

Practical-5

AIM Experiments on Packet capture tool: Wireshark

Packet Sniffer

- Sniffs messages being sent/received from/by your computer
- Store and display the contents of the various protocol fields in the messages
- Passive program
 - never sends packets itself
 - no packets addressed to it
 - receives a copy of all packets (sent/received)

Packet Sniffer Structure Diagnostic Tools

- Tcpdump
 - E.g. tcpdump -enx host 10.129.41.2 -w exe3.out
- Wireshark
 - wireshark -r exe3.out

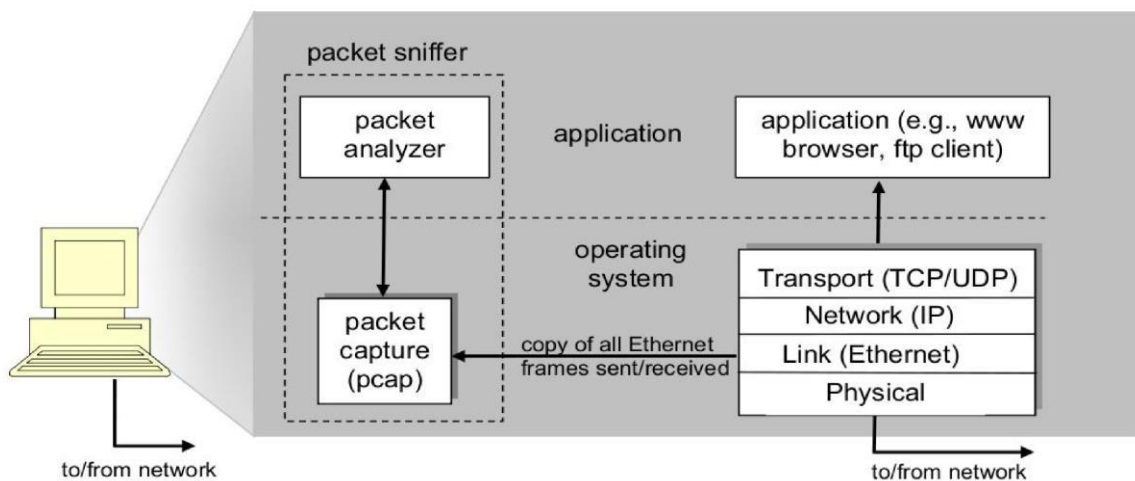


Figure 1: Packet sniffer structure

DESCRIPTION:

WIRESHARK

Wireshark, a network analysis tool formerly known as Ethereal, captures packets in real time and display them in human-readable format. Wireshark includes filters, color coding, and other features that let you dig deep into network traffic and inspect individual packets. You can use Wireshark to inspect a suspicious program's network traffic, analyze the traffic flow on your network, or troubleshoot network problems.

What we can do with Wireshark:

- Capture network traffic
- Decode packet protocols using dissectors
- Define filters – capture and display
- Watch smart statistics
- Analyze problems
- Interactively browse that traffic

Wireshark used for:

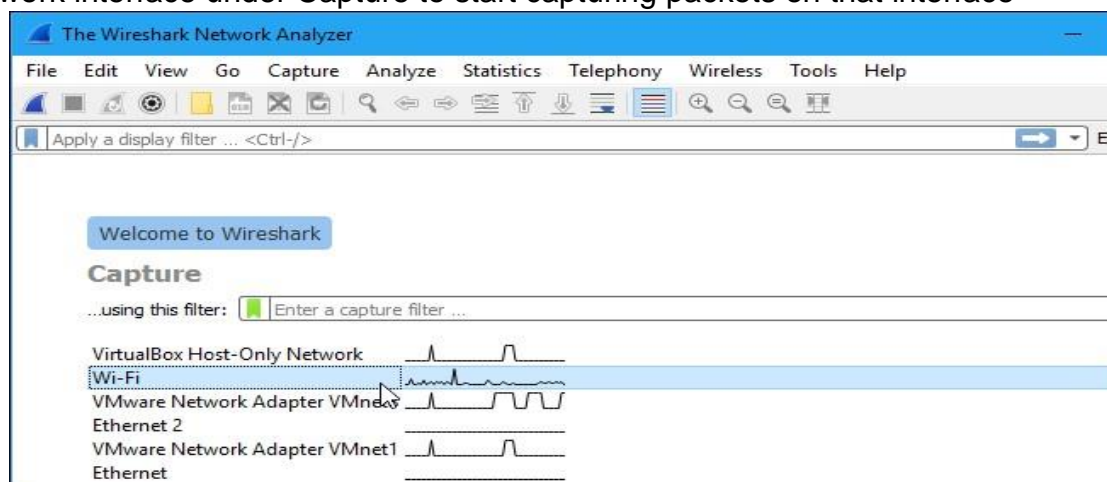
- Network administrators: troubleshoot network problems
- Network security engineers: examine security problems
- Developers: debug protocol implementations
- People: learn **network protocol internals**

Getting Wireshark

Wireshark can be downloaded for Windows or macOS from [its official website](#). For Linux or another UNIX-like system, Wireshark will be found in its package repositories. For Ubuntu, Wireshark will be found in the Ubuntu Software Center.

Capturing Packets

After downloading and installing Wireshark, launch it and double-click the name of a network interface under Capture to start capturing packets on that interface



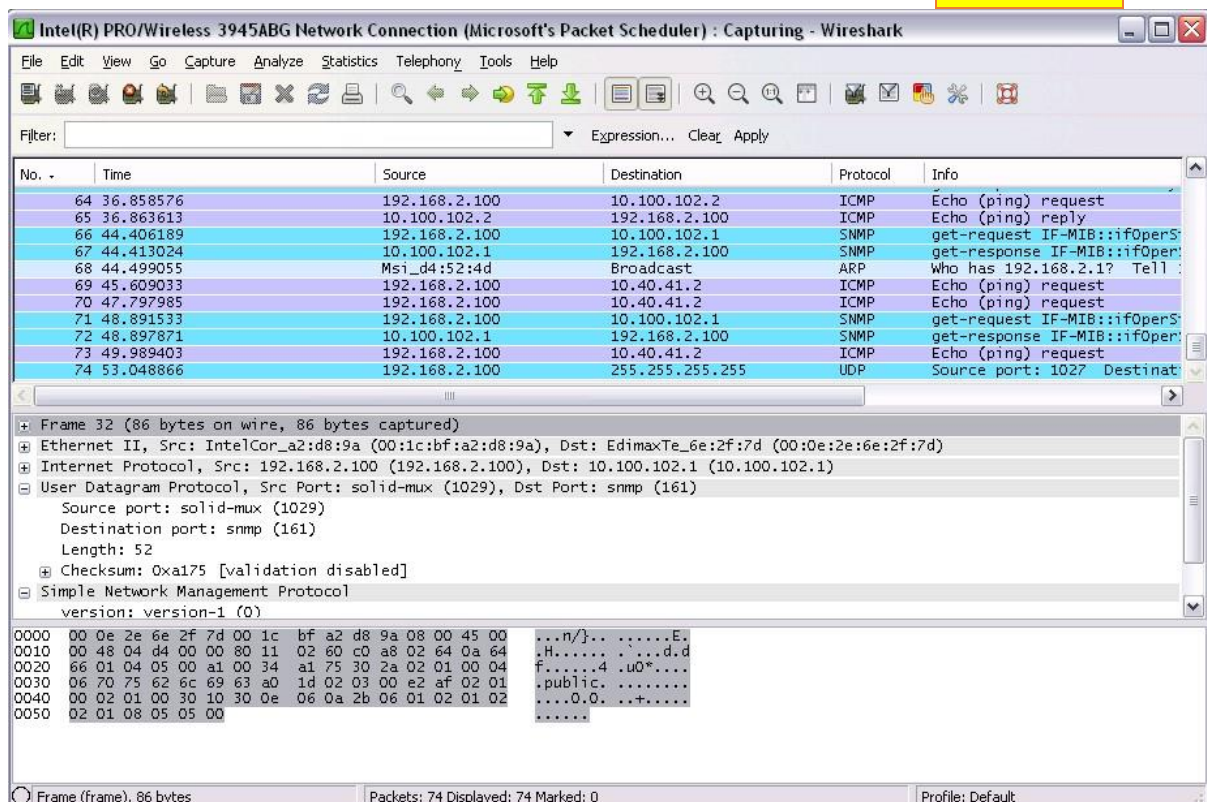
As soon as you click the interface's name, you'll see the packets start to appear in real time. Wireshark captures each packet sent to or from your system.

