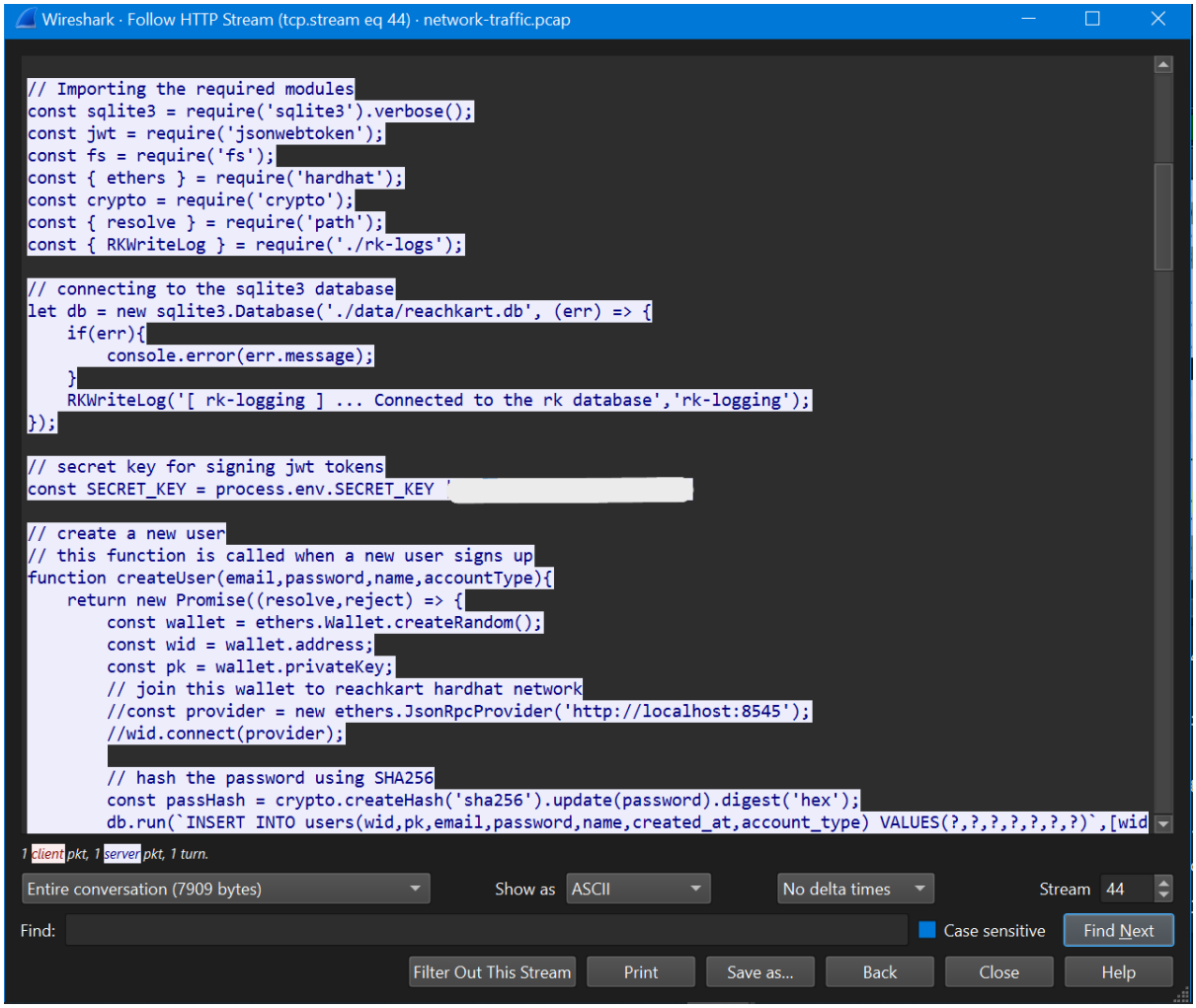
5. What is the signing key used by the server to sign JSON Web Tokens (JWT)?

Answer - [redacted]

Following the successful [redacted] exploitation, I examined the application's core authentication logic in [redacted] to identify the JWT signing mechanism.

