Project Report: Network Traffic Analysis Using Wireshark

Date: November 20, 2025

Prepared By: Ayushman Das

Subject: Analysis of Network Packets and Protocol Identification

1. Summary

This report details the procedures and findings from a network traffic analysis exercise conducted using the packet sniffing tool, Wireshark. The objective was to capture live network traffic, filter data based on specific communication protocols, and analyze the structure of data packets. The capture session lasted approximately **88 seconds**, recording over **1,200 packets**. The analysis successfully identified a mix of IPv4 and IPv6 traffic, modern encrypted transport protocols (QUIC, TLS 1.3), and local network discovery mechanisms (SSDP, MDNS).

2. Objectives

- To establish a working environment for packet capture on a local machine.
- To observe real-time data flow across the active network interface.
- To segregate traffic using display filters (HTTP, DNS, TCP, QUIC).
- To document specific packet details and export findings.

3. Methodology and Procedures

The following step-by-step workflow was executed to perform the analysis.

Phase 1: Setup and Acquisition

Step 1: Installation

- **Action:** Wireshark was installed on the host machine.
- **Verification:** Ensured the Npcap driver was active to allow promiscuous mode capturing.

Step 2: Interface Selection and Capture Initiation

- **Action:** Launched Wireshark and identified the active network interface (identified in trace as `\Device\NPF_{82B3B783...}`).
- **Execution:** Started the packet capture process to begin logging ingress and egress traffic.

Phase 2: Traffic Generation

Step 3: Data Stimulation

- **Action:** Network activity was generated by running standard applications.
- **Observed Activity:** Traffic patterns indicate the use of a web browser (Brave), music streaming services (Spotify), and background OS services.

Step 4: Data Collection Limit

- **Constraint:** The capture ran for approximately 1 minute and 28 seconds.
- **Action:** The capture was manually stopped after collecting sufficient data for analysis.

Phase 3: Analysis and Filtering

Step 5: Protocol Filtering

- **Action:** Applied specific display filters to isolate traffic types.
- **Filters Used:**
  - `dns` - To view domain resolution queries.
  - `tls` - To inspect encrypted handshake information.
  - `quic` - To observe modern UDP-based web transport.
  - `ssdp` - To analyze local device discovery.

Step 6: Protocol Identification

- **Observation:** Scanned the "Protocol" column in the packet list pane.
- **Requirement:** Identified distinct protocols including TCP, UDP, QUIC, TLSv1.2, TLSv1.3, ARP, ICMPv6, and SSDP.

Phase 4: Documentation

Step 7: Data Export

- **Action:** Saved the captured session state.
- **Format:** Exported as a `.pcap` file and parsed into Plain Text for detailed inspection.

4. Findings and Technical Observations

4.1 Protocol Analysis

The captured data reveals a modern, diverse network environment. Key findings include:

- **QUIC (Quick UDP Internet Connections):**
  - **Observation:** A significant volume of traffic was identified as QUIC (over UDP port 443). This indicates modern web browsing or streaming activity.
  - **Specifics:** Connection IDs (e.g., `DCID=e728c152c61dc649`) and cryptographic handshakes were visible.
- **TLS (Transport Layer Security):**
  - **Observation:** Both **TLS 1.2** and **TLS 1.3** were observed.
  - **Detail:** A "Client Hello" packet (Frame 1158) identified the Server Name Indication (SNI) as `collector.bsg.brave.com`, confirming the use of the Brave browser. Another SNI observed was `gae2-spclient.spotify.com`.
- **Local Discovery (SSDP & MDNS):**
  - **Observation:** High frequency of SSDP `NOTIFY * HTTP/1.1` packets from source IP `192.168.29.102`.
  - **Detail:** MDNS queries were observed looking for `_spotify-connect._tcp.local`, indicating a device attempting to cast or connect to Spotify.

4.2 Packet Details

Detailed inspection of individual packets extracted the following network specifics:

- **Network Configuration:**
  - **Host IPv4 Address:** `192.168.29.88` (Intel Wireless Adapter).
  - **Secondary Device IP:** `192.168.29.102` (Skyworth Digital device, likely a TV or Set-Top Box sending SSDP beacons).
  - **Gateway/Router:** `192.168.29.1` .
  - **IPv6 Address:** Observed extensive IPv6 traffic (e.g., `2405:201:a006...`) communicating with Google servers.
- **Key Conversations:**
  - **Spotify Streaming:** Traffic exchanged with `57.144.41.33` and `140.82.114.25` over TCP/TLS.
  - **Google/YouTube:** Traffic to `142.250.182.234` and `www.gstatic.com` via QUIC.
  - **Brave Services:** Connection established with `34.223.167.44` .

5. Risk Assessment & Recommendations

- **Privacy Risk:** The capture contains clear text DNS queries (`push.clients6.google.com` , `spclient.wg.spotify.com`). While payloads are encrypted, a passive observer can deduce which websites and services the user is visiting.
- **Multicast Traffic:** The high volume of SSDP and MDNS traffic (Frames 97-113) generates noise on the network. In a large enterprise environment, this should be segmented to prevent broadcast storms.
- **Recommendation:** Ensure endpoints utilize "Private DNS" (DoH or DoT) to encrypt DNS queries, preventing observers from seeing the domain names being resolved.

6. Conclusion

The project successfully demonstrated the utility of Wireshark for network diagnostics. The analysis confirmed that the host `192.168.29.88` is actively communicating with external services using state-of-the-art encryption protocols (TLS 1.3 and QUIC). Furthermore, the trace revealed the presence of other IoT devices (`192.168.29.102`) on the local subnet, providing a complete picture of the current network environment.