CS19541-COMPUTER NETWORKS-LAB MANUAL

If you have promiscuous mode enabled—it's enabled by default—you'll also see all the other packets on the network instead of only packets addressed to your network adapter. To check if promiscuous mode is enabled, click Capture > Options and verify the "Enable promiscuous mode on all interfaces" checkbox is activated at the bottom of this window.

Packet
List

Packet
Details



The image shows a Wireshark packet capture window titled "Intel(R) PRO/Wireless 3945ABG Network Connection (Microsoft's Packet Scheduler) : Capturing - Wireshark". The window displays a list of captured packets in the top pane and the details of the selected packet (Frame 32) in the bottom pane.

Packet List:

No.	Time	Source	Destination	Protocol	Info
64	36.858576	192.168.2.100	10.100.102.2	ICMP	Echo (ping) request
65	36.863613	10.100.102.2	192.168.2.100	ICMP	Echo (ping) reply
66	44.406189	192.168.2.100	10.100.102.1	SNMP	get-request IF-MIB::ifOperS
67	44.413024	10.100.102.1	192.168.2.100	SNMP	get-response IF-MIB::ifOperS
68	44.499055	Msi_d4:52:4d	Broadcast	ARP	Who has 192.168.2.1? Tell
69	45.609033	192.168.2.100	10.40.41.2	ICMP	Echo (ping) request
70	47.797985	192.168.2.100	10.40.41.2	ICMP	Echo (ping) request
71	48.891533	192.168.2.100	10.100.102.1	SNMP	get-request IF-MIB::ifOperS
72	48.897871	10.100.102.1	192.168.2.100	SNMP	get-response IF-MIB::ifOperS
73	49.989403	192.168.2.100	10.40.41.2	ICMP	Echo (ping) request
74	53.048866	192.168.2.100	255.255.255.255	UDP	Source port: 1027 Destinat

Packet Details (Frame 32):

- Frame 32 (86 bytes on wire, 86 bytes captured)
- Ethernet II, Src: IntelCor_a2:d8:9a (00:1c:bf:a2:d8:9a), Dst: EdimaxTe_6e:2f:7d (00:0e:2e:6e:2f:7d)
- Internet Protocol, Src: 192.168.2.100 (192.168.2.100), Dst: 10.100.102.1 (10.100.102.1)
- User Datagram Protocol, Src Port: solid-mux (1029), Dst Port: snmp (161)
 - Source port: solid-mux (1029)
 - Destination port: snmp (161)
 - Length: 52
 - Checksum: 0xa175 [validation disabled]
- Simple Network Management Protocol
 - version: version-1 (0)

Packet Data (Hex and ASCII):

```
0000 00 0e 2e 6e 2f 7d 00 1c bf a2 d8 9a 08 00 45 00 ...n/)...E.
0010 00 48 04 d4 00 00 80 11 02 60 c0 a8 02 64 0a 64 .H.....d.d
0020 66 01 04 05 00 a1 00 34 a1 75 30 2a 02 01 00 04 f.....4.u0*...
0030 06 70 75 62 6c 69 63 a0 1d 02 03 00 e2 af 02 01 .public.....
0040 00 02 01 00 30 10 30 0e 06 0a 2b 06 01 02 01 02 ....0.0...+....
0050 02 01 08 05 05 00 .....
```

Click the red “Stop” button near the top left corner of the window when you want to stop capturing traffic.

The “Packet List” Pane

The packet list pane displays all the packets in the current capture file. The “Packet List” pane Each line in the packet list corresponds to one packet in the capture file. If you select a line in this pane, more details will be displayed in the “Packet Details” and “Packet Bytes” panes.

The “Packet Details” Pane

The packet details pane shows the current packet (selected in the “Packet List” pane) in a more detailed form. This pane shows the protocols and protocol fields of the packet selected in the “Packet List” pane. The protocols and fields of the packet shown in a tree which can be expanded and collapsed.

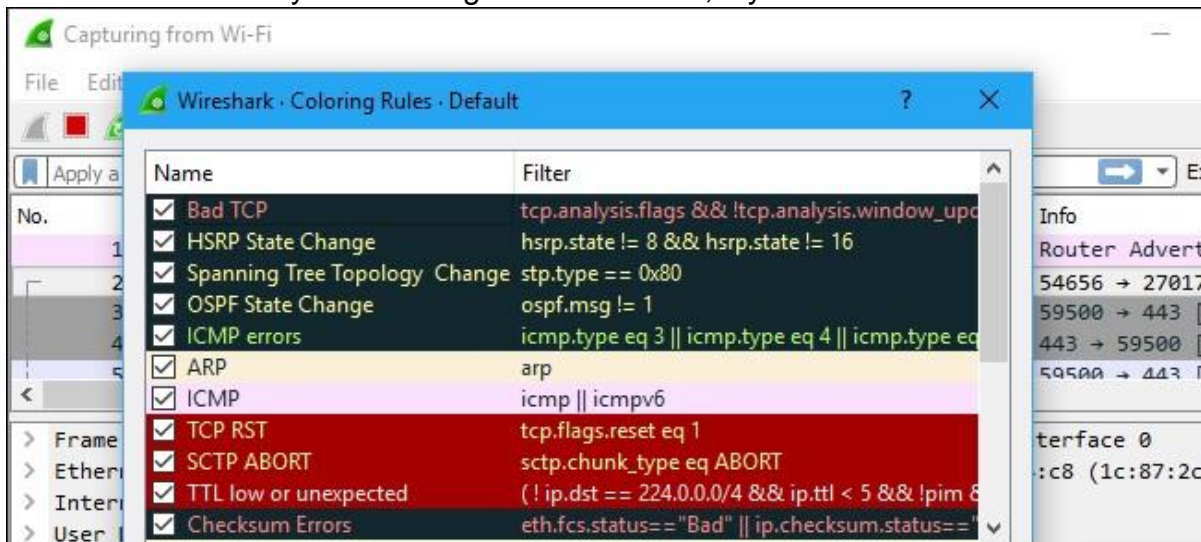
The “Packet Bytes” Pane

The packet bytes pane shows the data of the current packet (selected in the “Packet List” pane) in a hexdump style.

Color Coding

You’ll probably see packets highlighted in a variety of different colors. Wireshark uses colors to help you identify the types of traffic at a glance. By default, light purple is TCP traffic, light blue is UDP traffic, and black identifies packets with errors—for example, they could have been delivered out of order.

To view exactly what the color codes mean, click View > Coloring Rules. You can also customize and modify the coloring rules from here, if you like.

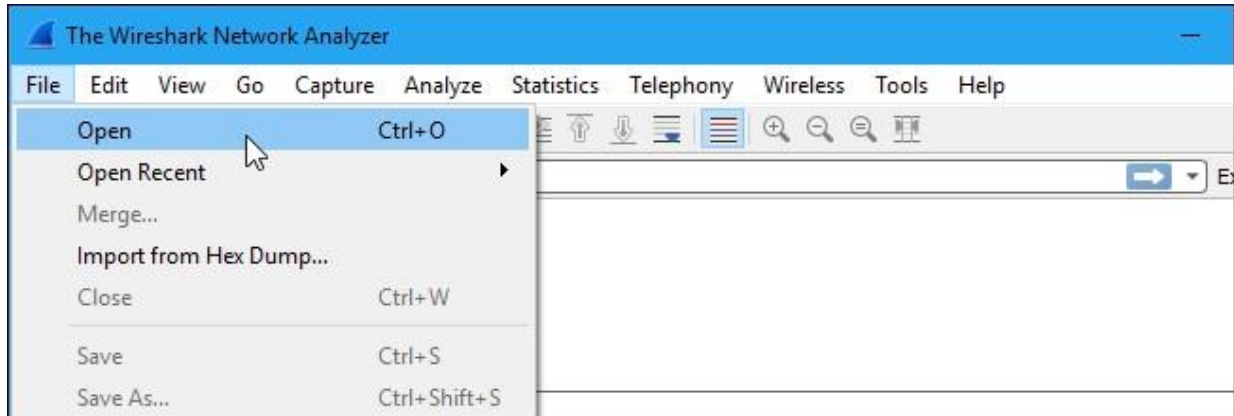


Sample Captures

If there’s nothing interesting on your own network to inspect, Wireshark’s wiki has you covered. The wiki contains a [page of sample capture files](#) that you can load and

inspect. Click File > Open in Wireshark and browse for your downloaded file to open one.