Critical Finding

The source code revealed a hardcoded JWT secret key with inadequate fallback security:



6. The attacker was able to generate a JWT from the signing key and log in to the admin panel. What is the JWT value?

Answer -

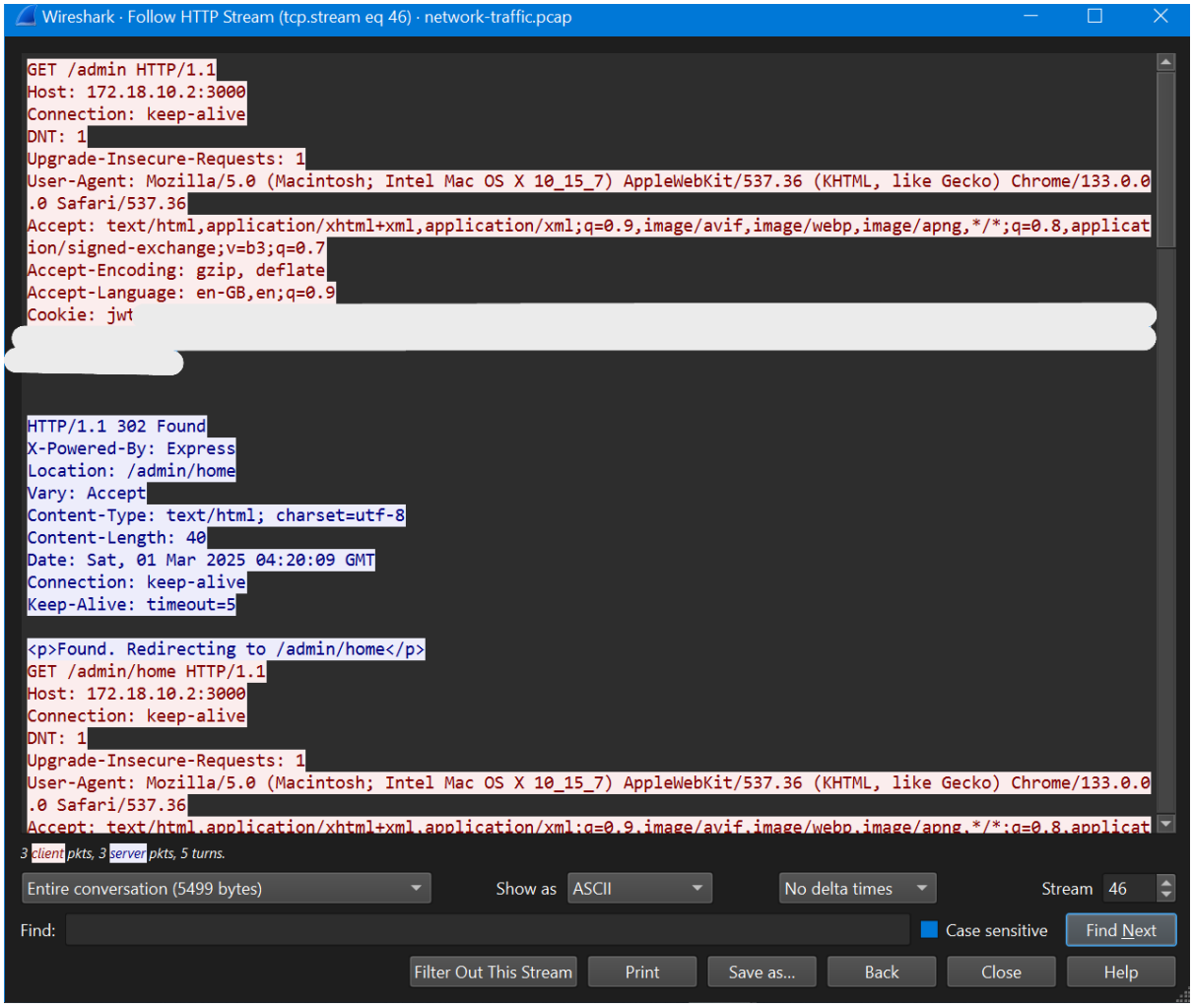
[redacted]

Following the extraction of the JWT signing key, the attacker forged an administrative token to gain unauthorized access to the admin panel.

HTTP 302 Redirect from /admin to /admin/home (authentication successful)

HTTP 200 OK response from /admin/home (full admin access granted)

Subsequent requests using the same token for admin resources



7. Decode the token and find the email used by the attacker to log in to the admin panel.

Answer - [redacted]

Using https://www.jwt.io/ to decode the jwt value get the follwing informartion.

The decoded token reveals the attacker used the email address [redacted] to authenticate to the admin panel. This identity, combined with the forged admin account type, granted full administrative privileges to the application.

8. The admin panel uses WebSocket to send and receive terminal input. What port is being used?

Answer - [redacted]

During analysis of the admin panel functionality, I identified WebSocket communications used for the terminal interface. Using Wireshark, I applied the filter websocket to isolate all WebSocket traffic and analyzed the port usage patterns.

