# Task 5: Capture and Analyze Network Traffic Using Wireshark

**Objective:** Capture live network packets and identify basic protocols and traffic types.

**Tools Used:** Wireshark (Free, Open Source)

## Method (Steps Taken):

1. Downloaded and installed Wireshark from the official website

(https://www.wireshark.org/download.html).

2. Launched Wireshark and reviewed the available network interfaces (Wi-Fi, Ethernet, Loopback).

3. Selected the active network interface (Wi-Fi) to ensure that only live internet traffic was captured.

4. Clicked the blue shark fin icon to start the packet capture process.

5. Generated traffic for analysis by opening websites in a browser (HTTP/HTTPS traffic) and running ping tests (ping google.com) in the terminal.

6. Allowed the capture to run for approximately 1 minute to gather a variety of traffic types.

7. Stopped the capture by pressing the red stop button once sufficient packets had been recorded.

8. Applied protocol filters such as 'dns', 'tcp', 'icmp', and 'http' in the Wireshark display filter bar to isolate specific traffic types.

9. Observed and analyzed the packet flows, including DNS lookups, TCP handshakes, and ICMP ping requests/replies.

10. Noted key details of different protocols such as source/destination IP addresses, port numbers, and packet lengths.

11. Saved the entire capture in. pcap format using File → Save As, which is a standard format for packet captures.

12. Documented findings including identified protocols (DNS, TCP, ICMP, SMB, ARP, Routing) and their role in network communication.

## Findings (Initial capture):

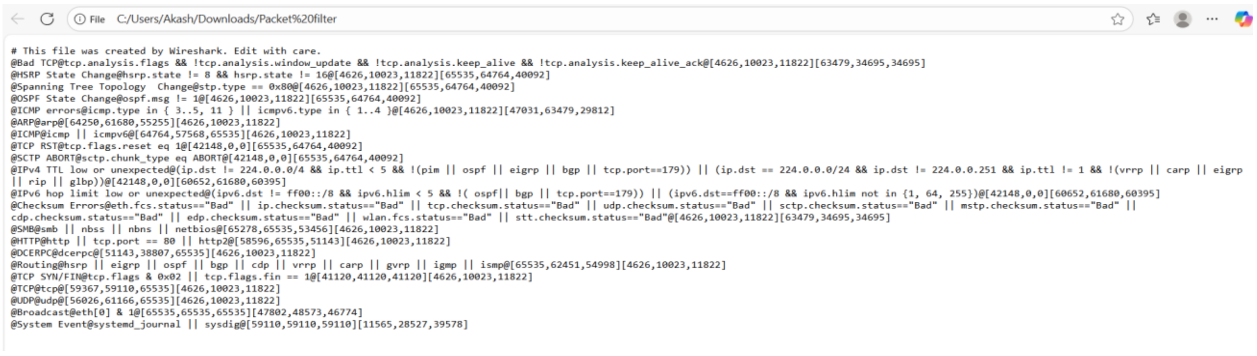
| Protocol | Description |
|---|---|
| DNS | Resolves domain names into IP address |
| TCP | Provides reliable communication in 3-way handshake |
| ICMP | Used for network diagnostics like ping request (Echo) and response (Reply) recorded. |

## Protocols observed from. PCap :

| Protocol | Description |
|---|---|
| System Event | Log from systemd/journald or syslog |
| TCP SYN/FIN | TCP handshake and connection termination flags |
| UDP | Connectionless communication protocol |
| SMB | Server Message Block, used for file/printer sharing in windows |
| DCE/RPC | Remote procedure call protocol over networks |
| ARP | Address resolution protocol, resolves IP to MAC addresses |
| ICMP errors | ICMP used for error reporting(eg., unreachable host) |
| SCTP ABORT/TCP RST | Abort/reset connection packets |
| Routing Protocols | OSPF, BGP, EIGRP, IGMP- routing and group communication |