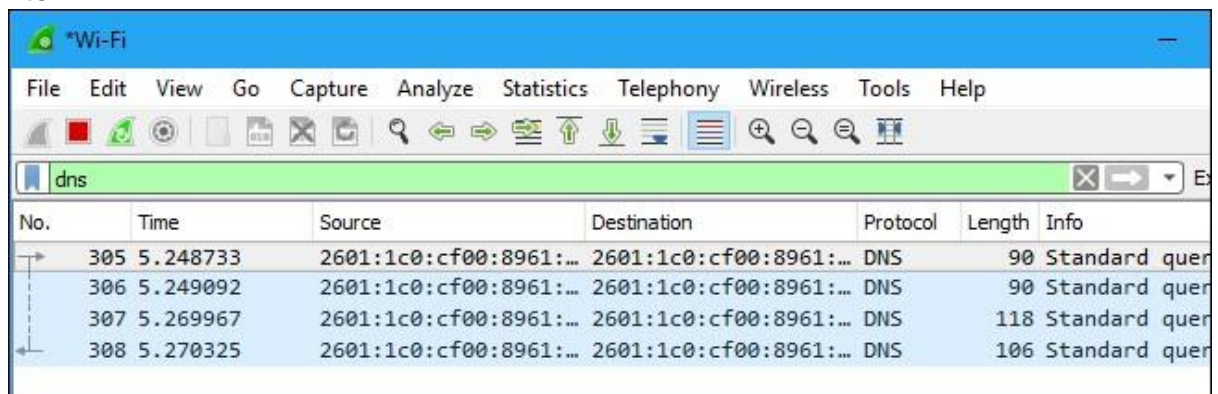
You can also save your own captures in Wireshark and open them later. Click File > Save to save your captured packets.



Filtering Packets

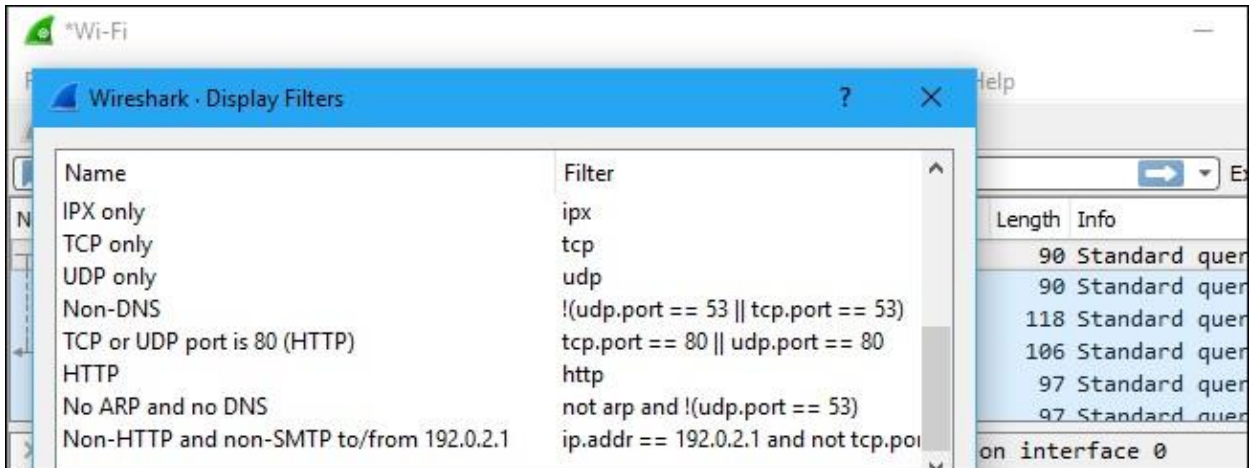
If you're trying to inspect something specific, such as the traffic a program sends when phoning home, it helps to close down all other applications using the network so you can narrow down the traffic. Still, you'll likely have a large amount of packets to sift through. That's where Wireshark's filters come in.

The most basic way to apply a filter is by typing it into the filter box at the top of the window and clicking Apply (or pressing Enter). For example, type "dns" and you'll see only DNS packets. When you start typing, Wireshark will help you autocomplete your filter.

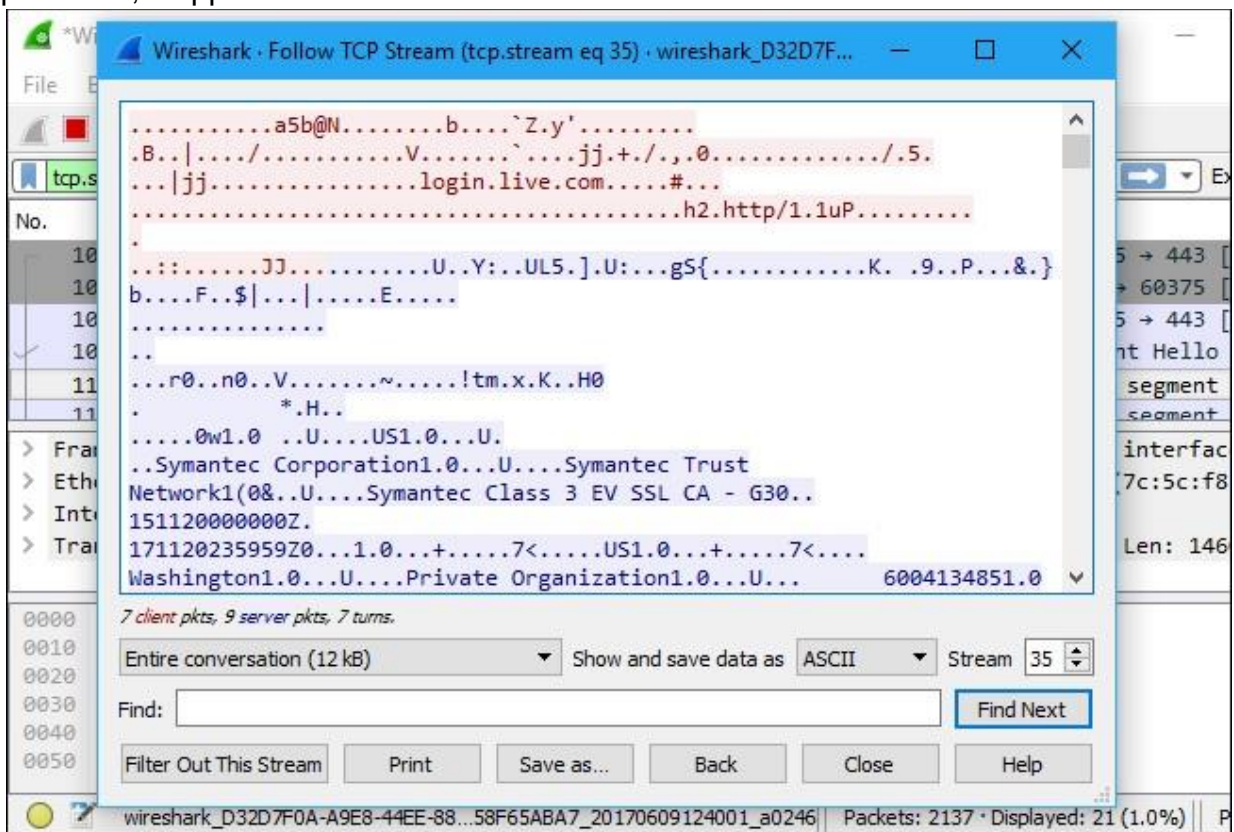


You can also click Analyze > Display Filters to choose a filter from among the default filters included in Wireshark. From here, you can add your own custom filters and save them to easily access them in the future.

