Course Name:	Computer Communication and Networking	Semester:	VI
Date of Performance:	18 / 03 / 2025	Batch No.:	B - 2
Faculty Name:	Dr. Sudha Gupta	Roll No.:	16014022050
Faculty Sign & Date:		Grade/Marks:	___ / 25

Experiment No: 10

Title: To understand TCP/IP cell using Wireshark software

Aim and Objective of the Experiment:

1. To see all traffic flow over the network.
2. To study of data capturing using sniffing Software Wireshark and analyzing the Transport and Application layer protocol.

COs to be achieved:

CO3: Understand concept of computer communication & Network models.
CO4: To analyses network application and current trends of computer communication and internetworking technology.

Theory:

Wireshark is network protocol analyser for Unix and Windows. It is a free and open-source packet analyzer. It is used for network troubleshooting, analysis, software and communication protocol development. Wireshark is cross-platform. Using the Qt widget toolkit in current releases to implement its user interface, and using pcap to capture packets, it runs on Linux, macOS, Solaris. Some other Unix-like operating systems, and Microsoft Windows,

Wireshark follow link:

- https://www.wireshark.org/download.html
- https://www.youtube.com/watch?v=TkCSr30UojM
- https://www.youtube.com/watch?v=jvuiI1Leg6w

Stepwise-Procedure:

Capturing Packets:
After downloading and installing Wireshark, launch it and click the name of an interface under Interface List to start capturing packets n that interface. For example, to capture traffic on the wireless network, click wireless interface. Configure advanced features by clicking Capture Options.

After clicking interface name. The packets start to appear in real time. Wireshark captures each

packet sent to or from the system

Click the stop capture button near the top left corner of the window to stop capturing traffic.

Color Coding:
Wireshark uses color to help you identify the types of traffic at a glance. By default, green is TCP traffic, dark blue is DNS traffic, light blue is UDP traffic, and black identifies TCP packets with problems – for example, they could have been delivered out of order.

Functionality:
Wireshark is software that "understands" the structure of different networking protocols. Thus, it is able to display the encapsulation and the fields along with their meanings of different packets specified by different networking protocols. Wireshark uses pcap to capture packets, so it can only capture the packets on the types of networks that pcap supports.

Data can be captured "from the wire" from a live network connection or read from a file that recorded already-captured packets.Live data can be read from a number of types of network, including Ethernet, IEEE802.11, PPP, and loopback.Captured network data can be browsed via a GUI, or via the terminal (command line) version of the utility, tshark.Captured files can be programmatically edited or converted via command-line switches to the "editcap" program.Data display can be refined using a display filter. Plug-ins can be created for dissecting new protocols.

To install wireshark:
Sudo install wireshark

To run wireshark:
Wireshark

Select the interface:
Default interface is eth0 for this network.

Grab the packets and see different options available.