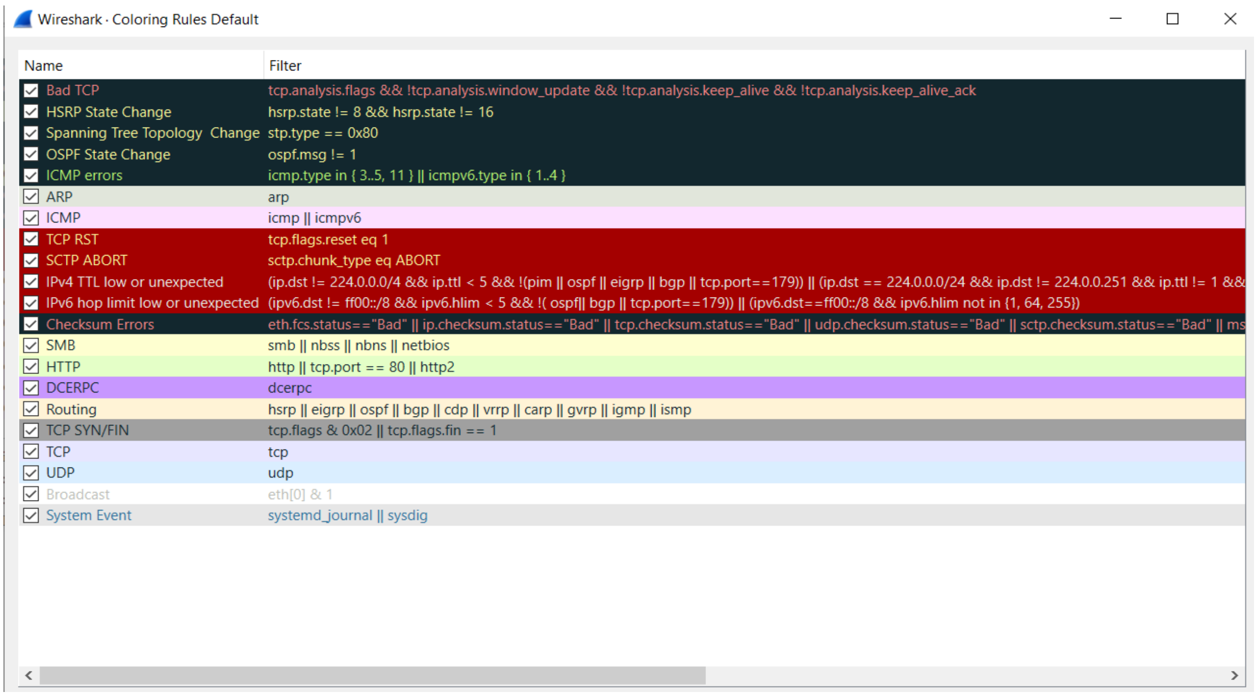
**Outcome:** Successfully captured and analyzed live traffic, identified common and advanced protocols (DNS, TCP, ICMP, SMB, ARP, Routing), and demonstrated filtering and analysis techniques in Wireshark.