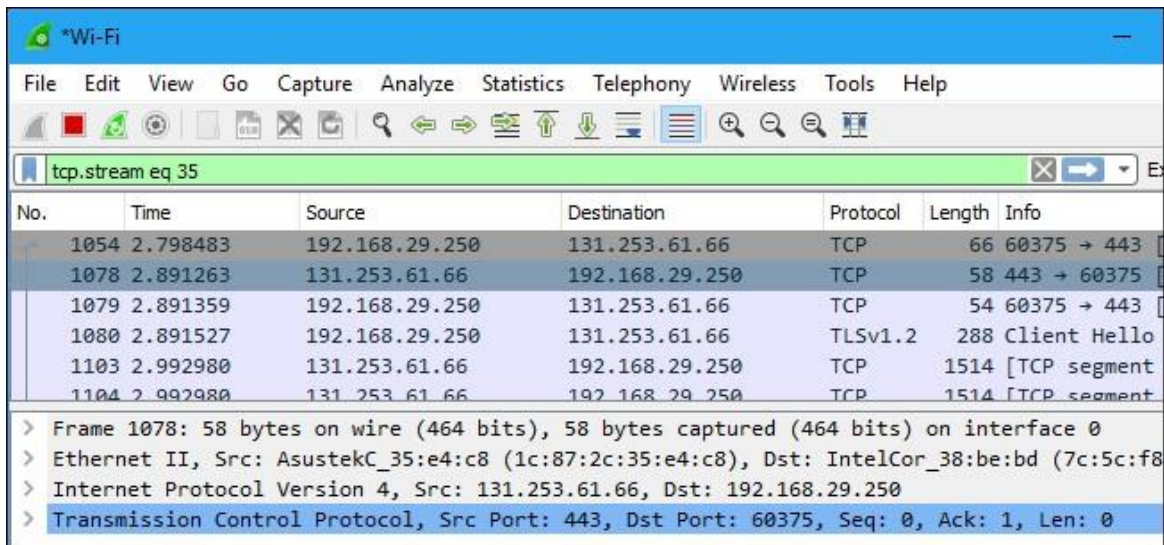
For more information on Wireshark's display filtering language, read the [Building display filter expressions](#) page in the official Wireshark documentation.



Another interesting thing you can do is right-click a packet and select Follow > TCP Stream. You'll see the full TCP conversation between the client and the server. You can also click other protocols in the Follow menu to see the full conversations for other protocols, if applicable.



Close the window and you'll find a filter has been applied automatically. Wireshark is showing you the packets that make up the conversation.

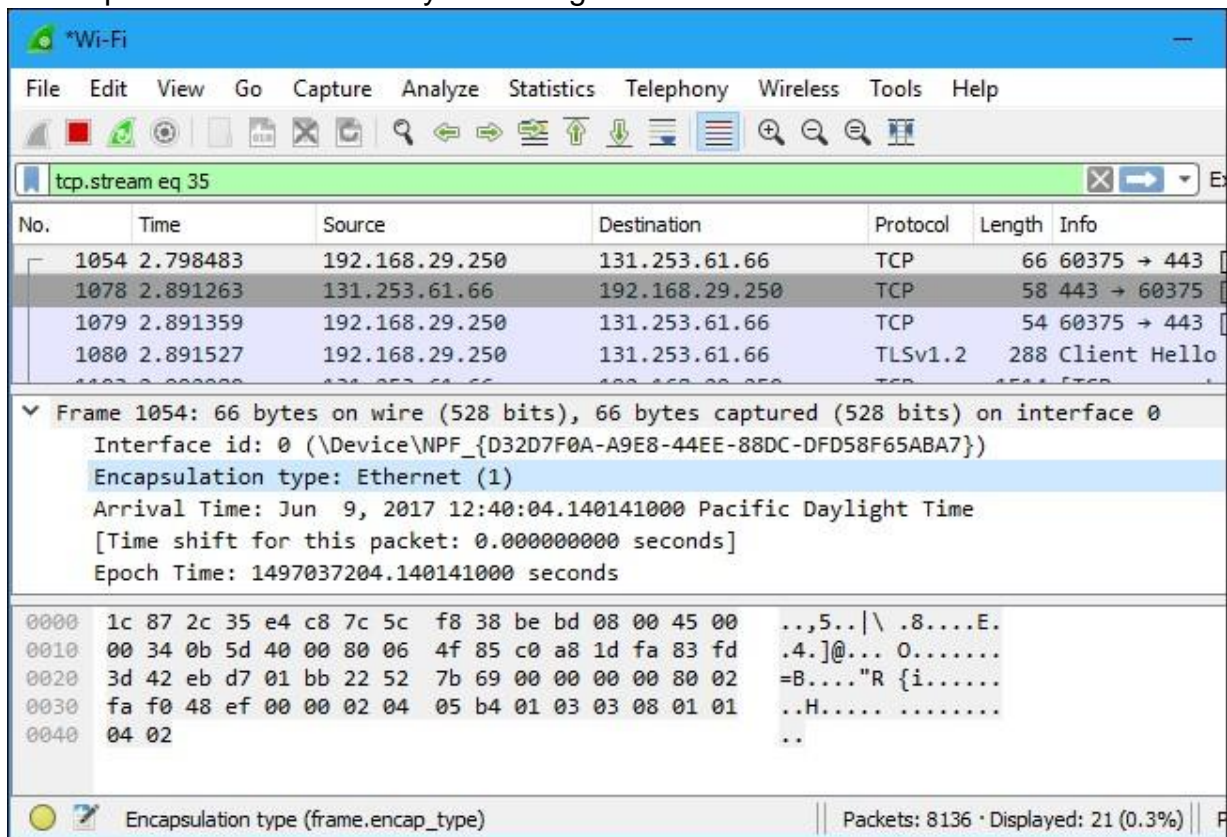


No.	Time	Source	Destination	Protocol	Length	Info
1054	2.798483	192.168.29.250	131.253.61.66	TCP	66	60375 → 443
1078	2.891263	131.253.61.66	192.168.29.250	TCP	58	443 → 60375
1079	2.891359	192.168.29.250	131.253.61.66	TCP	54	60375 → 443
1080	2.891527	192.168.29.250	131.253.61.66	TLSv1.2	288	Client Hello
1103	2.992980	131.253.61.66	192.168.29.250	TCP	1514	[TCP segment
1104	2.992980	131.253.61.66	192.168.29.250	TCP	1514	[TCP segment

> Frame 1078: 58 bytes on wire (464 bits), 58 bytes captured (464 bits) on interface 0
 > Ethernet II, Src: AsustekC_35:e4:c8 (1c:87:2c:35:e4:c8), Dst: IntelCor_38:be:bd (7c:5c:f8
 > Internet Protocol Version 4, Src: 131.253.61.66, Dst: 192.168.29.250
 > Transmission Control Protocol, Src Port: 443, Dst Port: 60375, Seq: 0, Ack: 1, Len: 0

Inspecting Packets

Click a packet to select it and you can dig down to view its details.