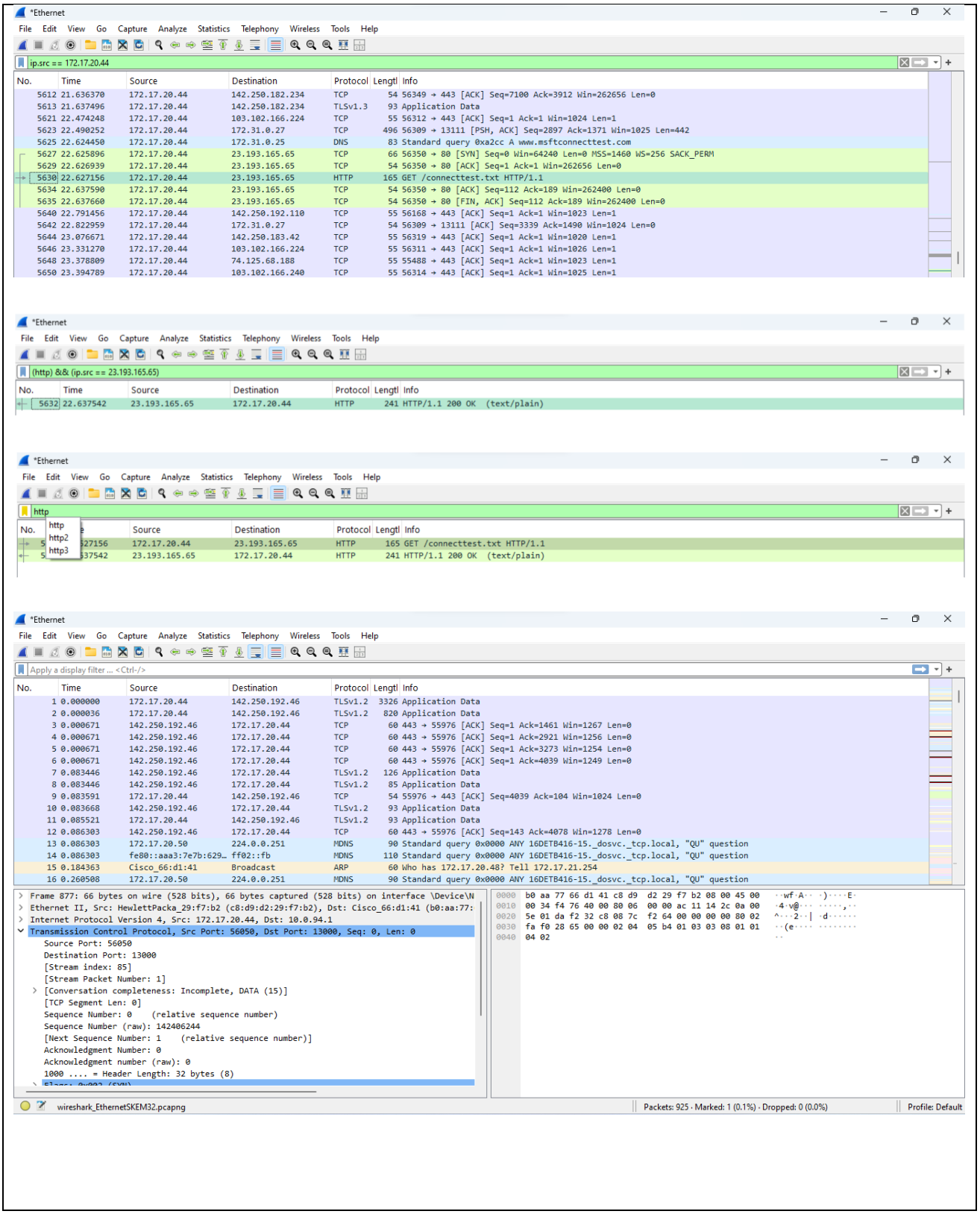
Observation Table:

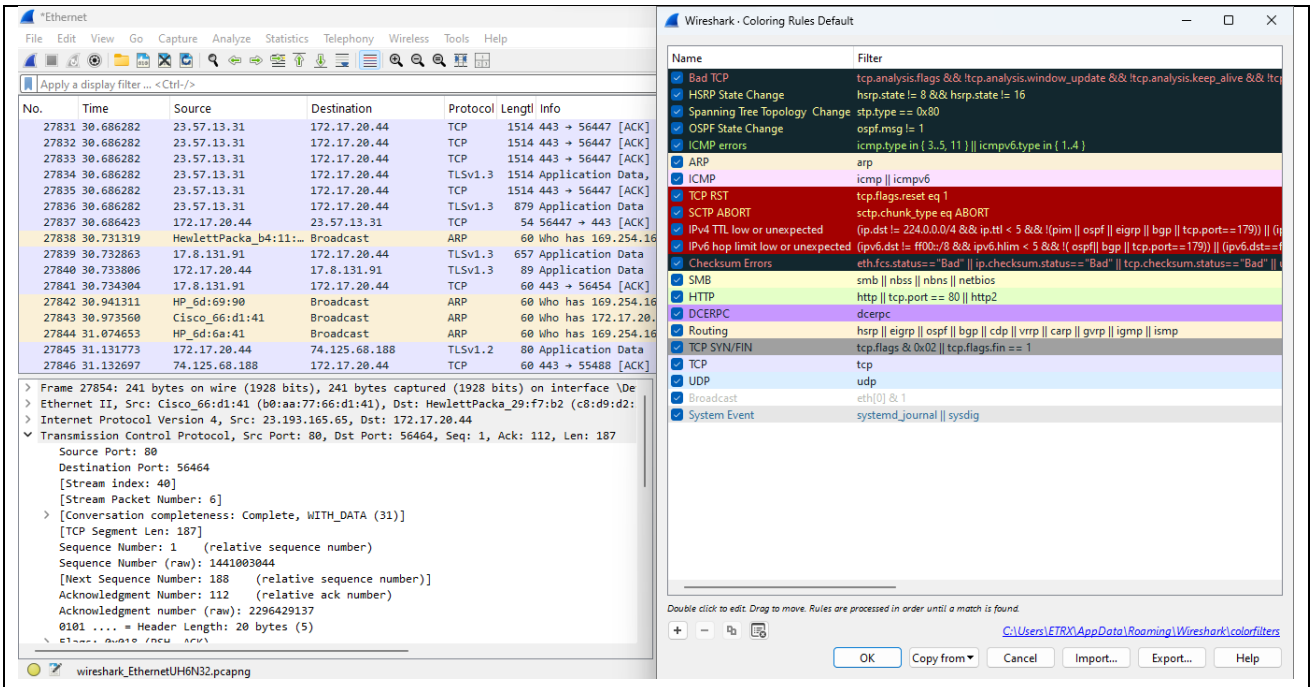
Packet Header Format

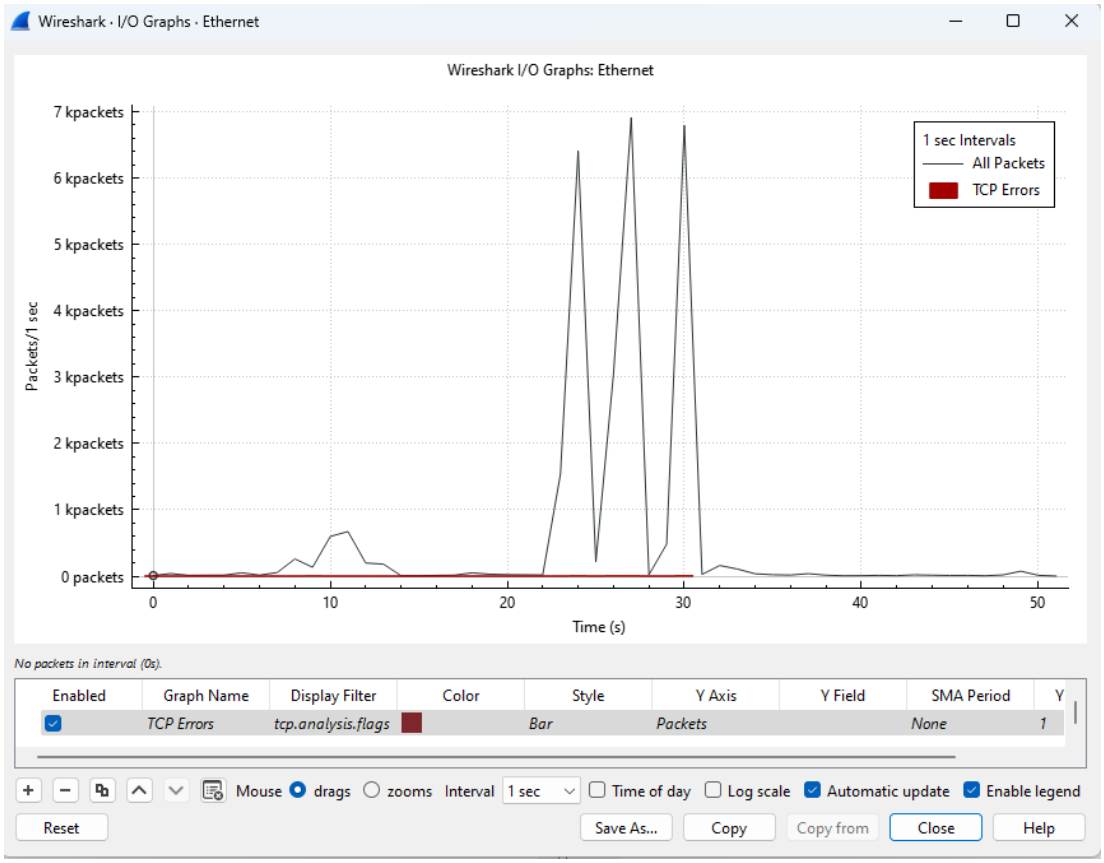