The WebSocket communication revealed two primary ports:

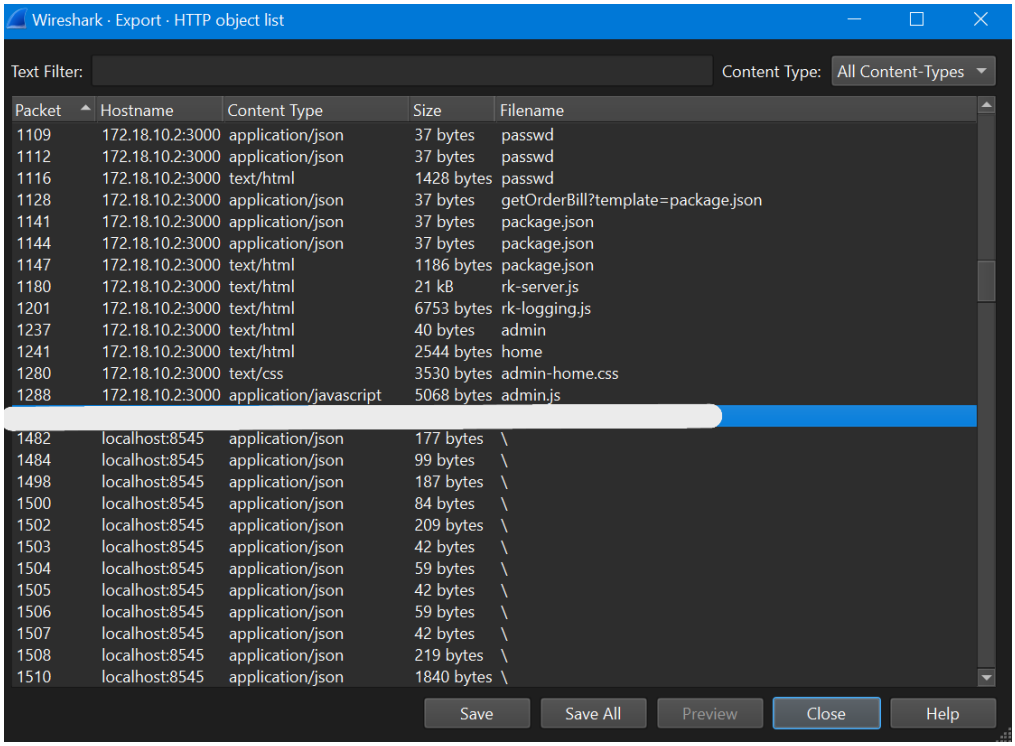
Port [redacted]: Ephemeral client port showing masked frames (client-originated traffic)

Port [redacted]: Persistent service port with unmasked frames (server-originated traffic)

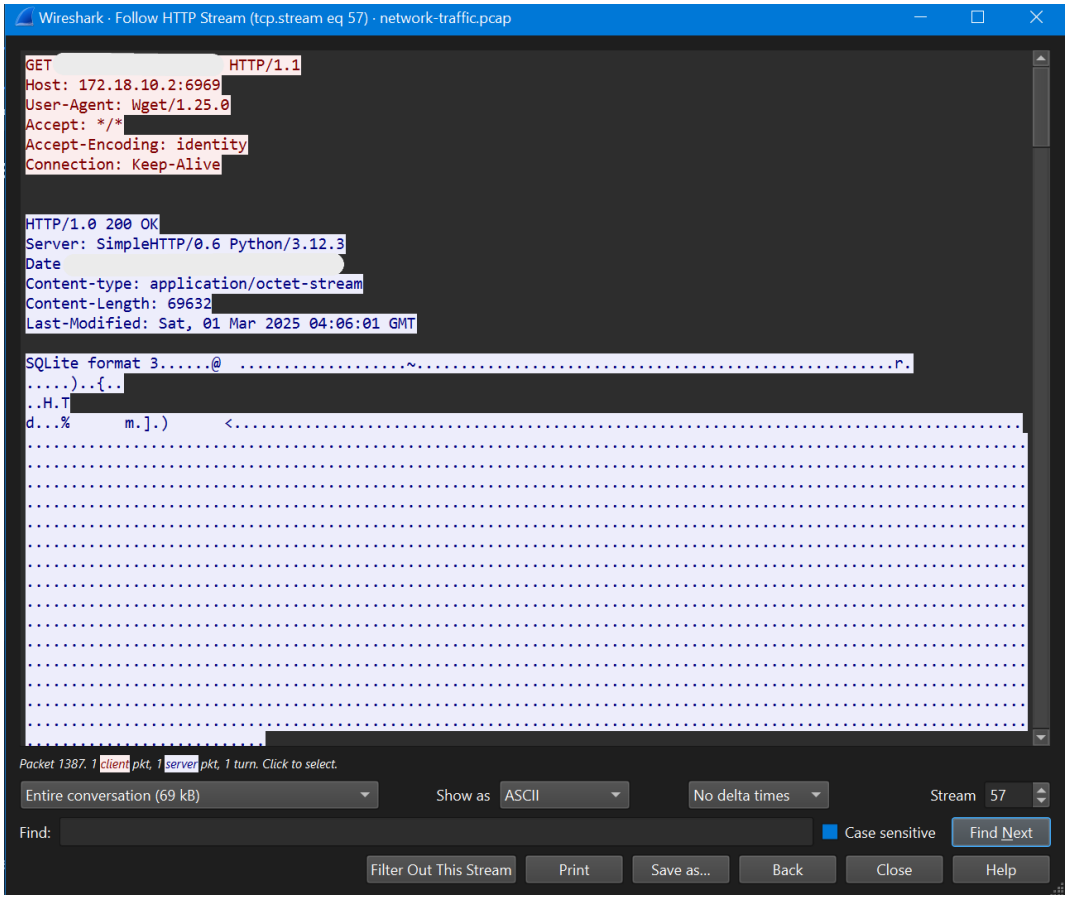
9. The attacker then was able to retrieve a sensitive file. When did the attacker get the file (UTC)?