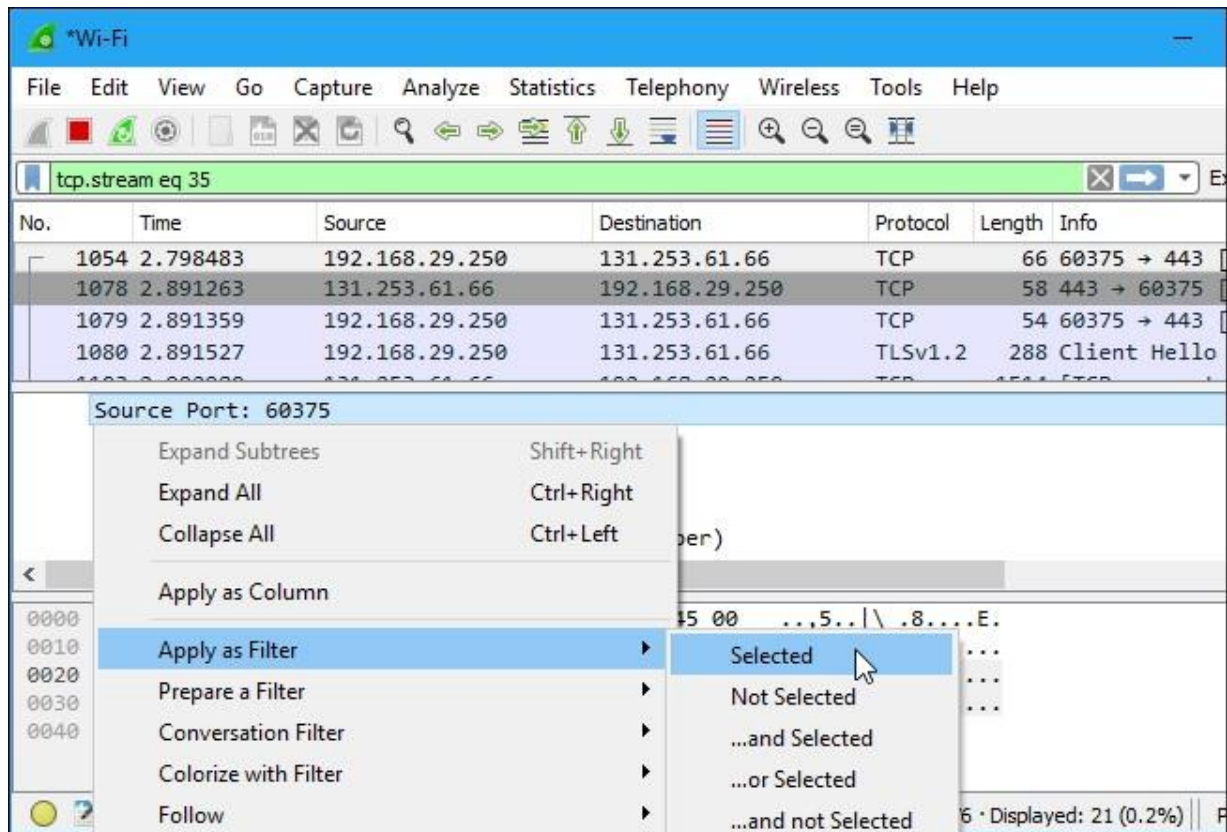
No.	Time	Source	Destination	Protocol	Length	Info
1054	2.798483	192.168.29.250	131.253.61.66	TCP	66	60375 → 443
1078	2.891263	131.253.61.66	192.168.29.250	TCP	58	443 → 60375
1079	2.891359	192.168.29.250	131.253.61.66	TCP	54	60375 → 443
1080	2.891527	192.168.29.250	131.253.61.66	TLSv1.2	288	Client Hello

▼ Frame 1054: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface 0
 Interface id: 0 (\Device\NPF_{D32D7F0A-A9E8-44EE-88DC-DFD58F65ABA7})
 Encapsulation type: Ethernet (1)
 Arrival Time: Jun 9, 2017 12:40:04.140141000 Pacific Daylight Time
 [Time shift for this packet: 0.000000000 seconds]
 Epoch Time: 1497037204.140141000 seconds

0000	1c 87 2c 35 e4 c8 7c 5c f8 38 be bd 08 00 45 00	...5.. \ .8....E.
0010	00 34 0b 5d 40 00 80 06 4f 85 c0 a8 1d fa 83 fd	.4.]@... 0.....
0020	3d 42 eb d7 01 bb 22 52 7b 69 00 00 00 00 80 02	=B...."R {i.....
0030	fa f0 48 ef 00 00 02 04 05 b4 01 03 03 08 01 01	..H.....
0040	04 02	..

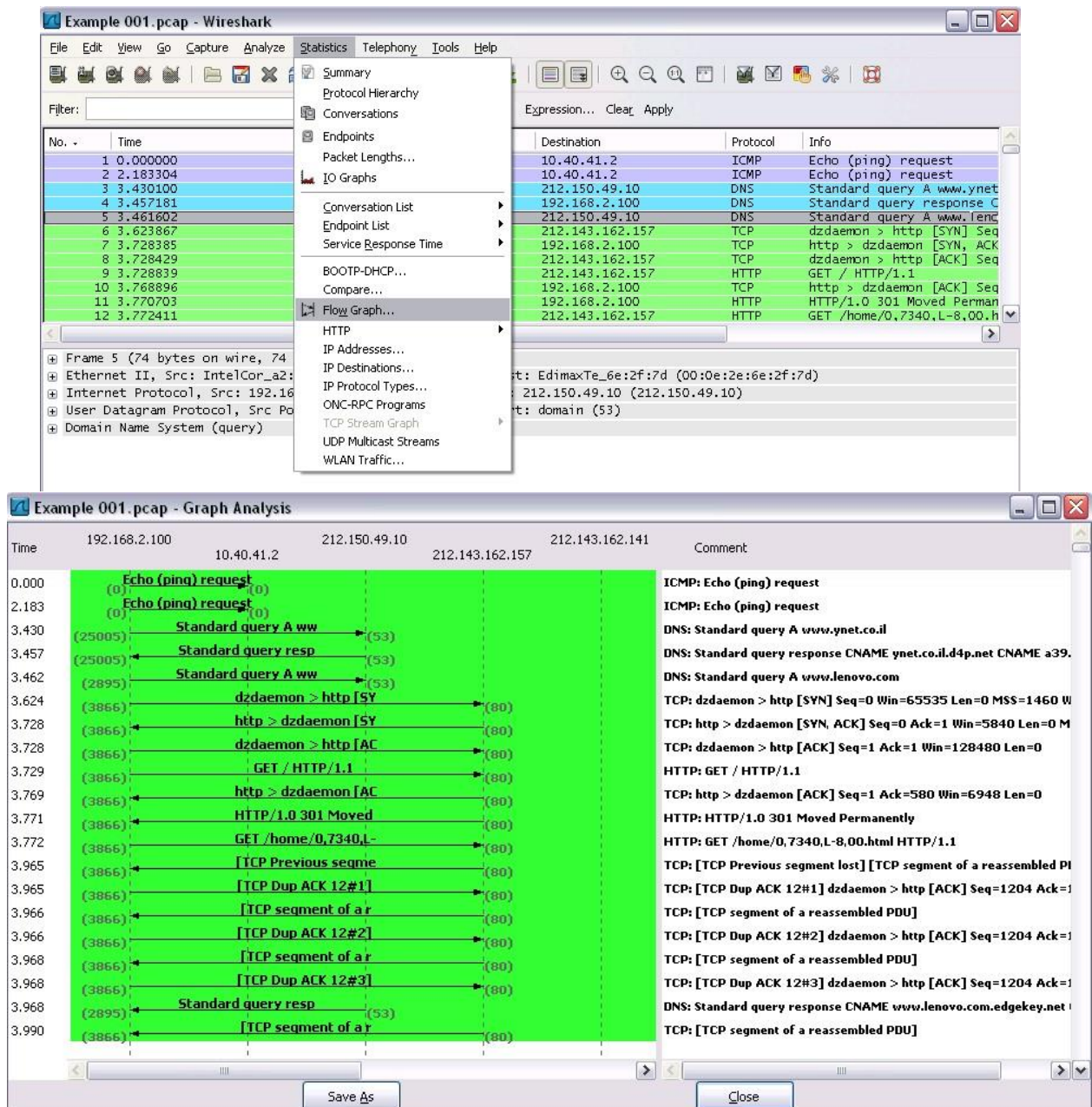
Encapsulation type (frame.encap_type) | Packets: 8136 · Displayed: 21 (0.3%)

You can also create filters from here — just right-click one of the details and use the Apply as Filter submenu to create a filter based on it.