Transport Layer Protocol

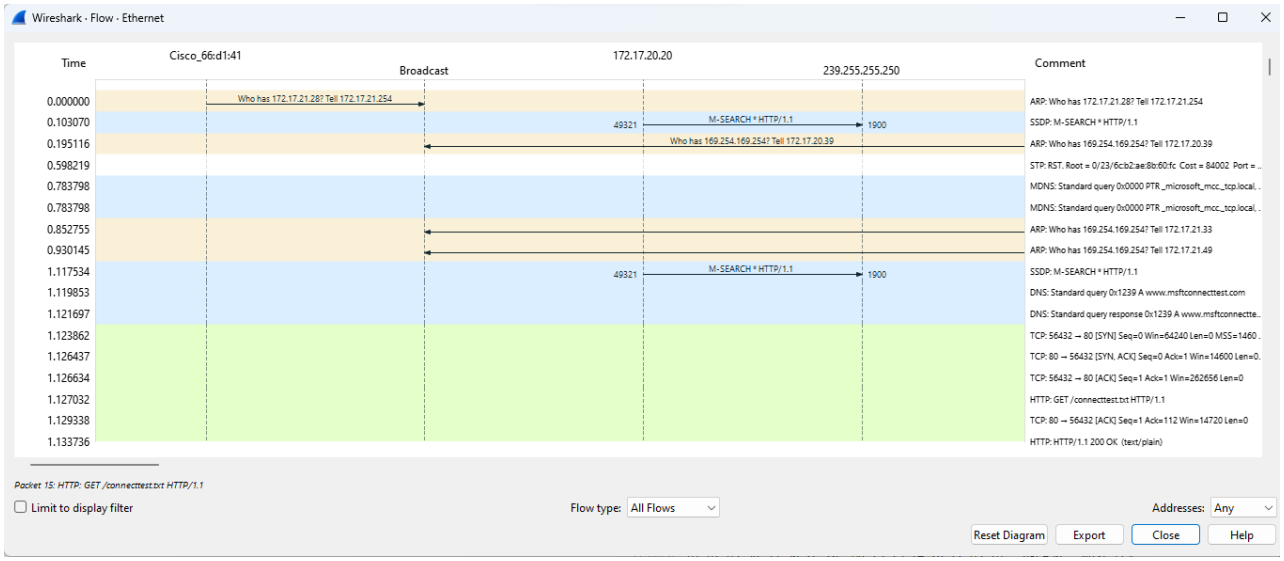
TCP header Format and UDP Header Format

Screenshots:


*Ethernet
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
ip.src == 172.17.20.44
No. Time Source Destination Protocol Lengtl Info
5612 21.636370 172.17.20.44 142.250.182.234 TCP 54 56349 → 443 [ACK] Seq=7100 Ack=3912 Win=262656 Len=0
5613 21.637496 172.17.20.44 142.250.182.234 TLSv1.3 93 Application Data
5621 22.474248 172.17.20.44 103.102.166.224 TCP 55 56312 → 443 [ACK] Seq=1 Ack=1 Win=1024 Len=1
5623 22.490252 172.17.20.44 172.31.0.27 TCP 496 56309 → 13111 [PSH, ACK] Seq=2897 Ack=1371 Win=1025 Len=442
5625 22.624450 172.17.20.44 172.31.0.25 DNS 83 Standard query 0xa2cc A www.msftconnecttest.com
5627 22.625896 172.17.20.44 23.193.165.65 TCP 66 56350 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
5629 22.626939 172.17.20.44 23.193.165.65 TCP 54 56350 → 80 [ACK] Seq=1 Ack=1 Win=262656 Len=0
5630 22.627156 172.17.20.44 23.193.165.65 HTTP 165 GET /connecttest.txt HTTP/1.1
5634 22.637590 172.17.20.44 23.193.165.65 TCP 54 56350 → 80 [ACK] Seq=112 Ack=189 Win=262400 Len=0
5635 22.637660 172.17.20.44 23.193.165.65 TCP 54 56350 → 80 [FIN, ACK] Seq=112 Ack=189 Win=262400 Len=0
5640 22.791456 172.17.20.44 142.250.192.110 TCP 55 56168 → 443 [ACK] Seq=1 Ack=1 Win=1023 Len=1
5642 22.822959 172.17.20.44 172.31.0.27 TCP 54 56309 → 13111 [ACK] Seq=3339 Ack=1490 Win=1024 Len=0
5644 23.076671 172.17.20.44 142.250.183.42 TCP 55 56319 → 443 [ACK] Seq=1 Ack=1 Win=1020 Len=1
5646 23.331270 172.17.20.44 103.102.166.224 TCP 55 56311 → 443 [ACK] Seq=1 Ack=1 Win=1026 Len=1
5648 23.378809 172.17.20.44 74.125.68.188 TCP 55 55488 → 443 [ACK] Seq=1 Ack=1 Win=1023 Len=1
5650 23.394789 172.17.20.44 103.102.166.240 TCP 55 56314 → 443 [ACK] Seq=1 Ack=1 Win=1025 Len=1
*Ethernet
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
(http) && (ip.src == 23.193.165.65)
No. Time Source Destination Protocol Lengtl Info
5632 22.637542 23.193.165.65 172.17.20.44 HTTP 241 HTTP/1.1 200 OK (text/plain)
*Ethernet
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
http
http
http2
http3
No. Source Destination Protocol Lengtl Info
172.17.20.44 23.193.165.65 HTTP 165 GET /connecttest.txt HTTP/1.1
23.193.165.65 172.17.20.44 HTTP 241 HTTP/1.1 200 OK (text/plain)
*Ethernet
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
No. Time Source Destination Protocol Lengtl Info
1 0.000000 172.17.20.44 142.250.192.46 TLSv1.2 3326 Application Data
2 0.000036 172.17.20.44 142.250.192.46 TLSv1.2 820 Application Data
3 0.000671 142.250.192.46 172.17.20.44 TCP 60 443 → 55976 [ACK] Seq=1 Ack=1461 Win=1267 Len=0
4 0.000671 142.250.192.46 172.17.20.44 TCP 60 443 → 55976 [ACK] Seq=1 Ack=2921 Win=1256 Len=0
5 0.000671 142.250.192.46 172.17.20.44 TCP 60 443 → 55976 [ACK] Seq=1 Ack=3273 Win=1254 Len=0
6 0.000671 142.250.192.46 172.17.20.44 TCP 60 443 → 55976 [ACK] Seq=1 Ack=4039 Win=1249 Len=0
7 0.083446 142.250.192.46 172.17.20.44 TLSv1.2 126 Application Data
8 0.083446 142.250.192.46 172.17.20.44 TLSv1.2 85 Application Data
9 0.083591 172.17.20.44 142.250.192.46 TCP 54 55976 → 443 [ACK] Seq=4039 Ack=104 Win=1024 Len=0
10 0.083668 142.250.192.46 172.17.20.44 TLSv1.2 93 Application Data
11 0.085521 172.17.20.44 142.250.192.46 TLSv1.2 93 Application Data
12 0.086303 142.250.192.46 172.17.20.44 TCP 60 443 → 55976 [ACK] Seq=143 Ack=4078 Win=1278 Len=0
13 0.086303 172.17.20.50 224.0.0.251 MDNS 90 Standard query 0x0000 ANY 16DETB416-15._dosvc._tcp.local, "QU" question
14 0.086303 fe80::aaa3:7e7b:629… ff02::fb MDNS 110 Standard query 0x0000 ANY 16DETB416-15._dosvc._tcp.local, "QU" question
15 0.184363 Cisco_66:d1:41 Broadcast ARP 60 Who has 172.17.20.48? Tell 172.17.21.254
16 0.260508 172.17.20.50 224.0.0.251 MDNS 90 Standard query 0x0000 ANY 16DETB416-15._dosvc._tcp.local, "QU" question
Frame 877: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface \Device\N
Ethernet II, Src: HewlettPacka_29:f7:b2 (c8:d9:d2:29:f7:b2), Dst: Cisco_66:d1:41 (b0:aa:77:
Internet Protocol Version 4, Src: 172.17.20.44, Dst: 10.0.94.1
Transmission Control Protocol, Src Port: 56050, Dst Port: 13000, Seq: 0, Len: 0
Source Port: 56050
Destination Port: 13000
[Stream index: 85]
[Stream Packet Number: 1]
[Conversation completeness: Incomplete, DATA (15)]
[TCP Segment Len: 0]
Sequence Number: 0 (relative sequence number)
Sequence Number (raw): 142406244
[Next Sequence Number: 1 (relative sequence number)]
Acknowledgment Number: 0
Acknowledgment number (raw): 0
1000 .... = Header Length: 32 bytes (8)
0000 b0 aa 77 66 d1 41 c8 d9 d2 29 f7 b2 08 00 45 00
0010 00 34 f4 76 40 00 80 06 00 00 ac 11 14 2c 0a 00
0020 5e 01 da f2 32 c8 08 7c f2 64 00 00 00 00 80 02
0030 fa f0 28 65 00 00 02 04 05 b4 01 03 03 08 01 01
0040 04 02
wireshark_EthernetSKEM32.pcapng
Packets: 925 · Marked: 1 (0.1%) · Dropped: 0 (0.0%)
Profile: Default