## **Screenshot of identifying ports:**



C:/Users/Akash/Downloads/Packet%20filter

```
# This file was created by Wireshark. Edit with care.
@Bad TCP@tcp.analysis.flags && !tcp.analysis.window_update && !tcp.analysis.keep_alive && !tcp.analysis.keep_alive_ack@[4626,10023,11822][63479,34695,34695]
@HSRP State Change@hsrp.state != 8 && hsrp.state != 16@[4626,10023,11822][65535,64764,40092]
@Spanning Tree Topology  Change@stp.type == 0x80@[4626,10023,11822][65535,64764,40092]
@OSPF State Change@ospf.msg != 1@[4626,10023,11822][65535,64764,40092]
@ICMP errors@icmp.type in { 3..5, 11 } || icmpv6.type in { 1..4 }@[4626,10023,11822][47031,63479,29812]
@ARP@arp@[64250,61680,55255][4626,10023,11822]
@ICMP@icmp || icmpv6@[64764,57568,65535][4626,10023,11822]
@TCP RST@tcp.flags.reset eq 1@[42148,0,0][65535,64764,40092]
@SCTP ABORT@sctp.chunk_type eq ABORT@[42148,0,0][65535,64764,40092]
@IPv4 TTL low or unexpected@(ip.dst != 224.0.0.0/4 && ip.ttl < 5 && !(pim || ospf || eigrp || bgp || tcp.port==179)) || (ip.dst == 224.0.0.0/24 && ip.dst != 224.0.0.251 && ip.ttl != 1 && !(vrrp || carp || eigrp
|| rip || glbp))@[42148,0,0][60652,61680,60395]
@IPv6 hop limit low or unexpected@(ipv6.dst != ff00::/8 && ipv6.hlim < 5 && !( ospf|| bgp || tcp.port==179)) || (ipv6.dst==ff00::/8 && ipv6.hlim not in {1, 64, 255})@[42148,0,0][60652,61680,60395]
@Checksum Errors@eth.fcs.status=="Bad" || ip.checksum.status=="Bad" || tcp.checksum.status=="Bad" || udp.checksum.status=="Bad" || sctp.checksum.status=="Bad" || mstp.checksum.status=="Bad" ||
cdp.checksum.status=="Bad" || edp.checksum.status=="Bad" || wlan.fcs.status=="Bad" || stt.checksum.status=="Bad"@[4626,10023,11822][63479,34695,34695]
@SMB@smb || nbss || nbns || netbios@[65278,65535,53456][4626,10023,11822]
@HTTP@http || tcp.port == 80 || http2@[58596,65535,51143][4626,10023,11822]
@DCERPC@dcerpc@[51143,38807,65535][4626,10023,11822]
@Routing@hsrp || eigrp || ospf || bgp || cdp || vrrp || carp || gvrp || igmp || ismp@[65535,62451,54998][4626,10023,11822]
@TCP SYN/FIN@tcp.flags & 0x02 || tcp.flags.fin == 1@[41120,41120,41120][4626,10023,11822]
@TCP@tcp@[59367,59110,65535][4626,10023,11822]
@UDP@udp@[56026,61166,65535][4626,10023,11822]
@Broadcast@eth[0] & 1@[65535,65535,65535][47802,48573,46774]
@System Event@systemd_journal || sysdig@[59110,59110,59110][11565,28527,39578]
```



Wireshark · Coloring Rules Default

| Name | Filter |
|---|---|
| ✓ Bad TCP | tcp.analysis.flags && !tcp.analysis.window_update && !tcp.analysis.keep_alive && !tcp.analysis.keep_alive_ack |
| ✓ HSRP State Change | hsrp.state != 8 && hsrp.state != 16 |
| ✓ Spanning Tree Topology  Change | stp.type == 0x80 |
| ✓ OSPF State Change | ospf.msg != 1 |
| ✓ ICMP errors | icmp.type in { 3..5, 11 } \|\| icmpv6.type in { 1..4 } |
| ✓ ARP | arp |
| ✓ ICMP | icmp \|\| icmpv6 |
| ✓ TCP RST | tcp.flags.reset eq 1 |
| ✓ SCTP ABORT | sctp.chunk_type eq ABORT |
| ✓ IPv4 TTL low or unexpected | (ip.dst != 224.0.0.0/4 && ip.ttl < 5 && !(pim \|\| ospf \|\| eigrp \|\| bgp \|\| tcp.port==179)) \|\| (ip.dst == 224.0.0.0/24 && ip.dst != 224.0.0.251 && ip.ttl != 1 && |
| ✓ IPv6 hop limit low or unexpected | (ipv6.dst != ff00::/8 && ipv6.hlim < 5 && !( ospf\|\| bgp \|\| tcp.port==179)) \|\| (ipv6.dst==ff00::/8 && ipv6.hlim not in {1, 64, 255}) |
| ✓ Checksum Errors | eth.fcs.status=="Bad" \|\| ip.checksum.status=="Bad" \|\| tcp.checksum.status=="Bad" \|\| udp.checksum.status=="Bad" \|\| sctp.checksum.status=="Bad" \|\| ms |
| ✓ SMB | smb \|\| nbss \|\| nbns \|\| netbios |
| ✓ HTTP | http \|\| tcp.port == 80 \|\| http2 |
| ✓ DCERPC | dcerpc |
| ✓ Routing | hsrp \|\| eigrp \|\| ospf \|\| bgp \|\| cdp \|\| vrrp \|\| carp \|\| gvrp \|\| igmp \|\| ismp |
| ✓ TCP SYN/FIN | tcp.flags & 0x02 \|\| tcp.flags.fin == 1 |
| ✓ TCP | tcp |
| ✓ UDP | udp |
| ✓ Broadcast | eth[0] & 1 |
| ✓ System Event | systemd_journal \|\| sysdig |