Wireshark is an extremely powerful tool, and this tutorial is just scratching the surface of what you can do with it. Professionals use it to debug network protocol implementations, examine security problems and inspect network protocol internals.

Flow Graph: Gives a better understanding of what we see.



CAPTURING AND ANALYSING PACKETS USING WIRESHARK TOOL

To filter, capture, view, packets in Wireshark Tool.

Capture 100 packets from the Ethernet: IEEE 802.3 LAN Interface and save it.

Procedure

- Select Local Area Connection in Wireshark.
- Go to capture ☐ option
- Select stop capture automatically after 100 packets.
- Then click Start capture.

- Save the packets.

Output

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	Pegatron_e0:87:9e	Broadcast	ARP	60	Who has 172.16.9.94? Tell 172.16.9.138
2	0.000180	RealtekS_55:2c:b8	Broadcast	ARP	60	Who has 172.16.10.36? Tell 172.16.10.50
3	0.000294	RealtekS_55:2c:b8	Broadcast	ARP	60	Who has 172.16.11.36? Tell 172.16.10.50
4	0.000295	RealtekS_55:2c:b8	Broadcast	ARP	60	Who has 172.16.8.37? Tell 172.16.10.50
5	0.000296	RealtekS_55:2c:b8	Broadcast	ARP	60	Who has 172.16.9.37? Tell 172.16.10.50
6	0.000296	RealtekS_55:2c:b8	Broadcast	ARP	60	Who has 172.16.11.37? Tell 172.16.10.50
7	0.001460	fe80::4968:12a7:5e3...	ff02::1:3	LLMNR	95	Standard query 0xae2b A TLFL3-HDC101701
8	0.001622	172.16.8.95	224.0.0.252	LLMNR	75	Standard query 0xae2b A TLFL3-HDC101701
9	0.001623	172.16.8.95	224.0.0.252	LLMNR	75	Standard query 0x28c0 AAAA TLFL3-HDC101701
10	0.001625	fe80::4968:12a7:5e3...	ff02::1:3	LLMNR	95	Standard query 0x28c0 AAAA TLFL3-HDC101701
11	0.045051	fe80::2d3b:daa7:1e00...	ff02::1:3	LLMNR	95	Standard query 0xae2b A TLFL3-HDC101701

▶ Frame 7: 95 bytes on wire (760 bits), 95 bytes captured (760 bits) on interface 0
 ▶ Ethernet II, Src: Dell_35:10:a8 (50:9a:4c:35:10:a8), Dst: IPv6mcast_01:00:03 (33:33:00:01:00:03)
 ▶ Internet Protocol Version 6, Src: fe80::4968:12a7:5e36:523e, Dst: ff02::1:3
 ▶ User Datagram Protocol, Src Port: 62374, Dst Port: 5355
 Source Port: 62374
 Destination Port: 5355
 Length: 41
 Checksum: 0x90e0 [unverified]
 [Checksum Status: Unverified]
 [Stream index: 0]
 ▶ Link-local Multicast Name Resolution (query)

```

0000  33 33 00 01 00 03 50 9a  4c 35 10 a8 86 dd 60 00  33...P L5...
0010  00 00 00 29 11 01 fe 80  00 00 00 00 00 00 49 68   ....Ih
0020  12 a7 5e 36 52 3e ff 02  00 00 00 00 00 00 00 00   ...6R>...
0030  00 00 00 01 00 03 f3 a6  14 eb 00 29 90 e0 ae 2b   ....+
0040  00 00 00 01 00 00 00 00  00 00 0f 54 4c 4c 33       ....TLFL3
0050  2d 48 44 43 31 30 31 37  30 31 00 00 01 00 01      -HDC1017 01-
  
```

1. Create a Filter to display only TCP/UDP packets, inspect the packets and provide the flow graph

Procedure

- Select Local Area Connection in Wireshark.
- Go to capture ☐ option
- Select stop capture automatically after 100 packets.
- Then click Start capture.
- Search TCP packets in search bar.
- To see flow graph click Statistics ☐ Flow graph.
- Save the packets.

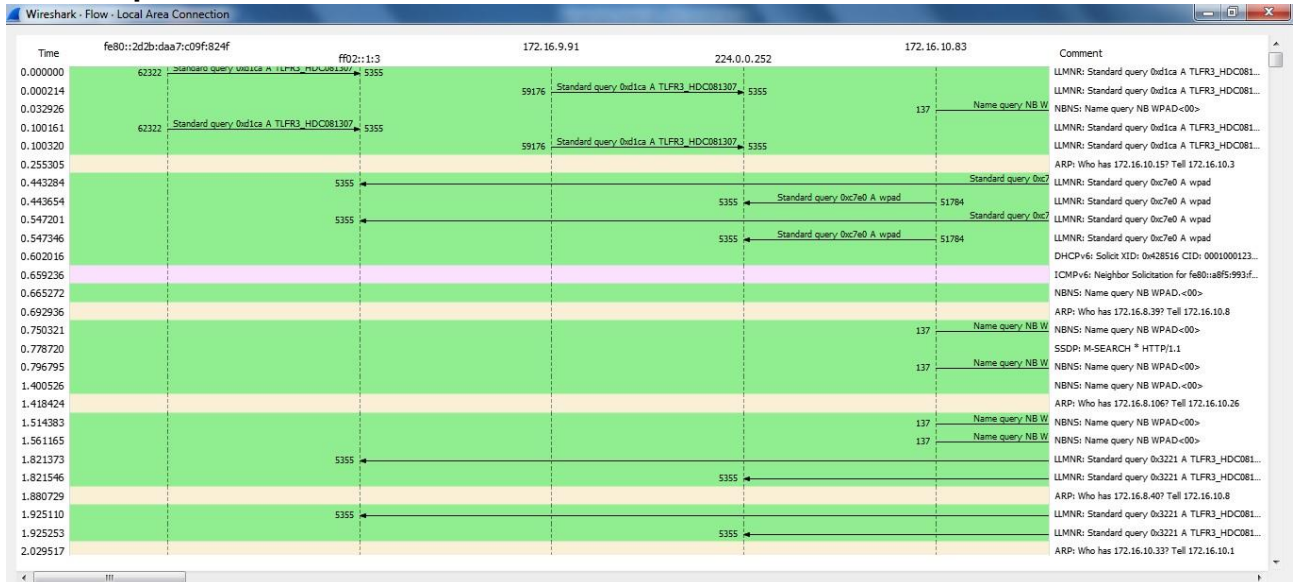
No.	Time	Source	Destination	Protocol	Length	Info
123	4.557832	fe80::8532:3a9f:aff...	fe80::5c2b:19eb:d33...	TCP	74	1509 → 2869 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
126	4.557993	172.16.9.106	172.16.9.96	TCP	60	1506 → 2869 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
1095	30.718732	172.16.8.83	172.16.9.96	TCP	66	51526 → 2869 [SYN, ECN, CWR] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
1096	30.718794	172.16.9.96	172.16.8.83	TCP	66	2869 → 51526 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
1097	30.719129	172.16.8.83	172.16.9.96	TCP	60	51526 → 2869 [ACK] Seq=1 Ack=1 Win=65536 Len=0
1098	30.719150	172.16.8.83	172.16.9.96	TCP	180	2869 → 51526 [PSH, ACK] Seq=1 Ack=133 Win=65536 Len=224 [TCP segment of a reassembled PDU]
1099	30.719919	172.16.9.96	172.16.8.83	TCP	278	2869 → 51526 [PSH, ACK] Seq=1 Ack=133 Win=65536 Len=224 [TCP segment of a reassembled PDU]
1100	30.719986	172.16.9.96	172.16.8.83	TCP	1514	2869 → 51526 [ACK] Seq=225 Ack=133 Win=65536 Len=1460 [TCP segment of a reassembled PDU]
1101	30.720279	172.16.8.83	172.16.9.96	TCP	60	51526 → 2869 [ACK] Seq=133 Ack=1685 Win=65536 Len=0
1102	30.720350	172.16.8.83	172.16.9.96	TCP	60	51526 → 2869 [ACK] Seq=133 Ack=1685 Win=65536 Len=0

▶ Frame 123: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
 ▶ Ethernet II, Src: RealtekS_b2:60:90 (00:e0:4c:b2:60:90), Dst: IntelCor_13:ed:7c (00:27:0e:13:ed:7c)
 ▶ Internet Protocol Version 6, Src: fe80::8532:3a9f:aff1:b3ca, Dst: fe80::5c2b:19eb:d33d:a1cd
 ▶ Transmission Control Protocol, Src Port: 1509, Dst Port: 2869, Seq: 1, Ack: 1, Len: 0

```

0000  00 27 0e 13 ed 7c 00 e0  4c b2 60 90 86 dd 60 00   ....L...
0010  00 00 00 14 06 80 fe 80  00 00 00 00 00 85 32      .....2
0020  3a 9f af f1 b3 ca fe 80  00 00 00 00 00 5c 2b      .....+
0030  19 eb d3 3d a1 cd 05 e5  00 35 3b ef f1 2f bf d2      ...5;...
0040  67 35 50 14 00 00 3e de  00 00                                gSP >...
  
```

Flow Graph



2. Create a Filter to display only ARP packets and inspect the packets.

Procedure

- Go to capture ☐ option
- Select stop capture automatically after 100 packets.
- Then click Start capture.
- Search ARP packets in search bar.
- Save the packets.

Output

arp						
No.	Time	Source	Destination	Protocol	Length	Info
6	0.255305	Foxconn_c9:c5:f0	Broadcast	ARP	60	Who has 172.16.10.15? Tell 172.16.10.3
14	0.692936	Foxconn_d0:ac:46	Broadcast	ARP	60	Who has 172.16.8.39? Tell 172.16.10.8
19	1.418424	Foxconn_c9:c9:91	Broadcast	ARP	60	Who has 172.16.8.106? Tell 172.16.10.26
24	1.880729	Foxconn_d0:ac:46	Broadcast	ARP	60	Who has 172.16.8.40? Tell 172.16.10.8
27	2.029517	Giga-Byt_92:d2:ef	Broadcast	ARP	60	Who has 172.16.10.33? Tell 172.16.10.1
41	2.509905	Giga-Byt_7c:c5:34	Broadcast	ARP	60	Who has 172.16.9.82? Tell 172.16.9.111
44	2.602358	Foxconn_c9:c8:24	Broadcast	ARP	60	Who has 172.16.8.139? Tell 172.16.10.22
46	2.743021	Dell_35:11:11	Broadcast	ARP	60	Who has 172.16.8.118? Tell 172.16.10.195
56	3.201822	Giga-Byt_92:d2:ef	Broadcast	ARP	60	Who has 172.16.10.34? Tell 172.16.10.1
60	3.237061	Giga-Byt_7c:c5:34	Broadcast	ARP	60	Who has 172.16.9.82? Tell 172.16.9.111
71	3.429062	Dell_35:11:11	Broadcast	ARP	60	Who has 172.16.8.118? Tell 172.16.10.195

▶ Frame 119: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0
 ▶ Ethernet II, Src: IntelCor_13:ed:7c (00:27:0e:13:ed:7c), Dst: RealtekS_b2:60:90 (00:e0:4c:b2:60:90)
 ▶ Address Resolution Protocol (reply)

```

0000  00 e0 4c b2 60 90 00 27 0e 13 ed 7c 08 06 00 01  ..L...+...|...
0010  00 00 06 04 00 02 00 27 0e 13 ed 7c ac 10 09 60  ....+...+...
0020  00 e0 4c b2 60 90 ac 10 09 6a  ..L...+...j
  
```

3. Create a Filter to display only DNS packets and provide the flow graph.

Procedure

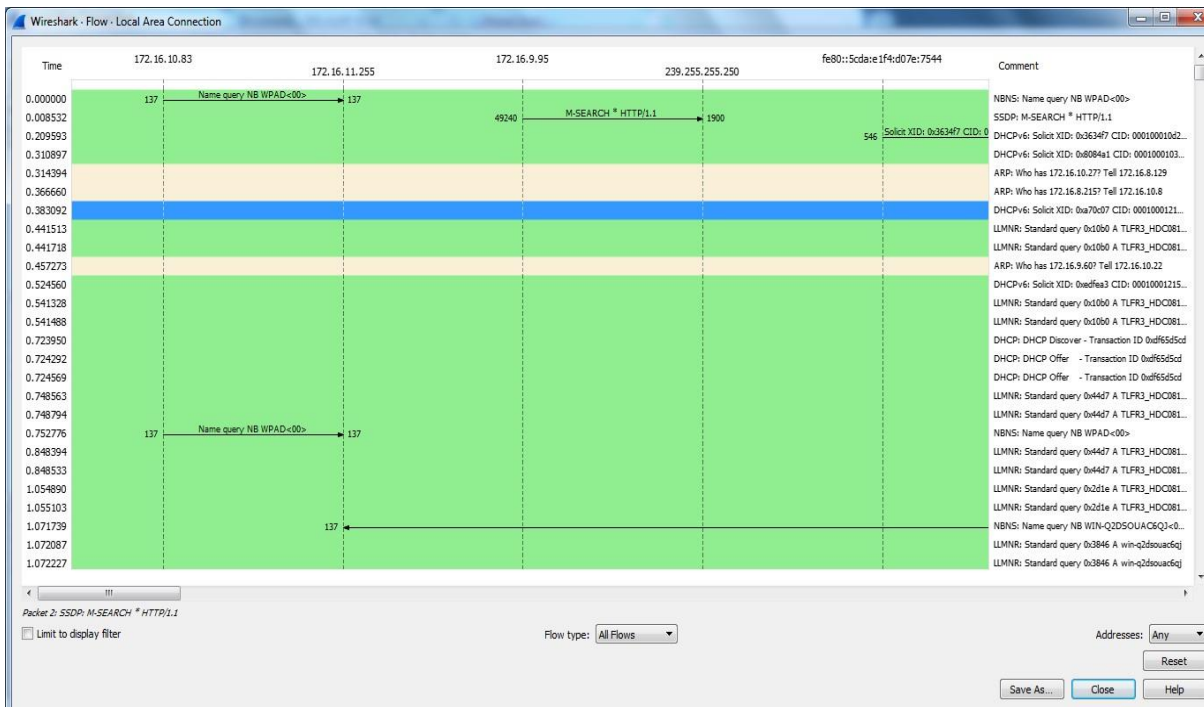
- Go to capture → option
- Select stop capture automatically after 100 packets.
- Then click Start capture.
- Search DNS packets in search bar.
- To see flow graph click Statistics→Flow graph.
- Save the packets.

*Local Area Connection						
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help						
dns						
No.	Time	Source	Destination	Protocol	Length	Info
989	32.977988	172.16.9.96	172.16.8.1	DNS	74	Standard query 0x9e40 A www.google.com
990	32.978738	172.16.8.1	172.16.9.96	DNS	90	Standard query response 0x9e40 A www.google.com A 172.217.163.132
1199	37.273599	172.16.9.96	172.16.8.1	DNS	79	Standard query 0xb58b A accounts.google.com
1200	37.273822	172.16.9.96	172.16.8.1	DNS	75	Standard query 0x6af4 A ssl.gstatic.com
1201	37.273837	172.16.8.1	172.16.9.96	DNS	95	Standard query response 0xb58b A accounts.google.com A 172.217.163.141
1202	37.273878	172.16.8.1	172.16.9.96	DNS	91	Standard query response 0x6af4 A ssl.gstatic.com A 172.217.26.163
1203	37.274368	172.16.9.96	172.16.8.1	DNS	77	Standard query 0xe76d A fonts.gstatic.com
1204	37.274541	172.16.8.1	172.16.9.96	DNS	129	Standard query response 0xe76d A fonts.gstatic.com CNAME.gstaticadssl1.google.com A 172.217.160.131
1738	38.875863	172.16.9.96	172.16.8.1	DNS	80	Standard query 0x7a60 A accounts.youtube.com
1739	38.875294	172.16.8.1	172.16.9.96	DNS	124	Standard query response 0x7a60 A accounts.youtube.com CNAME www3.l.google.com A 172.217.167.142

▶ Frame 989: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
 ▶ Ethernet II, Src: IntelCor_13:ed:7c (00:27:0e:13:ed:7c), Dst: Caswell_f2:b4:a1 (08:35:71:f2:b4:a1)
 ▶ Internet Protocol Version 4, Src: 172.16.9.96, Dst: 172.16.8.1
 ▶ User Datagram Protocol, Src Port: 62578, Dst Port: 53
 ▶ Domain Name System (query)

```

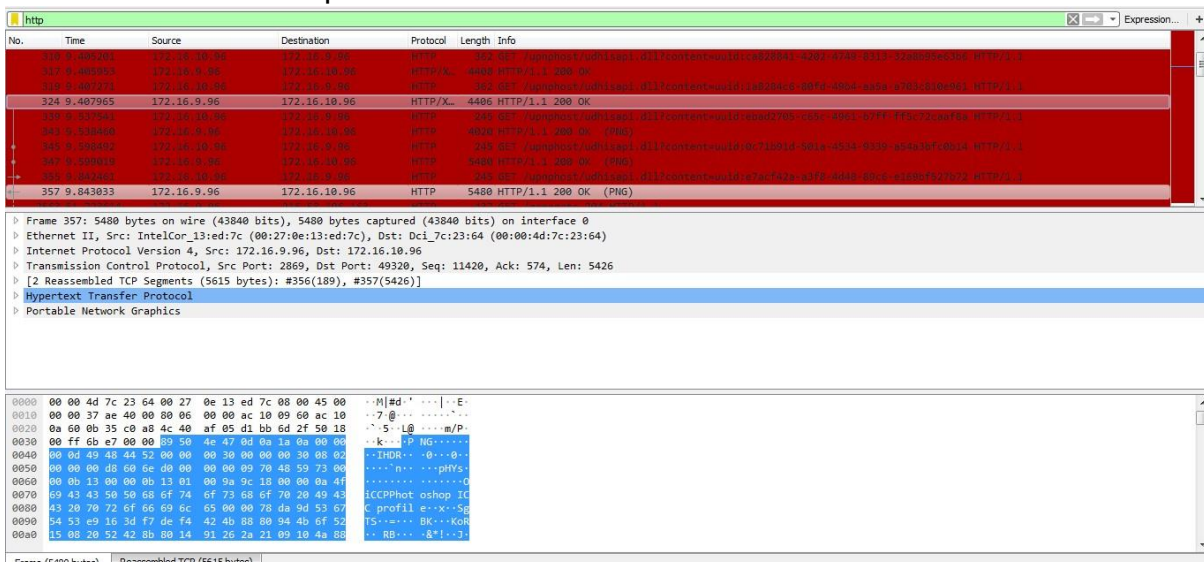
0000  08 35 71 f2 b4 a1 00 27 0e 13 ed 7c 08 00 45 00  5g.....|..E
0010  00 3c 37 bb 00 00 00 11 00 00 ac 10 09 60 ac 10  <7.....
0020  00 01 f3 46 00 35 00 28 69 bb 9e 40 01 00 00 01  ...F.S.(1-0...
0030  00 00 00 00 00 00 03 77 77 77 06 67 6f 6f 67 6c  ...w ww.googl
0040  05 03 03 6f 6d 00 00 01 00 01  ..e.com...
  
```

4. Create a Filter to display only HTTP packets and inspect the packets

Procedure

- Select Local Area Connection in Wireshark.
- Go to capture ☐ option
- Select stop capture automatically after 100 packets.
- Then click Start capture.
- Search HTTP packets in search bar.
- Save the packets.

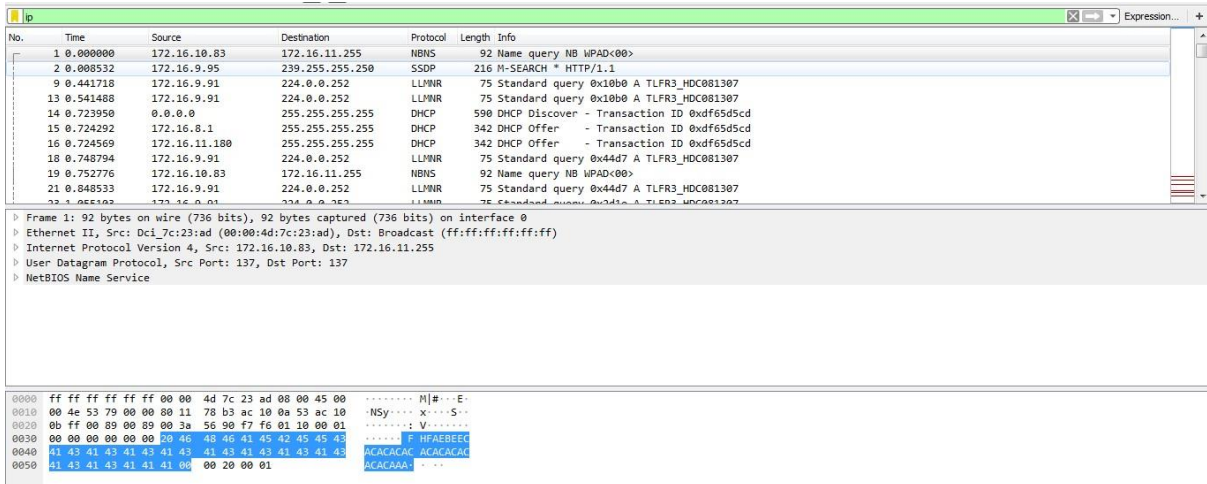


5. Create a Filter to display only IP/ICMP packets and inspect the packets.

Procedure

CS19541-COMPUTER NETWORKS-LAB MANUAL

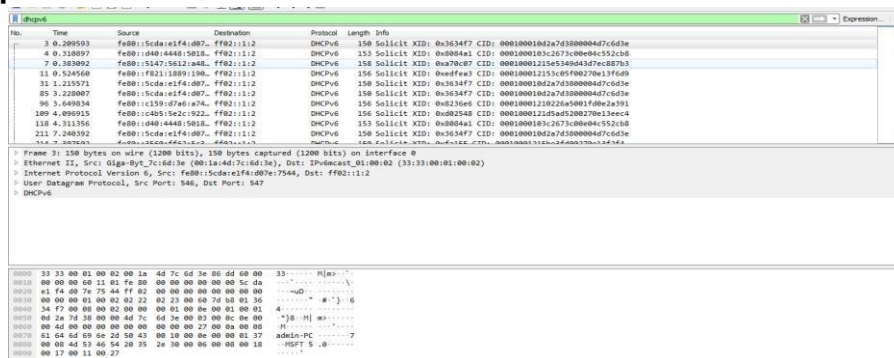
- Select Local Area Connection in Wireshark.
- Go to capture ☐ option
- Select stop capture automatically after 100 packets.
- Then click Start capture.
- Search ICMP/IP packets in search bar.
- Save the packets



6. Create a Filter to display only DHCP packets and inspect the packets. Procedure

- Select Local Area Connection in Wireshark.
- Go to capture ☐ option
- Select stop capture automatically after 100 packets.
- Then click Start capture.
- Search DHCP packets in search bar.
- Save the packets

Output



Student observation:

1. What is promiscuous mode?
2. Does ARP packets has transport layer header? Explain.
3. Which transport layer protocol is used by DNS?

4. What is the port number used by http protocol?
5. What is a broadcast ip address?

Answers:

1. Promiscuous Mode: NIC receives all network packets, not just those addressed to it.
2. ARP Packets and Transport Layer Header: No, ARP operates at Layer 2 and doesn't use a transport layer header.
3. Transport Layer Protocol for DNS: DNS typically uses UDP (and TCP when needed).
4. HTTP Port Number: Port 80(HTTPS uses port 443).
5. Broadcast IP Address: The address used to send packets to all devices in a network (e.g., 192.168.1.255 in a /24 subnet).

Result:

Thus the experiment on wireshark was studied and completed successfully.