*Ethernet
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
No. Time Source Destination Protocol Lengtl Info
27831 30.686282 23.57.13.31 172.17.20.44 TCP 1514 443 → 56447 [ACK]
27832 30.686282 23.57.13.31 172.17.20.44 TCP 1514 443 → 56447 [ACK]
27833 30.686282 23.57.13.31 172.17.20.44 TCP 1514 443 → 56447 [ACK]
27834 30.686282 23.57.13.31 172.17.20.44 TLSv1.3 1514 Application Data,
27835 30.686282 23.57.13.31 172.17.20.44 TCP 1514 443 → 56447 [ACK]
27836 30.686282 23.57.13.31 172.17.20.44 TLSv1.3 879 Application Data
27837 30.686423 172.17.20.44 23.57.13.31 TCP 54 56447 → 443 [ACK]
27838 30.731319 HewlettPacka_b4:11:... Broadcast ARP 60 Who has 169.254.16
27839 30.732863 17.8.131.91 172.17.20.44 TLSv1.3 657 Application Data
27840 30.733806 172.17.20.44 17.8.131.91 TLSv1.3 89 Application Data
27841 30.734304 17.8.131.91 172.17.20.44 TCP 60 443 → 56454 [ACK]
27842 30.941311 HP_6d:69:90 Broadcast ARP 60 Who has 169.254.16
27843 30.973560 Cisco_66:d1:41 Broadcast ARP 60 Who has 172.17.20.
27844 31.074653 HP_6d:6a:41 Broadcast ARP 60 Who has 169.254.16
27845 31.131773 172.17.20.44 74.125.68.188 TLSv1.2 80 Application Data
27846 31.132697 74.125.68.188 172.17.20.44 TCP 60 443 → 55488 [ACK]
Frame 27854: 241 bytes on wire (1928 bits), 241 bytes captured (1928 bits) on interface \De
Ethernet II, Src: Cisco_66:d1:41 (b0:aa:77:66:d1:41), Dst: HewlettPacka_29:f7:b2 (c8:d9:d2:
Internet Protocol Version 4, Src: 23.193.165.65, Dst: 172.17.20.44
Transmission Control Protocol, Src Port: 80, Dst Port: 56464, Seq: 1, Ack: 112, Len: 187
Source Port: 80
Destination Port: 56464
[Stream index: 40]
[Stream Packet Number: 6]
[Conversation completeness: Complete, WITH_DATA (31)]
[TCP Segment Len: 187]
Sequence Number: 1 (relative sequence number)
Sequence Number (raw): 1441003044
[Next Sequence Number: 188 (relative sequence number)]
Acknowledgment Number: 112 (relative ack number)
Acknowledgment number (raw): 2296429137
0101 .... = Header Length: 20 bytes (5)
wireshark_EthernetUH6N32.pcapng
Wireshark · Coloring Rules Default
Name Filter
Bad TCP tcp.analysis.flags && !tcp.analysis.window_update && !tcp.analysis.keep_alive && !tcp
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)) ||
IPv6 hop limit low or unexpected (ipv6.dst != ff00::/8 && ipv6.hlim < 5 && !(ospf|| bgp || tcp.port==179)) || (ipv6.dst==
Checksum Errors eth.fcs.status=="Bad" || ip.checksum.status=="Bad" || tcp.checksum.status=="Bad" ||
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
Double click to edit. Drag to move. Rules are processed in order until a match is found.
C:\Users\ETRX\AppData\Roaming\Wireshark\colorfilters
OK Copy from Cancel Import... Export... Help