Answer - [redacted]

During post-compromise traffic analysis, I used Wireshark's File → Export Objects → HTTP feature to inventory all files transferred.



Further examination of the HTTP stream revealed the attacker successfully downloaded this file at [redacted], bypassing the need for WebSocket terminal access and indicating direct file system access through the previously established [redacted] vulnerability.



10. What is the SHA-256 hash of the file that the attacker downloaded?

Answer - [redacted]

The downloaded [redacted] file was extracted from the network capture via Wireshark's HTTP object export functionality.

Wireshark HTTP object analysis identified the [redacted] file transfer.

File exported directly from network stream preserving original content

Cryptographic hash calculated using industry-standard PowerShell cmdlet.

NOTE: Screenshot evidence was intentionally limited to the first 10 investigation steps to demonstrate core forensic methodologies. Subsequent findings rely on extracted data analysis and textual evidence to maintain focus on attack pattern analysis rather than repetitive visual documentation.

11. How many sellers are there in the e-commerce website?

Answer - [redacted]

Following the successful extraction of [redacted], I conducted direct database analysis using DB Browser for SQLite to enumerate user accounts and privilege levels.

Analysis of the users table revealed:

Total Users: [redacted]

Seller Accounts: [redacted] users with account_type = 'seller'

Customer Accounts: 21 users with account_type = 'customer'

SQL Query Verification

SELECT COUNT(*) FROM users WHERE account_type = 'seller';

-- Count(*): [redacted]

12. The attacker started sending Ether from all identified sellers' wallets. What is the hash of the first transaction?

Answer - [redacted]

Following the successful database exfiltration at 04:22:01 UTC, the attacker rapidly escalated from data theft to direct financial gain. The entire process from database compromise to first financial transaction took approximately 4 minutes and 39 seconds, demonstrating highly efficient attack automation.

First Transaction Identification

The initial cryptocurrency theft occurred via a raw signed transaction at 2025-03-01 04:26:40 UTC, identified as the first [redacted] call following the database breach.

What was the total amount of Ether stolen by the attacker? (1 Eth = 10^{18} wei)

Answer - [redacted]

Analysis of the compromised [redacted] database revealed exactly [redacted] seller accounts in the users table, each with associated Ethereum wallet addresses. The account_type field clearly identified these as seller accounts, and the wid (wallet ID) field contained the Ethereum addresses that were subsequently targeted.

All [redacted] seller wallet addresses identified in the database were systematically drained in the cryptocurrency theft campaign, confirming the direct link between the database compromise and financial impact.

Value

[redacted]

13. What is the block number of the last transaction in which Ether was stolen? (Decimal)

Answer - [redacted]

The cryptocurrency theft campaign concluded with the final transaction at 2025-03-01 04:32:25 UTC, targeting the last remaining seller wallet.

14. What was the balance in the attacker's wallet after stealing the Ether?

Answer - [redacted]

The final balance query occurred at 2025-03-01 04:33:01 UTC,

targeting the attacker's wallet address 0x82b03246a287e5ed681b967cbd9b610a24bd5ef9.

The response contained the balance in hexadecimal format requiring conversion:

Raw Balance: 0x1529f07e833d46000 wei

Decimal Conversion: 24,400,230,000,000,000,000 wei

ETH Value: [redacted] ETH (verified using https://www.eth-to-wei.com/)

Formatted Result: [redacted] ETH

This final balance represents the total amount accumulated from all 8 seller wallets,

minus transaction gas fees, confirming the successful execution of the cryptocurrency theft campaign.