Wireshark · I/O Graphs · Ethernet
Wireshark I/O Graphs: Ethernet
7 kpackets
6 kpackets
5 kpackets
4 kpackets
3 kpackets
2 kpackets
1 kpackets
0 packets
Packets/1 sec
0
10
20
30
40
50
Time (s)
1 sec Intervals
All Packets
TCP Errors
No packets in interval (0s).
Enabled Graph Name Display Filter Color Style Y Axis Y Field SMA Period Y
TCP Errors tcp.analysis.flags Bar Packets None 1
Mouse drags zooms Interval 1 sec Time of day Log scale Automatic update Enable legend
Reset Save As... Copy Copy from Close Help



Post Lab Subjective/Objective type Questions:

1. **What are the Different types of analysis options provided by Wireshark?**
   Wireshark provides several analysis options to inspect and troubleshoot network traffic. These include packet capture and filtering, where you can capture live network packets and apply display filters to focus on specific data. It offers protocol analysis, showing detailed breakdowns of each protocol layer. Color coding helps differentiate packet types visually. There are statistics tools like protocol hierarchy, I/O graphs, and flow diagrams to analyze traffic patterns. Wireshark also provides expert information, which highlights warnings and potential problems in the captured data, such as retransmissions or malformed packets.

2. **What devices can Wireshark use to capture packets?**
   Wireshark can capture packets from various network interfaces on a computer. These include Ethernet adapters, Wi-Fi interfaces, Bluetooth devices, and virtual network interfaces used by VPNs or virtual machines. On some systems, it can also capture from USB interfaces or loopback interfaces. To capture packets directly from network hardware like switches or routers, devices must support port mirroring or SPAN (Switch Port Analyzer), allowing Wireshark to see all traffic on a network segment.

Conclusion:

The experiment successfully demonstrated the structure and functioning of TCP/IP communication using Wireshark. We analyzed and understood how data packets are transmitted and received in a network.

Signature of faculty in-charge with Date: