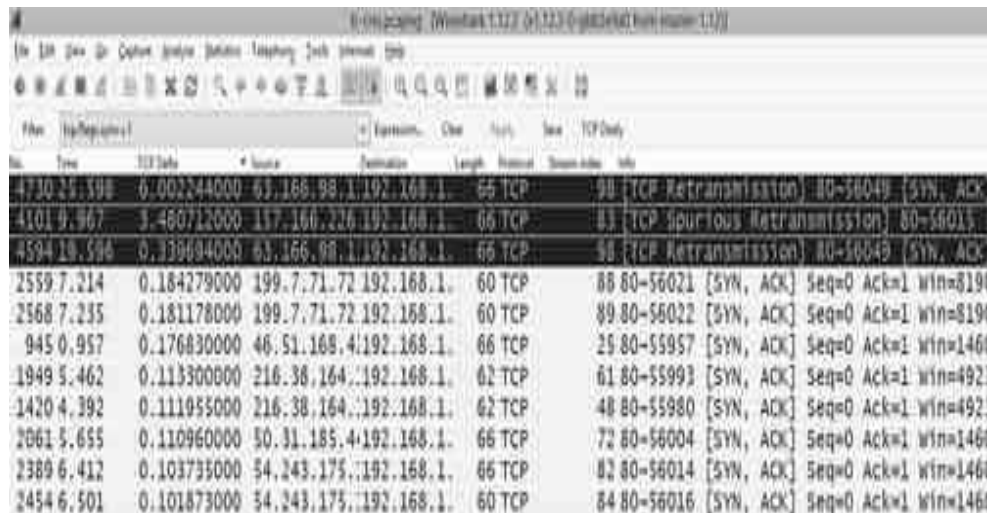




The image shows a Wireshark packet capture window. The top menu bar includes File, Edit, Display, Options, Analyze, Statistics, TCP/UDP, Tools, Window, and Help. The packet list pane shows several packets, with three highlighted in red to indicate retransmissions. The packet details pane shows the selected packet's structure, including Ethernet II, Internet Protocol Version 4, and Transmission Control Protocol. The packet bytes pane shows the raw data of the selected packet.

No.	Time	Source	Destination	Length	Protocol	Summary
4730	25.398	63.166.98.1	192.168.1.1	66	TCP	98 [TCP Retransmission] 80-56049 [SYN, ACK]
4101	9.967	137.160.226	192.168.1.1	66	TCP	83 [TCP Spurious Retransmission] 80-56015
4594	19.596	63.166.98.1	192.168.1.1	66	TCP	98 [TCP Retransmission] 80-56049 [SYN, ACK]
2559	7.214	199.7.71.72	192.168.1.1	60	TCP	88 80-56021 [SYN, ACK] Seq=0 Ack=1 win=8192
2568	7.235	199.7.71.72	192.168.1.1	60	TCP	89 80-56022 [SYN, ACK] Seq=0 Ack=1 win=8192
945	0.957	46.51.168.4	192.168.1.1	66	TCP	25 80-55957 [SYN, ACK] Seq=0 Ack=1 win=1460
1949	5.462	216.38.164.	192.168.1.1	62	TCP	61 80-55993 [SYN, ACK] Seq=0 Ack=1 win=492
1420	4.392	216.38.164.	192.168.1.1	62	TCP	48 80-55980 [SYN, ACK] Seq=0 Ack=1 win=492
2061	5.655	50.31.185.4	192.168.1.1	66	TCP	72 80-56004 [SYN, ACK] Seq=0 Ack=1 win=1460
2389	6.412	54.243.175.	192.168.1.1	66	TCP	82 80-56014 [SYN, ACK] Seq=0 Ack=1 win=1460
2454	6.501	54.243.175.	192.168.1.1	60	TCP	84 80-56016 [SYN, ACK] Seq=0 Ack=1 win=1460

Figure 123.1: Obtain the Round Trip Time (RTT) Using the TCP Handshake

We can see that there are three packets that are marked as Retransmissions. These are because of connection establishment problems. We can also observe at the RTT to various servers.

Topic 124: RTT: Packets 2 and 3 of TCP Handshake

This topic describes how to use Packets 2 & 3 of TCP handshake for obtaining RTT in Wireshark.

Packets 2 and 3 of the TCP handshake can be used to measure the path latency or round trip time (RTT). We need to run a packet capturing tool such as Wireshark on the server host, and look at the **tcp.time_delta** value between the server's TCP SYN/ACK packet and the client's TCP ACK response. The SYN/ACK packet is easy to locate with a display filter, but the ACK (third packet of the handshake) is a bit trickier. The second (the SYN/ACK) packet of the TCP handshake can be detected by applying the filter: **tcp.flags.syn==1 && tcp.flags.ack==1**.

Detection of the third packet of the handshake is difficult. Use the following characteristics:

- **a) tcp.seq==1** (Required) TCP Seq. Number 1 (Relative Seq. Number)
- **b) tcp.ack==1** (Required) TCP Ack. Number 1 (Relative Ack. Number)
- **c) tcp.len > 0** (Optional) data in the third packet of the handshake
- **d) tcp.push==1** (Optional) PUSH bit set

The filter then becomes **(tcp.flags.syn==1 && tcp.flags.ack==1) || (tcp.seq==1 && tcp.ack==1)**.

Topic 125: Wireshark Lab 32

This topic obtains the round trip time (RTT) using Display Filters in Wireshark.

Step 1: Open *tr-chappellu.pcapng*.

Step 2: Enter the filter **tcp.flags.syn==1** in the display filter area and then click **Apply**. You will find that 58 packets match this filter. The first two packets are sent from the client port 35,621. Packet 3 and Packet 4 are the first two packets of a new TCP connection. The TCP Delta column indicates the time from the TCP SYN from port 35,622 and the SYN/ACK to that same port, RTT is about 17 ms.

No.	Time	TCP Delta	Source	Destination	Length	Protocol	Stream order	Info
1	0.000	0.000000000	24.6.173.2	198.66.239.24	66	TCP	0	35621->80 [SYN] Seq=0 win=8192
3	3.007	3.007726000	24.6.173.2	198.66.239.24	66	TCP	0	(TCP Retransmission) 35621->80
4	3.008	0.000000000	24.6.173.2	198.66.239.24	66	TCP	1	35622->80 [SYN] Seq=0 win=8192
14	8.317	0.017599000	198.66.239.24	24.6.173.2	66	TCP	180	35622->80 [SYN, ACK] Seq=0 Ack=

Figure 125.1: Obtain RTT using Display Filters

This trace file, which was captured at the client can be used to locate the second and third packet of the TCP handshake. These packets can be used to determine RTT when capturing at the server.

Step 3: Enter the following filter in the display filter area: **(tcp.flags.syn==1 && tcp.flags.ack==1) || (tcp.seq==1 && tcp.ack==1)**. Click **Apply** and let's examine the results. There are 69 packets match this filter. Several packets from this set are not of interest to us.

Step 4: Let's enhance the filter with the following conditions: **(tcp.flags.syn==1 && tcp.flags.ack==1) || (tcp.seq==1 && tcp.ack==1) && tcp.len==0 && tcp.flags.fin==0**. Click **Apply**.

No.	Time	Source	Destination	Protocol	Length	Info
148.317	0.017590000	24.6.173.2198	66.250.24.6	TCP	60	66 → 35622 [SYN, ACK] Seq=0 Ack=1 Win=0
158.318	0.000186000	24.6.173.2198	66.250.24.6	TCP	60	66 → 35622 [ACK] Seq=1 Ack=1 Win=657
198.409	0.017623000	69.59.180.224	24.6.173.2198	TCP	60	24 → 35623 [SYN, ACK] Seq=0 Ack=1 Win=0
208.409	0.000159000	24.6.173.2198	69.59.180.224	TCP	60	24 → 35623 [ACK] Seq=1 Ack=1 Win=662
328.575	0.015573000	69.59.180.224	24.6.173.2198	TCP	60	24 → 35625 [SYN, ACK] Seq=0 Ack=1 Win=0
338.575	0.000126000	24.6.173.2198	69.59.180.224	TCP	60	24 → 35625 [ACK] Seq=1 Ack=1 Win=662
528.675	0.020736000	141.101.121	24.6.173.2198	TCP	60	24 → 35627 [SYN, ACK] Seq=0 Ack=1 Win=0
538.675	0.020893000	141.101.121	24.6.173.2198	TCP	60	24 → 35626 [SYN, ACK] Seq=0 Ack=1 Win=0
548.676	0.000137000	24.6.173.2198	141.101.121	TCP	60	24 → 35627 [ACK] Seq=1 Ack=1 Win=657
558.676	0.000179000	24.6.173.2198	141.101.121	TCP	60	24 → 35626 [ACK] Seq=1 Ack=1 Win=657
708.772	0.094363000	184.73.250	24.6.173.2198	TCP	60	24 → 35629 [SYN, ACK] Seq=0 Ack=1 Win=0

We now see only the SYN/ACK and ACK packets of the handshakes in the trace file. Using the TCP Delta column, we can determine the time between each of these packets in each of TCP conversations. For future use, we can save the filter expressions as buttons so that high path latency can be identified.

Step 5: Click **Clear** to remove the filter.

Topic 126: Wireshark Lab 33

In this topic, we discuss how to plot TCP delays in Wireshark.

Wireshark's advanced IO Graph can be used to plot the maximum **tcp.time_delta** value to locate TCP conversation delays in a trace file.

Step 1: Open *tr-chappellu.pcapng*.

Step 2: Select **Statistics | IO Graph**.

Step 3: In the Y Axis **Unit** area, select **Advanced...**

Step 4: Select the **MAX(*)** Graph 1 Calc option and enter **tcp.time_delta** in the Calc area.

Step 5: Click the **Graph 1** button to graph your results.

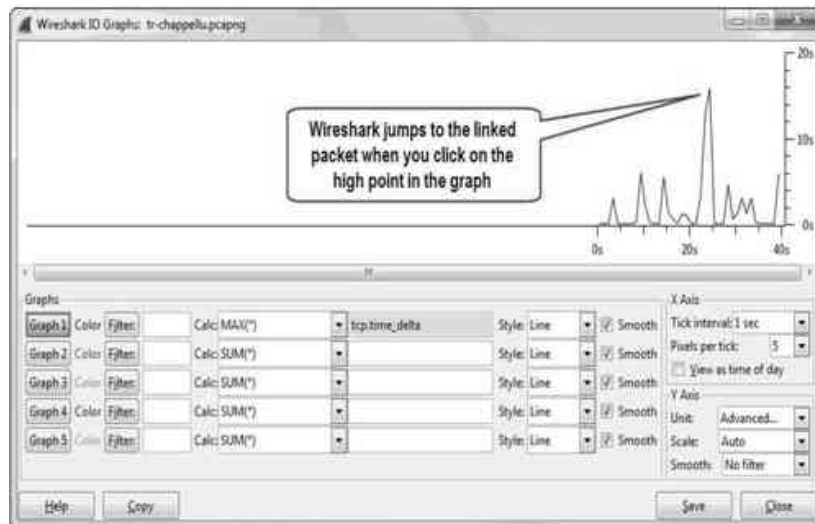
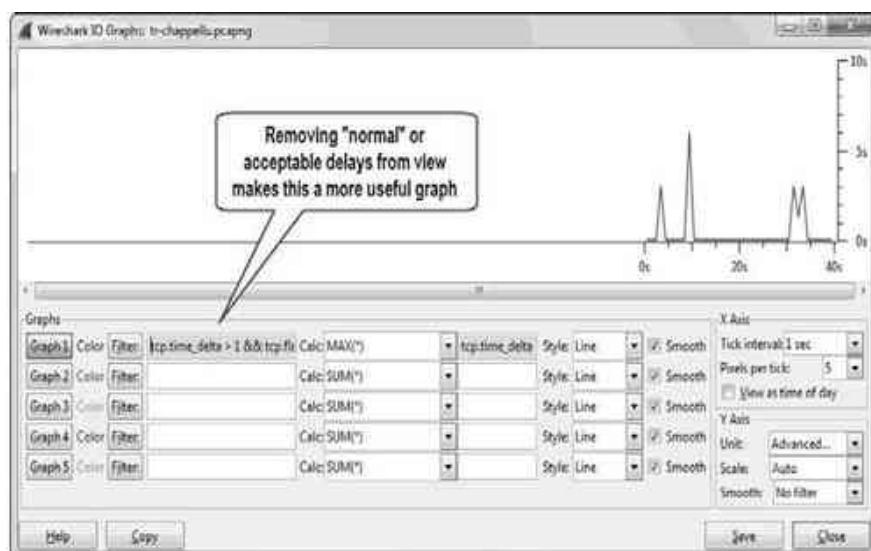


Figure 126.1: Graph TCP Delays

From the graph of this trace file traffic, we can observe that there is a spike in the RTT values around 25 seconds. As you click on this high point in the graph, Wireshark will jump to that packet in the main window. Wireshark takes us to packet number 155. The TCP Delta column value is 15.757807 seconds. This is a TCP FIN. For such a delay, we don't care much. Let's add a filter to remove such normal delays.

Step 6: Enter the following expression in the Filter area of Graph1: **tcp.time_delta > 1 && tcp.flags.fin==0 && tcp.flags.reset==0 && !http.request.method=="GET"**. Click the **Graph 1** button again to so that the filter can be applied.



Now when you click on the largest delay point in this graph, Wireshark jumps to Packet 86.

No.	Time	Source	Destination	Length	Protocol	Stream	Info
80	8.830	24.6.173.275	75.75.75.75	79	DNS		Standard query 0xfd77 AAAA www.nitrore
81	8.838	75.75.75.724	6.173.102	DNS			Standard query response 0xe4dd AAAA 204
82	8.860	24.6.173.275	75.75.75.75	79	DNS		Standard query 0xfd77 AAAA www.nitrore
83	8.869	75.75.75.724	6.173.400	DNS			Standard query response 0xfd77 Server f
84	8.873	75.75.75.724	6.173.400	DNS			Standard query response 0xfd77 Server f
85	8.913	0.69586280	184.73.25024.6.173	60	TCP		700 35629 [ACK] Seq=1 Ack=601 win=7168
86	9.004	5.99643000	24.6.173.2198.66.2	62	TCP		0 [TCP Retransmission] 35621-80 [SYN] Seq=
87	9.020	0.01670400	198.66.23024.6.173	62	TCP		0 80 35621 [SYN, ACK] Seq=0 Ack=1 win=655
88	9.021	0.00018000	24.6.173.2198.66.2	54	TCP		0 35621-80 [ACK] Seq=1 Ack=1 win=64240 Le
89	9.153	0.24018900	184.73.25024.6.173	885	HTTP		7 HTTP/1.1 200 OK (application/json)

We can see that this is a TCP SYN retransmission packet. It took almost 6 seconds after the previous packet. Thus, indicating a connection request that is not receiving responses.

Topic 127: Wireshark Lab 34

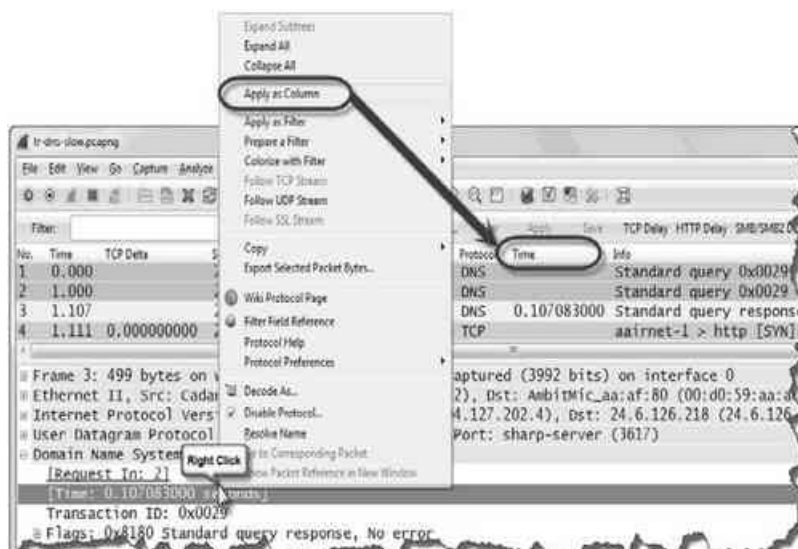
This topic how to find DNS response times in Wireshark.

Domain Name Service (DNS) provides the network IP address of a machine for a network name such as (such as www.wireshark.org). Applications protocols such as SMTP, HTTP generate DNS queries when a user provides a network name to them. DNS, which is a request/response protocol, employs UDP.

Step 1: Open *tr-dns-slow.pcapng*.

Step 2: The **dns.time** field exists only in DNS response packets. In this trace file, Packet 3 is the first DNS response packet. Let's use this packet to create a **dns.time** column. Expand Packet 3's the **Domain Name System (response)** section.

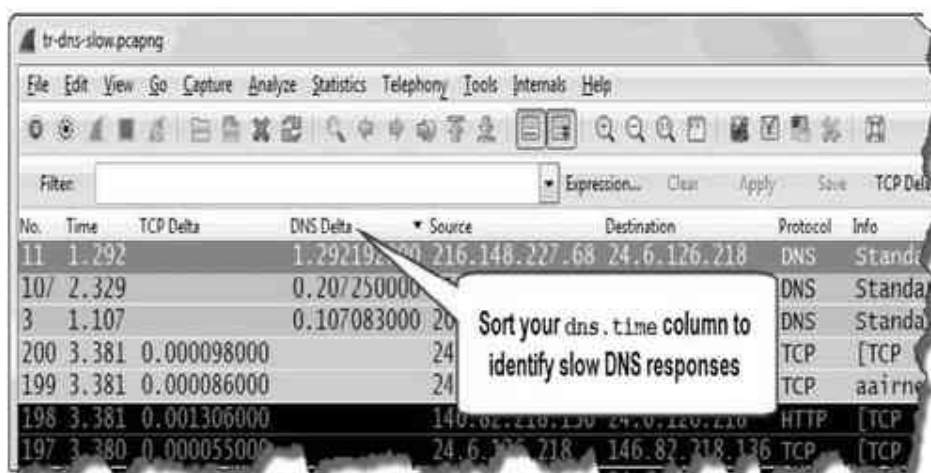
Step 3: Right-click on the **[Time: 0.107083000 seconds]** line and click **Apply as Column**.



Step 4: The newly created column appears to the left of the Info column. You can click and drag your new column to the right of the existing **TCP Delta** column.

Step 5: Right-click on the new column heading and selecting **Edit Column Details**. Rename the new column to **DNS Delta**. Click **OK**.

Step 6: By clicking the new **DNS Delta** column heading twice, it gets sorted from high to low.



A significant delay preceding Packet 11 of this trace file can be observed —almost 1.3 seconds. We need to investigate further to determine why the DNS server is working slowly.

Topic 128: Wireshark Lab 35

This topic describes how to create a Button to detect High DNS response times in Wireshark.

Domain Name System is a distributed database implemented in a hierarchy of many DNS servers. It is an application-layer protocol that allows hosts to query the distributed database and to resolve names (address/name translation). Consider a browser (i.e. an HTTP client) running on some a host A, who requests www.wireshark.org. In order for host A to be able to send an HTTP request message, host A must first obtain IP address of server. The host A runs the client side of the DNS application. The browser extracts the hostname from the URL and passes it to the client side of the DNS application. The DNS client sends a query using UDP to the DNS server listening at port 53. The DNS client eventually receives a reply which includes the IP address for the host name. Once the browser receives the IP address from DNS, it can initiate a TCP connection to the HTTP server process located at port 80 at that IP address.

Let's create a button to detect DNS response times larger than 1 second.

Step 1: Open *tr-dns-slow.pcapng*.

Step 2: Type **dns.time > 1** in the display filter area and then click **Save**.

Step 3: Name your button **DNS Delay** and click **OK**.

Step 4: Click your new **DNS Delay** button. We observe that Packet 11 is the only packet that matched this filter i.e Packet 11 contains the largest DNS delay time.

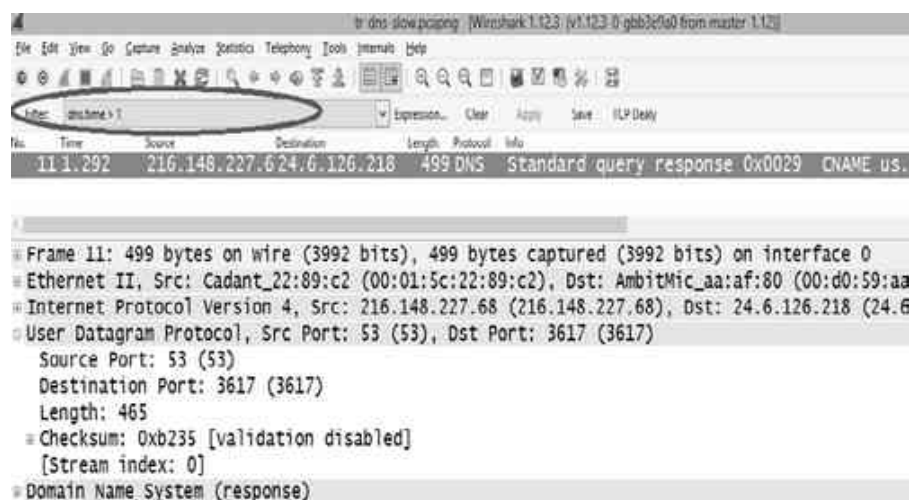


Figure 128.1: Create a Button to Detect High DNS Response Times

Topic 129: Wireshark Lab 36

In this topic, we describe how to plot DNS response times in Wireshark.

To highlight DNS delays in a trace file, let's create a graph.

Step 1: Open *tr-dns-slow.pcapng*.

Step 2: Select **Statistics | IO Graph**.

Step 3: In the Y Axis Unit area, select **Advanced...**

Step 4: Select the **MAX(*)** Graph 1 Calc option and enter **dns.time** in the Calc area.

Step 5: Click the **Graph 1** button to plot results.

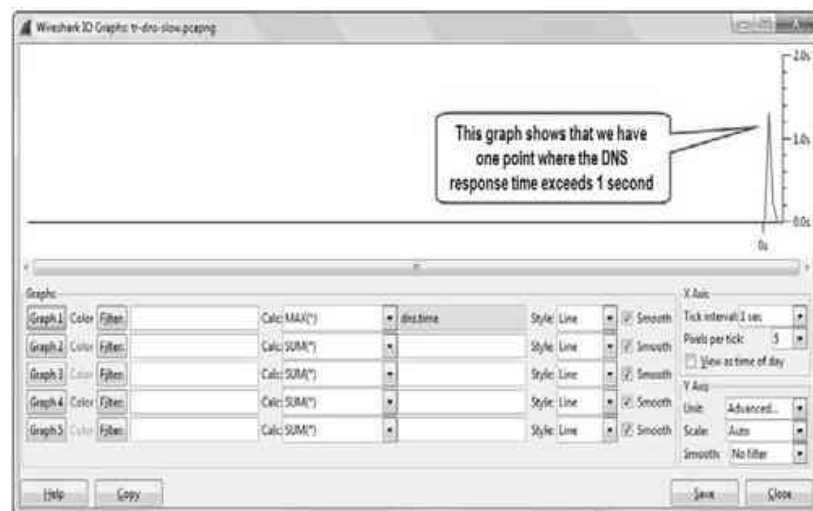


Figure 129.1: Graph DNS Response Times

Step 6: Click the highest point in the graph. Wireshark takes us to that packet.

No.	Time	Source	Destination	Length	Protocol	Info
1	0.000	24.6.126.218	216.148.227.6	73	DNS	Standard query 0x0029 A www.ncmec.org
2	1.000	24.6.126.218	204.127.202.4	73	DNS	Standard query 0x0029 A www.ncmec.org
3	1.107	204.127.202.4	24.6.126.218	499	DNS	Standard query response 0x0029 CNAME us.mt
4	1.211	24.6.126.218	146.82.218.13	62	TCP	3618->80 [SYN] Seq=0 Win=64512 Len=0 MSS=146
5	1.189	146.82.218.13	24.6.126.218	62	TCP	80->3618 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0
6	1.189	24.6.126.218	146.82.218.13	54	TCP	3618->80 [ACK] Seq=1 Ack=1 Win=64512 Len=0
7	1.191	24.6.126.218	146.82.218.13	459	HTTP	GET / HTTP/1.1
8	1.275	146.82.218.13	24.6.126.218	60	TCP	80->3618 [ACK] Seq=1 Ack=406 Win=6432 Len=0
9	1.291	146.82.218.13	24.6.126.218	1514	HTTP	HTTP/1.1 200 OK (text/html)
10	1.292	24.6.126.218	146.82.218.13	54	TCP	3618->80 [ACK] Seq=406 Ack=1461 Win=64512 Len=0

We can see it is packet 3. It is a DNS query response packet has caused the largest delay. Sorting **dns.time** column can help us to detect delays more quickly.

Topic 130: Wireshark Lab 37

This topic discusses how to reassemble TCP streams by disabling the Allow Subdissector in Wireshark.

The delta time between an HTTP request (e.g. GET) and the response (e.g. 200 OK) is the HTTP response time. In Wireshark, HTTP response time can be estimated with **http.time** field. If a web server is overloaded with the connection, or requests, or needs to consult another server to answer client's requests. This can result in high HTTP response times. If enabled, Wireshark measures time from HTTP Request to the final data packet of the response. If you want to estimate the total time taken by downloading an object, then enable this setting. By disabling it, we can learn how quickly the server responded. There are two methods to change this TCP preference setting.

Step 1: Open *tr-http-pcaprnet101.pcapng*.

Step 2: Right-click on the **TCP header** in the Packet Details of Packet 5. Select **Protocol Preferences**. Then, uncheck *Allow subdissector to reassemble TCP streams*.

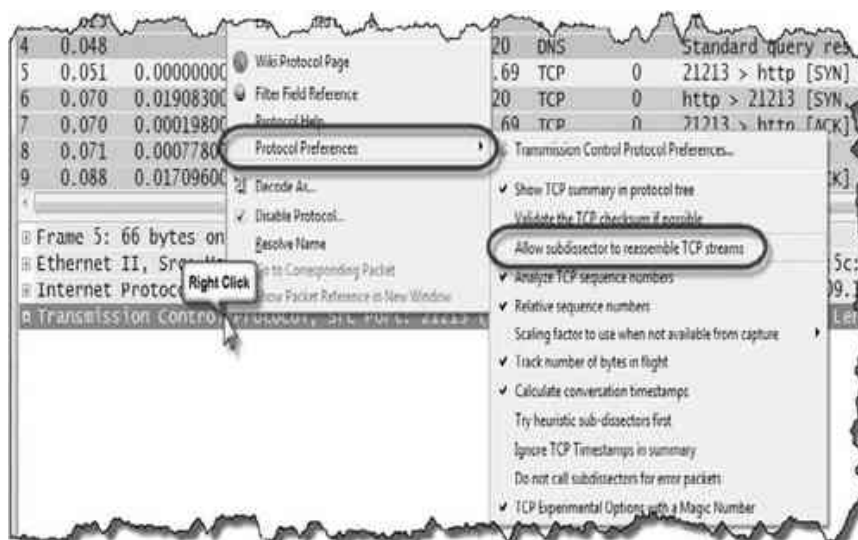
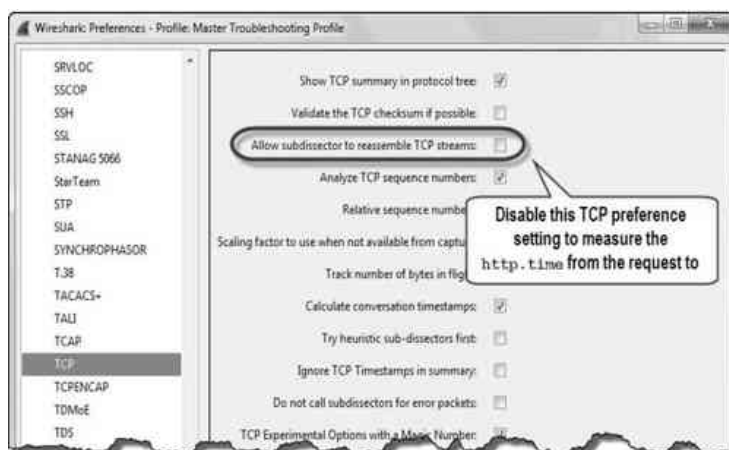


Figure 130.1: Disable the *Allow Subdissector to Reassemble TCP Streams* Preference Setting

Step 3: The 2nd method is: Select **Edit | Preferences | (+) Protocols | TCP**.



Topic 131: Wireshark Lab 38

This topic describes how to locate HTTP response times in Wireshark.

Step 1: Open *tr-http-pcapnet101.pcapng*.

Step 2: Only the HTTP response packets contain the **http.time** field. In this trace file, the first HTTP response packet is Packet 10 (HTTP 303 See Other).

No.	Time	Source	Destination	Length	Protocol	Info
1	0.000	24.6.173.220	75.75.75.75	73	DNS	Standard query 0xc3bf A www.pcapr.net
2	0.021	75.75.75.75	24.6.173.220	89	DNS	Standard query response 0xc3bf A 209.133.32.6
3	0.023	24.6.173.220	75.75.75.75	73	DNS	Standard query 0x406e AAAA www.pcapr.net
4	0.048	75.75.75.75	24.6.173.220	146	DNS	Standard query response 0x406e
5	0.051	24.6.173.220	209.133.32.69	66	TCP	21213->80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 W
6	0.070	209.133.32.69	24.6.173.220	66	TCP	80->21213 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0
7	0.070	24.6.173.220	209.133.32.69	54	TCP	21213->80 [ACK] Seq=1 Ack=1 Win=65700 Len=0
8	0.071	24.6.173.220	209.133.32.69	341	HTTP	GET / HTTP/1.1
9	0.088	209.133.32.69	24.6.173.220	60	TCP	80->21213 [ACK] Seq=1 Ack=288 Win=6912 Len=0
10	0.097	209.133.32.69	24.6.173.220	357	HTTP	HTTP/1.1 303 See other
11	0.098	209.133.32.69	24.6.173.220	60	TCP	80->21213 [FIN, ACK] Seq=304 Ack=288 Win=6912 L
12	0.099	24.6.173.220	209.133.32.69	54	TCP	21213->80 [ACK] Seq=288 Ack=305 Win=65396 Len=0
13	0.099	24.6.173.220	209.133.32.69	54	TCP	21213->80 [FIN, ACK] Seq=288 Ack=305 Win=65396
14	0.105	24.6.173.220	209.133.32.69	66	TCP	21214->80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 W

Figure 131.1: Add/Sort an HTTP Response Time Column to Find HTTP Response Times

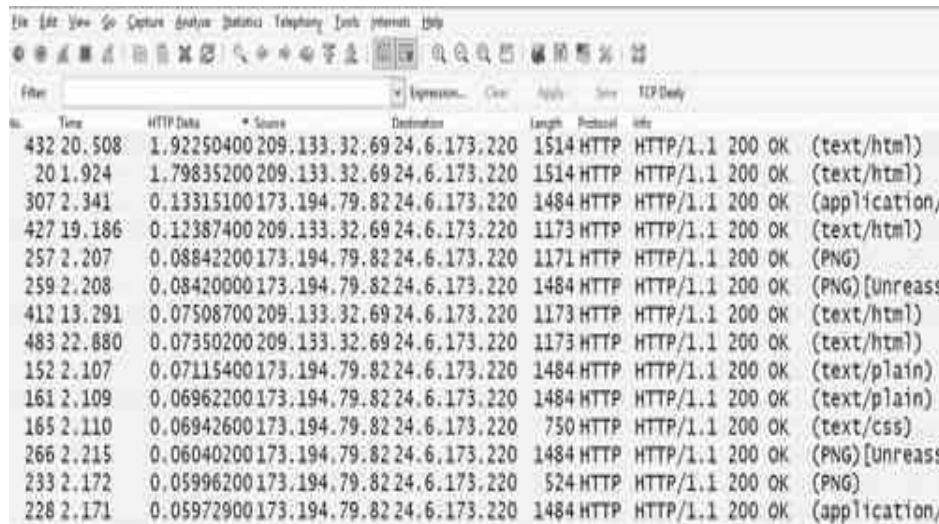
Let's use this packet to create our **http.time** column. In the Packet Details pane of Packet 10, right-click on the **Hypertext Transfer Protocol** section and select **Expand Subtrees**.

Step 3: Locate the **[Time since request: 0.026416000 seconds]** line and right-click on it and select **Apply as Column**.

Step 4: Your newly created column appears to the left of the Info column. Click and drag your new column to a place with better visibility.

Step 5: Right-click on the column heading and select **Edit Column Details**. Rename this new column to **HTTP Delta**. Click **OK**.

Step 6: Sort HTTP responses from high to low by clicking your new **HTTP Delta** column heading twice. Packets with the largest delays will appear at the top of the list.



No.	Time	HTTP Delta	Source	Destination	Length	Protocol	Info
432	20.508	1.92250400	209.133.32.69	24.6.173.220	1514	HTTP	HTTP/1.1 200 OK (text/html)
20	1.924	1.79835200	209.133.32.69	24.6.173.220	1514	HTTP	HTTP/1.1 200 OK (text/html)
307	2.341	0.13315100	173.194.79.82	24.6.173.220	1484	HTTP	HTTP/1.1 200 OK (application/)
427	19.186	0.12387400	209.133.32.69	24.6.173.220	1173	HTTP	HTTP/1.1 200 OK (text/html)
257	2.207	0.08842200	173.194.79.82	24.6.173.220	1171	HTTP	HTTP/1.1 200 OK (PNG)
259	2.208	0.08420000	173.194.79.82	24.6.173.220	1484	HTTP	HTTP/1.1 200 OK (PNG)[Unreass
412	13.291	0.07508700	209.133.32.69	24.6.173.220	1173	HTTP	HTTP/1.1 200 OK (text/html)
483	22.680	0.07350200	209.133.32.69	24.6.173.220	1173	HTTP	HTTP/1.1 200 OK (text/html)
152	2.107	0.07115400	173.194.79.82	24.6.173.220	1484	HTTP	HTTP/1.1 200 OK (text/plain)
161	2.109	0.06962200	173.194.79.82	24.6.173.220	1484	HTTP	HTTP/1.1 200 OK (text/plain)
165	2.110	0.06942600	173.194.79.82	24.6.173.220	750	HTTP	HTTP/1.1 200 OK (text/css)
266	2.215	0.06040200	173.194.79.82	24.6.173.220	1484	HTTP	HTTP/1.1 200 OK (PNG)[Unreass
233	2.172	0.05996200	173.194.79.82	24.6.173.220	524	HTTP	HTTP/1.1 200 OK (PNG)
228	2.171	0.05972900	173.194.79.82	24.6.173.220	1484	HTTP	HTTP/1.1 200 OK (application/)

We can observe that HTTP response time from the HTTP server (209.133.32.69) is over 1.7 seconds twice in this trace file. Those packets are bearing numbers 432 and 20.

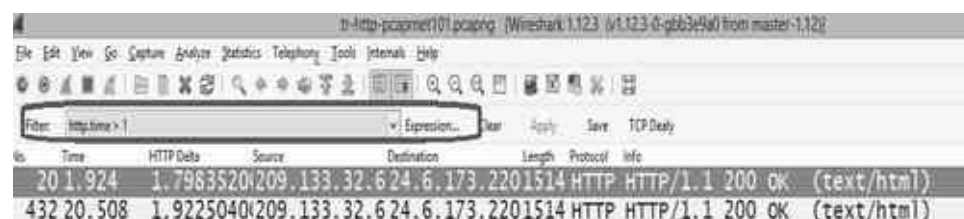
Topic 132: Wireshark Lab 39

In this topic, we create a button to quickly detect high HTTP response times in Wireshark.

Wireshark allows us to create buttons based on filter expressions. It is one of the most powerful features to improve your efficiency in locating network problems.

Step 1: Open *tr-http-pcapnet101.pcapng*.

Step 2: Type **http.time > 1** in the display filter area and then, click **Save**.



No.	Time	HTTP Delta	Source	Destination	Length	Protocol	Info
20	1.924	1.79835200	209.133.32.69	24.6.173.220	1514	HTTP	HTTP/1.1 200 OK (text/html)
432	20.508	1.92250400	209.133.32.69	24.6.173.220	1514	HTTP	HTTP/1.1 200 OK (text/html)

Step 3: When you click **save**, Wireshark prompts to name the button. Name it **HTTP Delay** and then click **OK**.



Always remember to clear your display filter when you are done reviewing the results.

Step 4: As you click your new **HTTP Delay** button, you will find two packets of this trace file matching the filter—Packet 20 and Packet 432.

Topic 133: Wireshark Lab 40

In this topic, we plot HTTP response times in Wireshark.

Wireshark allows a user to define up to five differently colored graphs. We can make the following configurations:

Graphs

- **Graph 1-5:** enable the specific graph 1-5 (only graph 1 is enabled by default).
- **Color:** (cannot be changed)
- **Filter:** a display filter for this graph
- **Style:**(Line/Impulse/FBar/Dot)

X Axis

- **Tick interval:** an interval in x direction lasts (10/1 mins or 10/1/0.1/0.01/0.001s)
- **Pixels per tick:** use 10/5/2/1 pixels per tick interval
- **View as time of day:** instead of sec or mins, view x as time of day

Y Axis

- **Unit:** the unit for y direction (Packets/Tick, Bytes/Tick, Bits/Tick, Advanced...)
- **Scale:** the scale for the y unit (Logarithmic,Auto,10,20,50,100,200,500,...)

Let's Graph HTTP Response Times.

Step 1: Open *tr-http-pcapnet101.pcapng*.

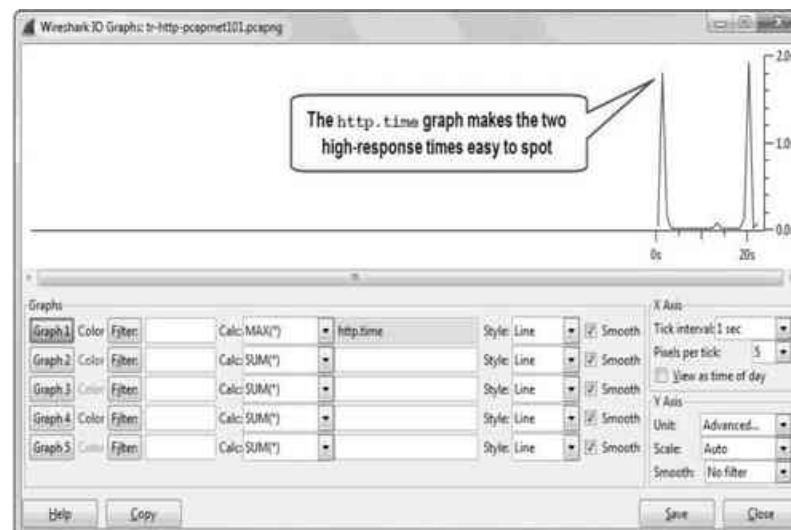
No.	Time	Source	Destination	Length	Protocol	Info
1	0.000	24.6.173.220	75.75.75.75	73	DNS	Standard query 0xc3bf A w
2	0.021	75.75.75.75	24.6.173.220	89	DNS	Standard query response 0x
3	0.023	24.6.173.220	75.75.75.75	73	DNS	Standard query 0x406e AAA
4	0.048	75.75.75.75	24.6.173.220	146	DNS	Standard query response 0x
5	0.051	24.6.173.220	209.133.32.6	66	TCP	21213-80 [SYN] Seq=0 win=8
6	0.070	209.133.32.6	24.6.173.220	66	TCP	80-21213 [SYN, ACK] Seq=0
7	0.070	24.6.173.220	209.133.32.6	54	TCP	21213-80 [ACK] Seq=1 Ack=1
8	0.071	24.6.173.220	209.133.32.6	341	HTTP	GET / HTTP/1.1
9	0.088	209.133.32.6	24.6.173.220	60	TCP	80-21213 [ACK] Seq=1 Ack=2
10	0.097	209.133.32.6	24.6.173.220	357	HTTP	HTTP/1.1 303 see other
11	0.098	209.133.32.6	24.6.173.220	60	TCP	80-21213 [FIN, ACK] Seq=30
12	0.099	24.6.173.220	209.133.32.6	54	TCP	21213-80 [ACK] Seq=288 Ack
13	0.099	24.6.173.220	209.133.32.6	54	TCP	21213-80 [FIN, ACK] Seq=28
14	0.105	24.6.173.220	209.133.32.6	66	TCP	21214-80 [SYN] Seq=0 Win=8

Step 2: Select **Statistics | IO Graph**.

Step 3: In the Y Axis Unit area, select **Advanced...**

Step 4: Select the **MAX(*)** Graph 1 Calc option and then, enter **http.time** in the Calc area.

Step 5: To plot your results, click the **Graph 1** button.



If you click the highest points in the graph, Wireshark will take you to those packets which have encountered highest delays. Packet 20 and Packet 432 show that the HTTP response time from the HTTP server (209.133.32.69) was over 1.7 seconds

Topic 134: Wireshark Lab 41

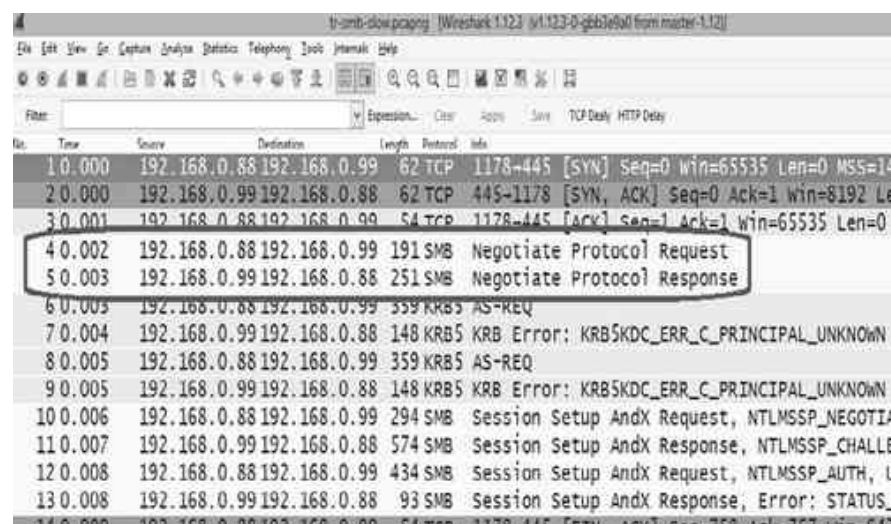
In this topic, we add Server Message Block (SMB) response time column in Wireshark.

What is a Server Message Block (SMB)?

On Windows-based networks, Microsoft employs a file sharing protocol called Server Message Block (SMB). SMB and SMB version 2 are the two versions that are commonly employed on Windows-based networks. SMB is a request/response protocol. Its response codes are available on Microsoft's Open Specification site www.microsoft.com/openspecifications/. SMB delays can be estimated by looking at the **smb.time** field in the SMB response packets.

Step 1: Open *tr-smb-slow.pcapng*.

Step 2: The first SMB response packet is Packet 5 (Negotiate Protocol Response).

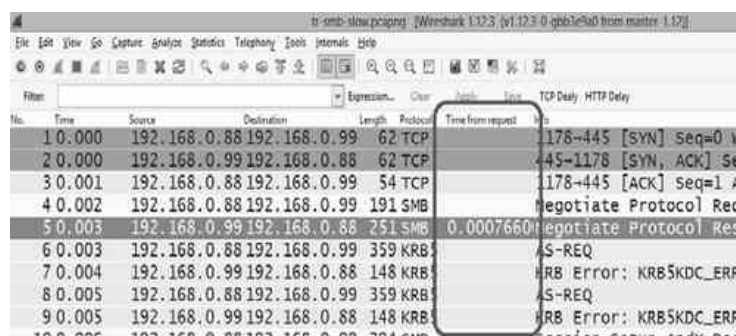
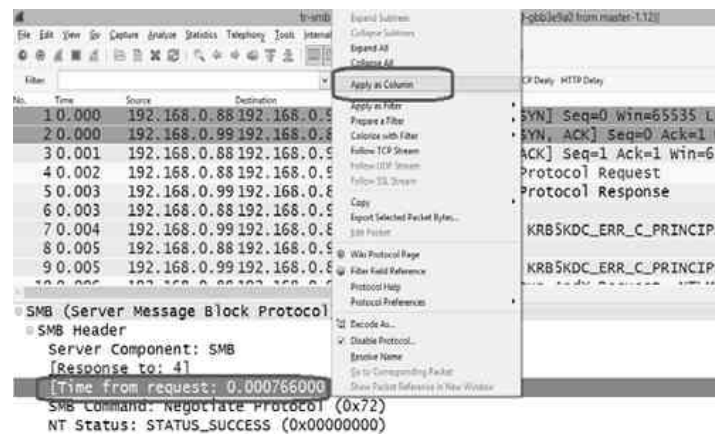


The image shows a Wireshark packet capture window titled 'tr-smb-slow.pcapng [Wireshark 1.12.3 (v1.12.3-0-gbb3e9d0) from master-1.12]'. The packet list pane shows several packets. Packet 5 is highlighted with a red box. The packet details pane shows the SMB section expanded, and the SMB Header section expanded, showing the 'smb.time' field.

No.	Time	Source	Destination	Length	Protocol	Info
1	0.000	192.168.0.88	192.168.0.99	62	TCP	1178→445 [SYN] Seq=0 win=65535 Len=0 MSS=1460
2	0.000	192.168.0.99	192.168.0.88	62	TCP	445→1178 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0
3	0.001	192.168.0.88	192.168.0.99	54	TCP	1178→445 [ACK] Seq=1 Ack=1 Win=65535 Len=0
4	0.002	192.168.0.88	192.168.0.99	191	SMB	Negotiate Protocol Request
5	0.003	192.168.0.99	192.168.0.88	251	SMB	Negotiate Protocol Response
6	0.003	192.168.0.88	192.168.0.99	359	KRB5	AS-REQ
7	0.004	192.168.0.99	192.168.0.88	148	KRB5	KRB Error: KRB5KDC_ERR_C_PRINCIPAL_UNKNOWN
8	0.005	192.168.0.88	192.168.0.99	359	KRB5	AS-REQ
9	0.005	192.168.0.99	192.168.0.88	148	KRB5	KRB Error: KRB5KDC_ERR_C_PRINCIPAL_UNKNOWN
10	0.006	192.168.0.88	192.168.0.99	294	SMB	Session Setup AndX Request, NTLMSSP_NEGOTIATE
11	0.007	192.168.0.99	192.168.0.88	574	SMB	Session Setup AndX Response, NTLMSSP_CHALLENGE
12	0.008	192.168.0.88	192.168.0.99	434	SMB	Session Setup AndX Request, NTLMSSP_AUTH, U
13	0.008	192.168.0.99	192.168.0.88	93	SMB	Session Setup AndX Response, Error: STATUS_
14	0.009	192.168.0.88	192.168.0.99	54	TCP	1178→445 [ACK] Seq=1 Ack=1 Win=65535 Len=0

The **smb.time** field exists only in SMB response packets. Let's use packet 5 to create a **smb.time** column. Expand the **SMB** section and the **SMB Header** section of Packet 5 in the Packet Details pane.

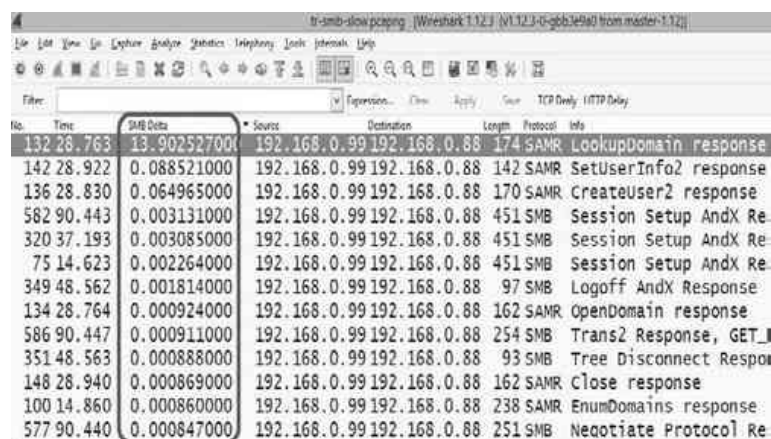
Step 3: Right-click on the **[Time from request: 0.000766000 seconds]** line and click **Apply as Column**.



Step 4: The newly created column appears to the left of the Info column. Click and drag your new column to a suitable location where visibility is much better.

Step 5: Wireshark assigned the new **smb.time** column the label *Time from request*. On column heading click right and then, select **Edit Column Details**. Rename it's title to **SMB Delta**. Click **OK**.

Step 6: To examine the SMB response packets with the largest delays, click the newly created **SMB Delta** column heading twice. This will sort packets from high to low.

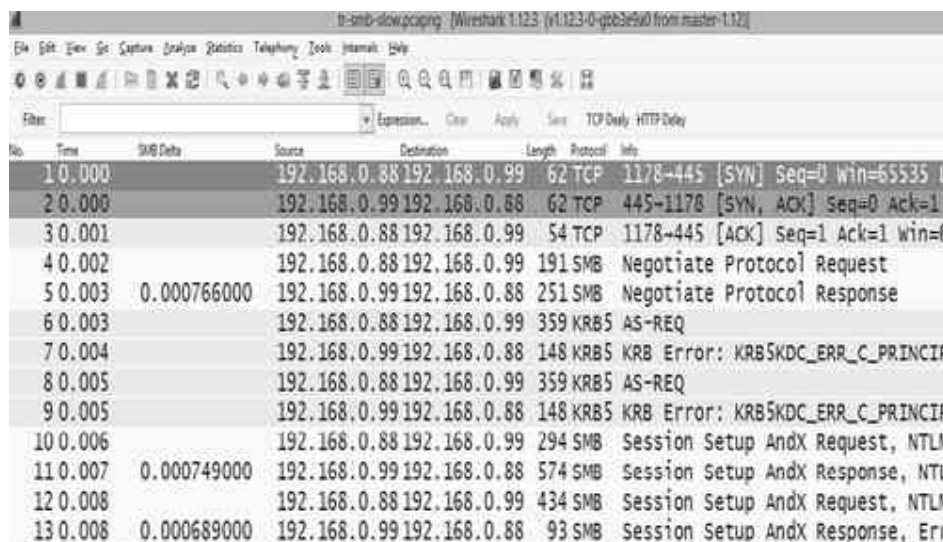


Topic 135: Wireshark Lab 42

In this topic, we examine Server Message Block (SMB) statistics in Wireshark.

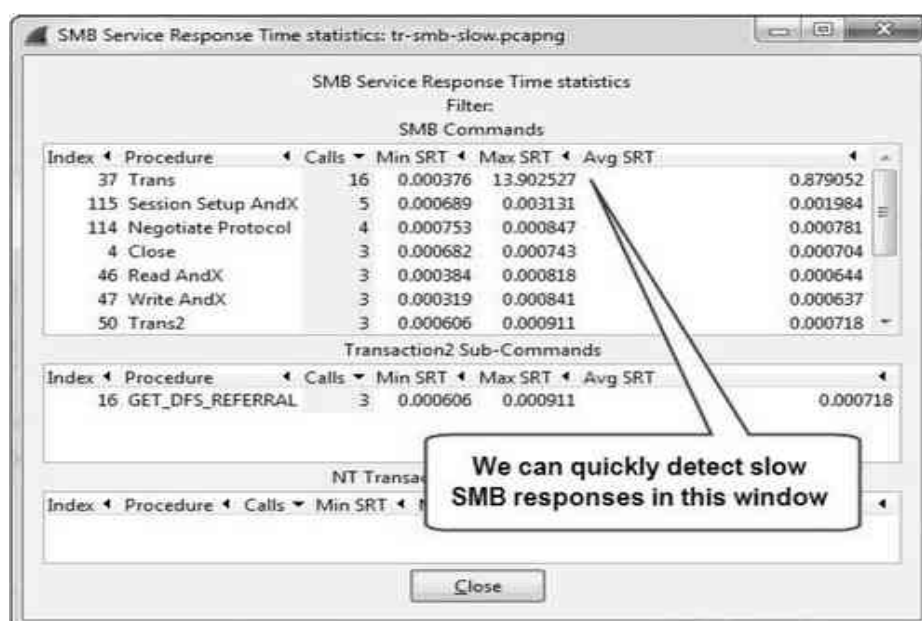
Service response times, including SMB and SMB2 response times are created and tracked by Wireshark.

Step 1: Open *tr-smb-slow.pcapng*.



No.	Time	SMB Delta	Source	Destination	Length	Protocol	Info
1	0.000		192.168.0.88	192.168.0.99	62	TCP	1178→445 [SYN] Seq=0 Win=65535 L
2	0.000		192.168.0.99	192.168.0.88	62	TCP	445→1178 [SYN, ACK] Seq=0 Ack=1
3	0.001		192.168.0.88	192.168.0.99	54	TCP	1178→445 [ACK] Seq=1 Ack=1 Win=6
4	0.002		192.168.0.88	192.168.0.99	191	SMB	Negotiate Protocol Request
5	0.003	0.000766000	192.168.0.99	192.168.0.88	251	SMB	Negotiate Protocol Response
6	0.003		192.168.0.88	192.168.0.99	359	KRB5	AS-REQ
7	0.004		192.168.0.99	192.168.0.88	148	KRB5	KRB Error: KRB5KDC_ERR_C_PRINCIP
8	0.005		192.168.0.88	192.168.0.99	359	KRB5	AS-REQ
9	0.005		192.168.0.99	192.168.0.88	148	KRB5	KRB Error: KRB5KDC_ERR_C_PRINCIP
10	0.006		192.168.0.88	192.168.0.99	294	SMB	Session Setup AndX Request, NTLM
11	0.007	0.000749000	192.168.0.99	192.168.0.88	574	SMB	Session Setup AndX Response, NTL
12	0.008		192.168.0.88	192.168.0.99	434	SMB	Session Setup AndX Request, NTLM
13	0.008	0.000689000	192.168.0.99	192.168.0.88	93	SMB	Session Setup AndX Response, Err

Step 2: Select **Statistics | Service Response Time | SMB**. As Wireshark prompts you, Click **Create Stat**. For a trace file, the SMB Service Response Time statistics window shows the minimum, maximum and average Service Response Time (SRT).



Index	Procedure	Calls	Min SRT	Max SRT	Avg SRT
37	Trans	16	0.000376	13.902527	0.879052
115	Session Setup AndX	5	0.000689	0.003131	0.001984
114	Negotiate Protocol	4	0.000753	0.000847	0.000781
4	Close	3	0.000682	0.000743	0.000704
46	Read AndX	3	0.000384	0.000818	0.000644
47	Write AndX	3	0.000319	0.000841	0.000637
50	Trans2	3	0.000606	0.000911	0.000718

Index	Procedure	Calls	Min SRT	Max SRT	Avg SRT
16	GET_DFS_REFERRAL	3	0.000606	0.000911	0.000718

We can quickly detect slow SMB responses in this window

This SMB Service Response Time statistics window provides a list of all the request procedures.

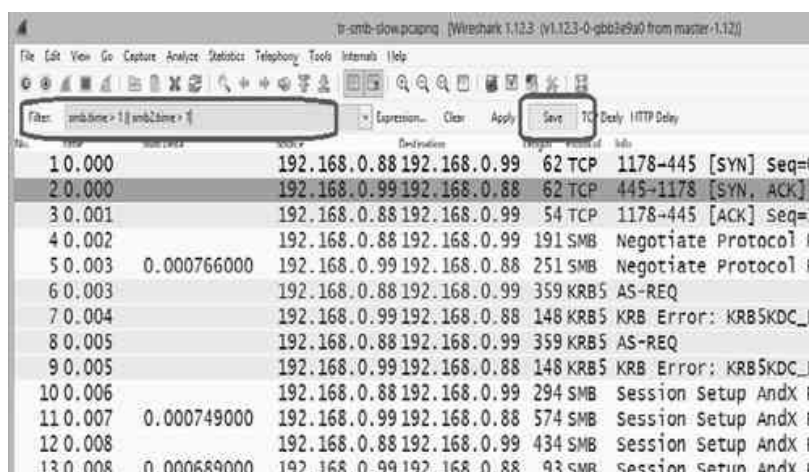
Topic 136: Wireshark Lab 43

In this topic, we create a button to detect Server Message Block (SMB) Response Times in Wireshark.

In Wireshark, SMB and SMB2 response times can be estimated by using **smb.time** and **smb2.time**. Let's learn how to create a button to detect SMB and SMB2 response times larger than 1 second.

Step 1: Open *tr-smb-slow.pcapng*.

Step 2: Type **smb.time > 1 || smb2.time > 1** in the display filter area. Click **Save**.



Step 3: You can name your button **SMB/SMB2 Delay** and then, click **OK**.

Step 4: Click your newly created **SMB Delay** button. We find that Packet 132 is the only packet that matched your filter.



Topic 137: Wireshark Lab 44

This topic plots Server Message Block (SMB) Response Times in Wireshark.

Let's learn how to plot SMB response times in Wireshark.

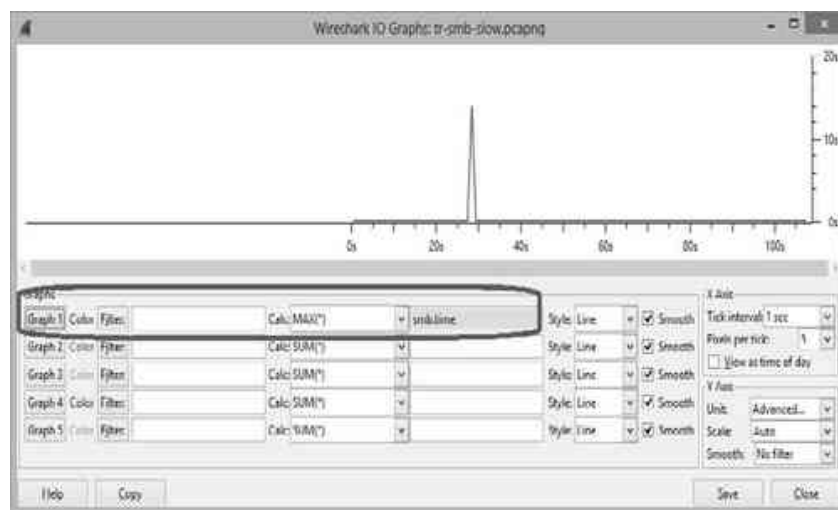
Step 1: Open *tr-smb-slow.pcapng*.

Step 2: Select **Statistics | IO Graph**.

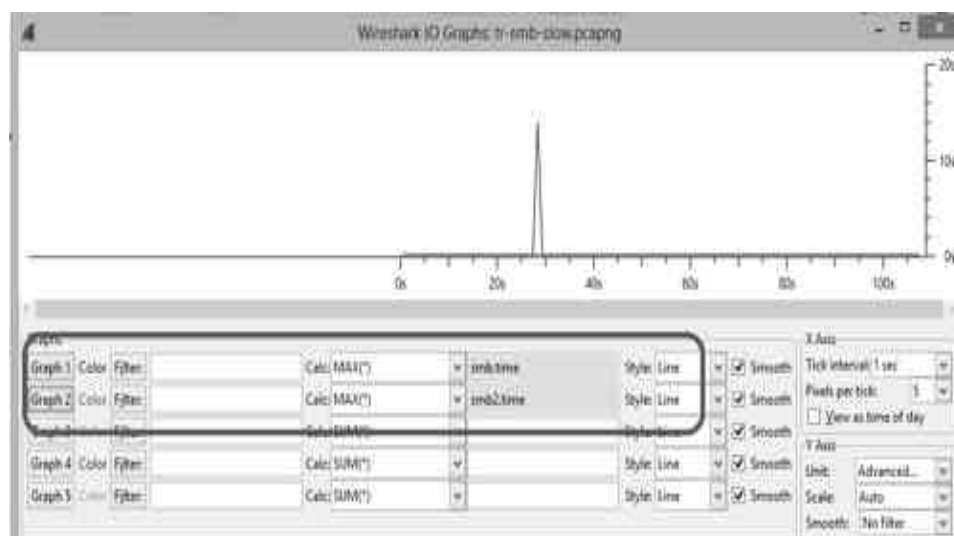
Step 3: In the Y Axis **Unit** area, select **Advanced...**

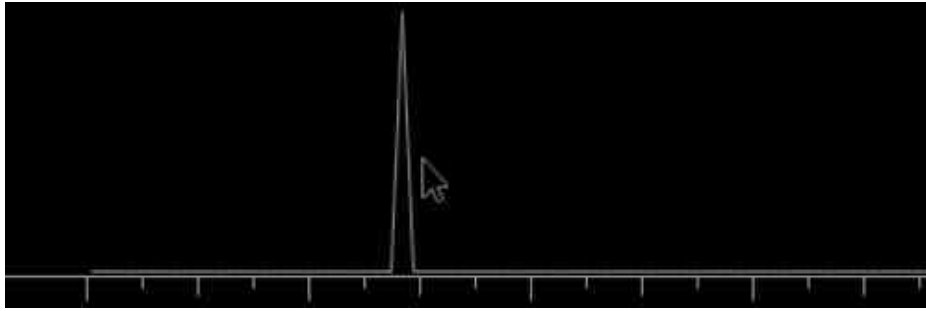
Step 4: Select the **MAX(*)** Graph 1 Calc option and enter **smb.time** in the Calc area.

Step 5: Click the **Graph 1** button to plot the SMB response times.



Step 6: Since SMB and SMB2 use different time fields, we have to create two graphs; one for each delta time.





Step 7: As you click on the highest points in the graph, Wireshark will jump to those packets in the main window for further analysis.

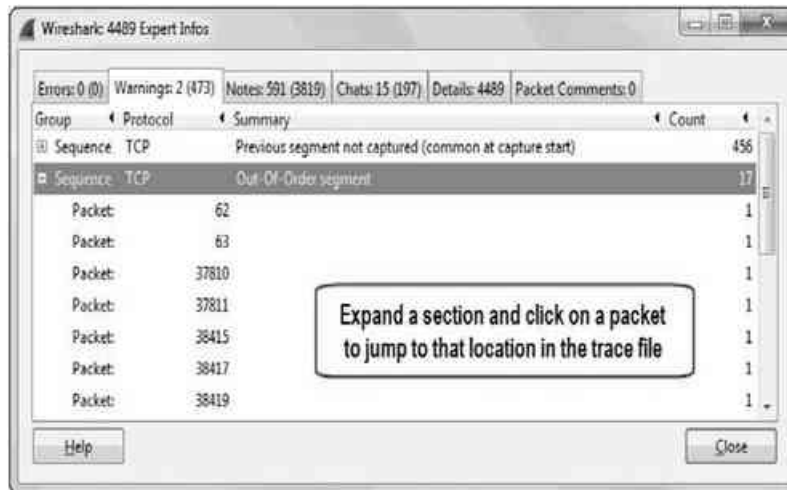
Topic 138: Wireshark's Expert Infos System

This topic provides an overview of Wireshark's Expert Infos System.

There are two buttons and three columns in Wireshark's Status Bar. The first button is named the **Expert Infos** button. Wireshark generates alerts for you to numerous network concerns seen in the trace file as well as packet comments in the Expert Infos window. Expert infos are only a hint and help both novice and expert users to find probable network problems a lot faster, compared to scanning the packet list "manually". The Expert classifies information into 6 categories:

- Errors: red
- Warnings: yellow
- Notes: cyan
- Chats: blue
- Details: grey
- Packet Comments: green

When you click the **Expert Infos** button in the bottom left corner of the Status Bar, the Expert Infos window will open.



If we understand the causes of the errors, warnings, and notes, then this will help us to figure out what is affecting the network performance. We next look at examples of various commonly occurring Expert warnings, notes, and chats.

Chats: information about usual workflow: e.g. a TCP packet with the SYN flag set

Notes: notable things: e.g. an application returned an “usual” error code like HTTP 404

Warnings: e.g. application returned an “unusual” error code like a connection problem.

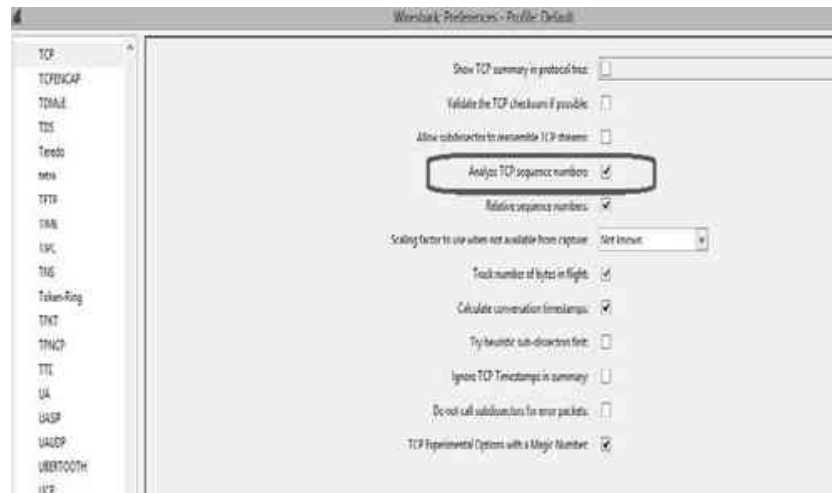
Topic 139: Wireshark's Packet Loss Detection

This topic provides an overview of Wireshark's Packet loss detection process.

Most of the time packets are dropped by interconnecting devices such as a switch, a router, a firewall i.e. anything that makes forwarding decisions.

What Causes Packet Loss: The common reasons are a switch or router is overloaded. It is unable to keep up with the required packet forwarding rate. The device is simply faulty.

How Wireshark's detects Packet Loss: The TCP sequencing process is used by Wireshark to detect lost packets. You can also turn off this feature by disabling the **Analyze TCP Sequence Numbers** TCP Preference setting. Click the Preferences button, then expand **Protocols** and choose **TCP**.



Wireshark creates a value called *nextseq* (displayed in the Next Sequence Number field). Its value is determined by adding together the Sequence Number field value of each packet to the number of data bytes in it. Therefore, the next expected sequence number from the sender is available in *nextseq*. Wireshark assumes that one or more packets have been lost when it sees sequence number jump beyond the *nextseq* value.

Topic 140: Packet Loss Recovery Methods

In this topic, we understand Fast Recovery and Retransmission Timeout (RTO) methods for Packet loss. An Internet application developer has two choices for the transport layer protocol: Transmission Control Protocol (TCP), or User Datagram Protocol (UDP). TCP provides a packet loss recovery mechanism while UDP does not. Based on this feature, there are applications which can and which cannot tolerate data loss. File transfer, e-mail, and Web documents, instant messaging are examples of those applications which cannot tolerate loss. Loss-tolerant apps include real-time audio/video, stored audio/video, and interactive games.

Packet Loss Recovery Method #1—Fast Recovery

This method can be used only if the receiver supports Fast Recovery. In this method, when the receiver observes a jump in the sequence number value, it will immediately begin sending Duplicate Acknowledgments. Upon receipt of four identical ACKs (the original and three Duplicate ACKs), the sender will retransmit the packet that has been lost.

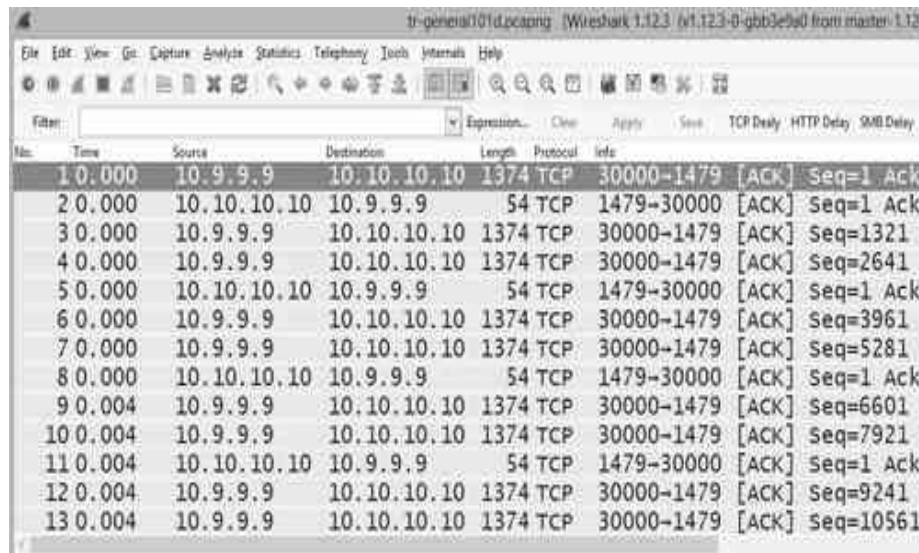
Packet Loss Recovery Method #2—Sender Retransmission Timeout (RTO)

In this method, a sender maintains a timer for a packet. If the packet has not been acknowledged within Retransmission Timeout (RTO) value, the sender retransmits the packet.

Topic 141: Wireshark Lab 45

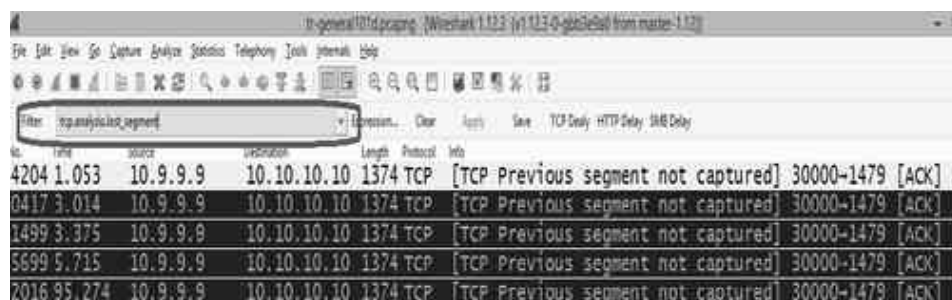
In this topic, we apply a Filter to count previous segments in Wireshark.

Step 1: Open *tr-general101d.pcapng*.



No.	Time	Source	Destination	Length	Protocol	Info
1	0.000	10.9.9.9	10.10.10.10	1374	TCP	30000-1479 [ACK] Seq=1 Ack=
2	0.000	10.10.10.10	10.9.9.9	54	TCP	1479-30000 [ACK] Seq=1 Ack=
3	0.000	10.9.9.9	10.10.10.10	1374	TCP	30000-1479 [ACK] Seq=1321 A
4	0.000	10.9.9.9	10.10.10.10	1374	TCP	30000-1479 [ACK] Seq=2641 A
5	0.000	10.10.10.10	10.9.9.9	54	TCP	1479-30000 [ACK] Seq=1 Ack=
6	0.000	10.9.9.9	10.10.10.10	1374	TCP	30000-1479 [ACK] Seq=3961 A
7	0.000	10.9.9.9	10.10.10.10	1374	TCP	30000-1479 [ACK] Seq=5281 A
8	0.000	10.10.10.10	10.9.9.9	54	TCP	1479-30000 [ACK] Seq=1 Ack=
9	0.004	10.9.9.9	10.10.10.10	1374	TCP	30000-1479 [ACK] Seq=6601 A
10	0.004	10.9.9.9	10.10.10.10	1374	TCP	30000-1479 [ACK] Seq=7921 A
11	0.004	10.10.10.10	10.9.9.9	54	TCP	1479-30000 [ACK] Seq=1 Ack=
12	0.004	10.9.9.9	10.10.10.10	1374	TCP	30000-1479 [ACK] Seq=9241 A
13	0.004	10.9.9.9	10.10.10.10	1374	TCP	30000-1479 [ACK] Seq=10561

Step 2: In the display filter area, enter the filter **tcp.analysis.lost_segment**. Click **Apply**.



No.	Time	Source	Destination	Length	Protocol	Info
4204	1.053	10.9.9.9	10.10.10.10	1374	TCP	[TCP Previous segment not captured] 30000-1479 [ACK] S
4417	3.014	10.9.9.9	10.10.10.10	1374	TCP	[TCP Previous segment not captured] 30000-1479 [ACK] S
1499	3.375	10.9.9.9	10.10.10.10	1374	TCP	[TCP Previous segment not captured] 30000-1479 [ACK] S
5699	5.715	10.9.9.9	10.10.10.10	1374	TCP	[TCP Previous segment not captured] 30000-1479 [ACK] S
2016	95.274	10.9.9.9	10.10.10.10	1374	TCP	[TCP Previous segment not captured] 30000-1479 [ACK] S

From the Status Bar, we can observe that Wireshark has detected packet loss five times in this trace file. In the Info column, we can see “Previous Segment Not Captured”. This warning means that Wireshark was not able to see the previous packet(s) in a TCP communication. Wireshark keeps a track of the packet ordering using TCP Sequence Numbers. From this, it can easily detect when packets are missing.

Step 3: As you finish analyzing the trace file, click **Clear** to remove your filter. Applying the filter **tcp.analysis.lost_segment** on the traffic of a trace file, we easily found that packet loss is occurring. To determine how many packets were actually lost, we need to examine the TCP sequence numbers.

Topic 142: Wireshark Lab 46

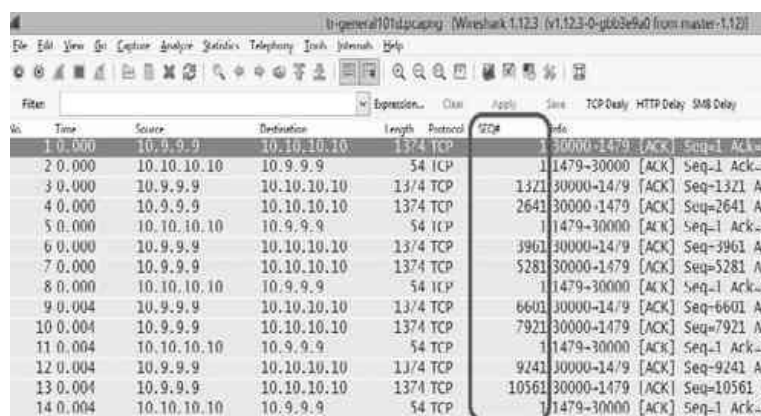
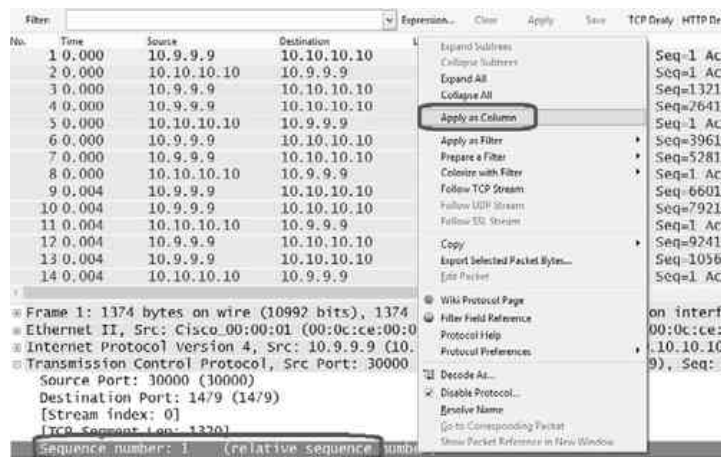
In this topic, we add TCP sequencing columns in Wireshark.

In this lab exercise, we will create three columns to help understand how TCP sequencing works in a trace file. Using these columns, we will be able to determine how many packets are actually lost when Wireshark displays **Previous Segment Not Captured**.

Step 1: Open *tr-general101d.pcapng*.

Step 2: Expand the **TCP header** in Packet 1.

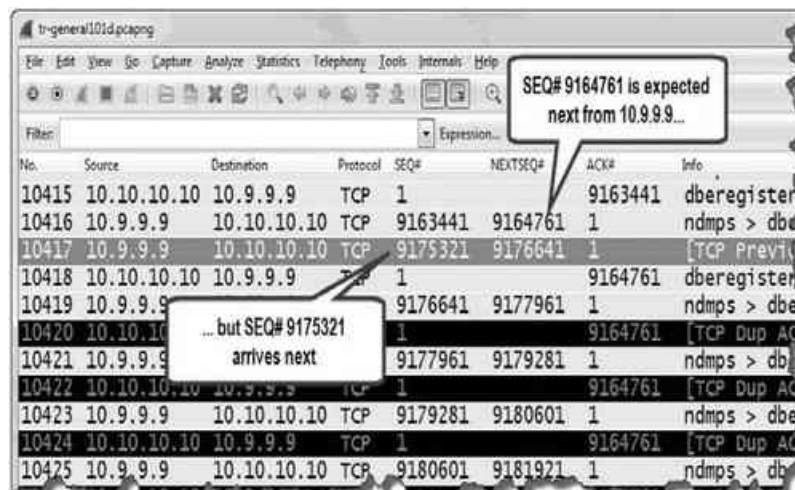
Step 3: Right-click on the **Sequence Number** field and select **Apply as Column**. Rename this column **SEQ#** by right-clicking on your new **Sequence Number** column and selecting **Edit Column Details**.



Step 4: Select **Apply as Column** after right-clicking on the **Next Sequence Number** field. Rename this column **NEXTSEQ#** by right-clicking on this newly created column and selecting **Edit Column Details**.

Step 5: Create the 3rd new column by right-clicking on the **Acknowledgment Number** field and select **Apply as Column**. Rename this column **ACK#** as we did in the previous step.

Step 6: Click the **Go To Packet** button on the Main Toolbar. As packet 10417 is one of the packets tagged with the *Previous Segment Not Captured* Expert indication. Enter 10417 and click **Jump To**. From the NEXTSEQ# column, we find that the next packet (after Packet 10,416) from 10.9.9.9 should use sequence number 9,164,761. But SEQ# is equal to 9,175,321 for the next packet from 10.9.9.9.



No.	Source	Destination	Protocol	SEQ#	NEXTSEQ#	ACK#	Info
10415	10.10.10.10	10.9.9.9	TCP	1		9163441	dberegister
10416	10.9.9.9	10.10.10.10	TCP	9163441	9164761	1	ndmps > dberegister
10417	10.9.9.9	10.10.10.10	TCP	9175321	9176641	1	[TCP Previous Segment Not Captured]
10418	10.10.10.10	10.9.9.9	TCP	1		9164761	dberegister
10419	10.9.9.9	10.10.10.10	TCP	9176641	9177961	1	ndmps > dberegister
10420	10.10.10.10	10.9.9.9	TCP	1		9164761	[TCP Duplicate ACK]
10421	10.9.9.9	10.10.10.10	TCP	9177961	9179281	1	ndmps > dberegister
10422	10.10.10.10	10.9.9.9	TCP	1		9164761	[TCP Duplicate ACK]
10423	10.9.9.9	10.10.10.10	TCP	9179281	9180601	1	ndmps > dberegister
10424	10.10.10.10	10.9.9.9	TCP	1		9164761	[TCP Duplicate ACK]
10425	10.9.9.9	10.10.10.10	TCP	9180601	9181921	1	ndmps > dberegister

Because of the mismatch, Wireshark tags Packet 10,417 with *Previous Segment Not Captured*. $9,175,321 - 9,164,761 = 10,560$ bytes have been lost. The TCP header of Packet 10,417 indicates that it contains 1,320 bytes of data. Assuming all the lost packets to be of 1,320 bytes long, $10,560 / 1,320 = 8$ packets were lost prior packet 10,417.

Topic 143: Wireshark Lab 47

In this topic, we build a bad TCP Filter expression in Wireshark.

We would like to create a button, which when clicked once will view many of the key TCP problems in a trace file.

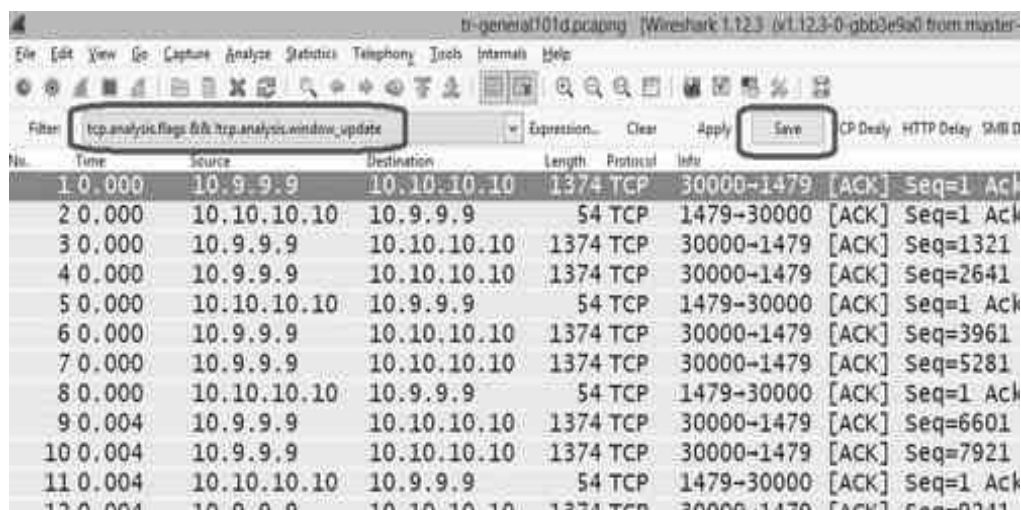
Step 1: Open *tr-general101d.pcapng*.

Step 2: Type the following expression in the display filter area: **tcp.analysis.flags && !tcp.analysis.window_update**. Click the **Coloring Rules** button and then, look for

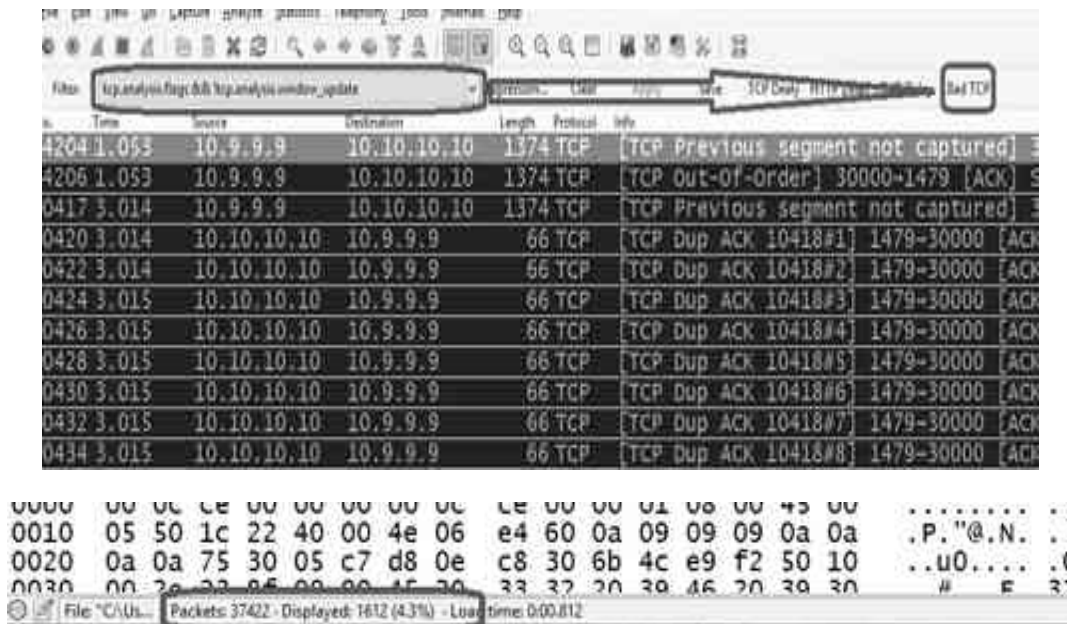
the **Bad TCP coloring rule**. The filter expression we typed in Step 2 is the same as the **Bad TCP coloring rule string**.



Step 3: Click **Save** on the display filter toolbar to create a new button. Name this new button **Bad TCP**.



Step 4: Click the newly created **Bad TCP** button.



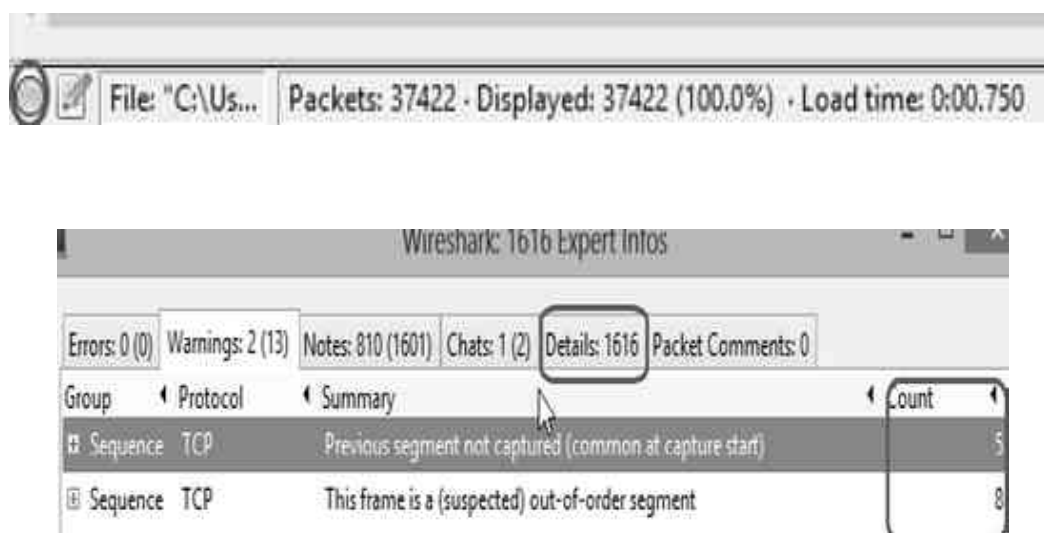
There are 1,612 packets (4.3% of the traffic) matching our Bad TCP button. These packets can badly affect e.g. a file transfer process to a noticeable extent. Click **Clear** to remove your filter when you are finished.

Topic 144: Wireshark Lab 48

In this topic, we find packet loss counts with Expert Infos in Wireshark.

Step 1: Open *tr-general101d.pcapng*.

Step 2: Click the **Expert Infos** button on the Status Bar.



Step 3: There are 1,616 Expert Infos items in the title bar. There are no entries under the Errors tab as Wireshark could not detect any serious problem in the trace file. When you click the **Warnings** tab, Wireshark has observed 5 packet losses. This is because of the sudden jump in the sequence numbers. Wireshark does not take into

account the size of the increase. From the **Warnings** tab, we can see that there are 8 instances of Out-of-Order segments. If you want to analyze a problem, expand the Expert Infos sections and click on that packet listed. Wireshark will jump to that problem in the trace file. In the end, close the Expert Infos window.

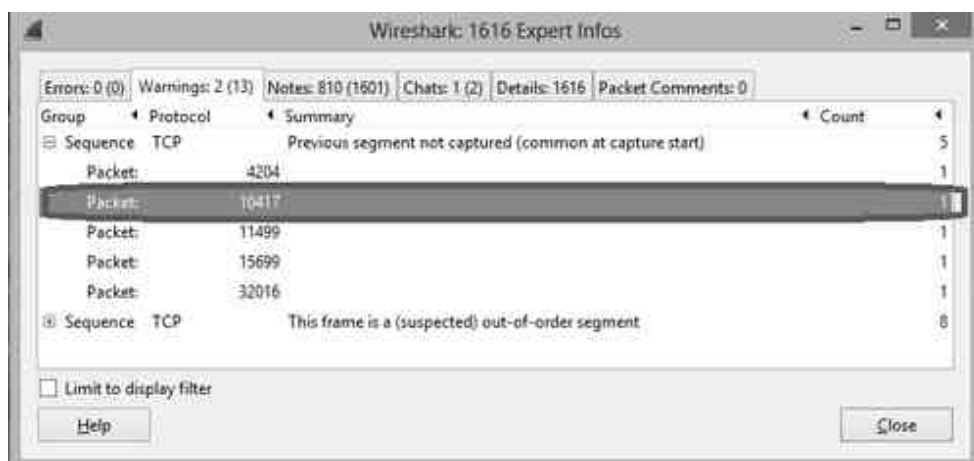
Topic 145: Wireshark Lab 49

In this topic, we find out where packets are being dropped using Wireshark.

Step 1: Open *tr-general101d.pcapng*.

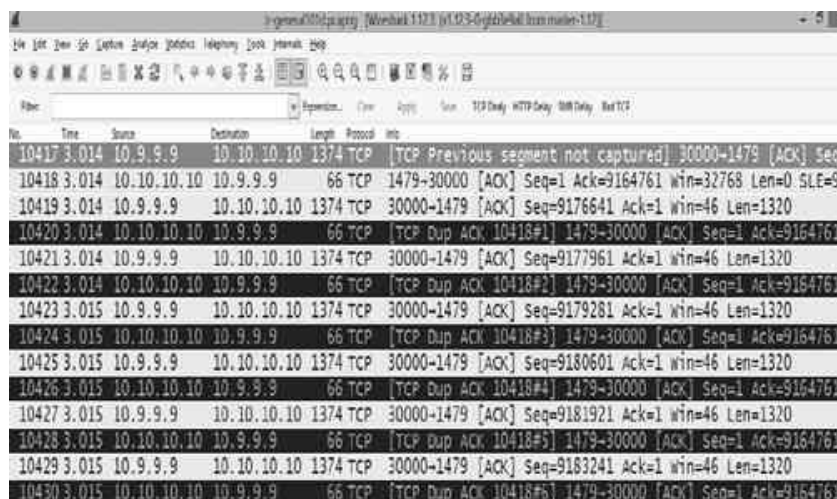
Step 2: To view the Expert Infos window, click **Expert Infos** button on the Status Bar.

Step 3: Click the **Warnings** tab and expand the *Previous segment not captured (common at capture start)* section.

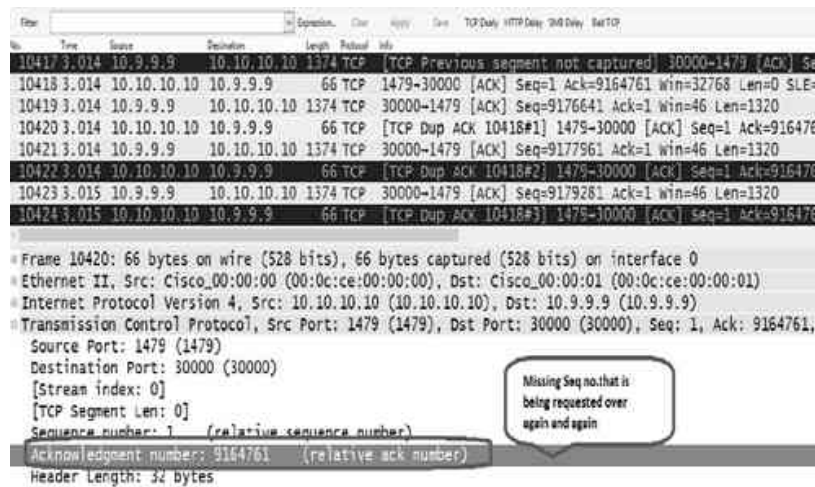


In the listing, let's examine **Packet 10,417**. Click on it and Wireshark will take you to that packet in Main window.

Step 4: After the point of packet loss, we can see numerous Duplicate ACKs after the missing packet indication.



10.10.10.10 (the receiver) supports Fast Recovery as it is using Duplicate ACKs. Each Duplicate ACK is a request for sequence number 9,164,761. See Acknowledgment Number field in the Duplicate ACKs. Close the Expert Infos window.



Step 5: Packets were captured upstream from the point of packet loss (closer to the data sender than the point of packet loss) or downstream from the point of packet loss (closer to the receiver than the point of packet loss)?

Step 6: We can use the sequence number of the packet that is missing to determine if we see the original and the Retransmission or just Retransmission. Enter **tcp.seq==9164761** in display filter area.



We can see that traffic has been captured downstream from point of packet loss. Packet loss always affects network performance. Locate it and then figure out why it is happening.

Topic 146: Duplicate ACKs and their Causes

This topic describes duplicate ACKs and their causes.

Duplicate ACKs are used to inform the sender about packet loss. They can also be an indication of out-of-order packets. A host generates Duplicate ACKs if it supports Fast Recovery and notices that a packet has arrived with a sequence number beyond the calculated next sequence number. Details of Fast Recovery can be found in RFC 5681, "TCP Congestion Control." To determine whether an ACK is a duplicate ACK,

Wireshark uses Data bytes, Window Size, Sequence Number, and ACK Number fields. When two or more packets with same Data bytes, Window Size, Sequence Number, and ACK Number fields arrive at a host from a source, we mark the second and onwards as duplicates of the first ACK. In Duplicate ACKs, the ACK Number field value indicates the requested sequence. A TCP host keeps on sending Duplicate ACKs until it receives missing packet. When a receiver receives a packet with the incoming sequence number value jumping higher than the expected, it does not know if the packet is lost or it is just out of order and will be arriving soon. Wireshark marks a packet as Out-of-Order if the missing sequence number packet arrives within 3 ms. In case, it arrives later than 3 ms, Wireshark marks it a Retransmission or Fast Retransmission. In Wireshark, we can use **tcp.analysis.duplicate_ack** to determine the number of duplicate ACKs, while **tcp.analysis.out_of_order** is used to keep a record of Out-of-Order packets.

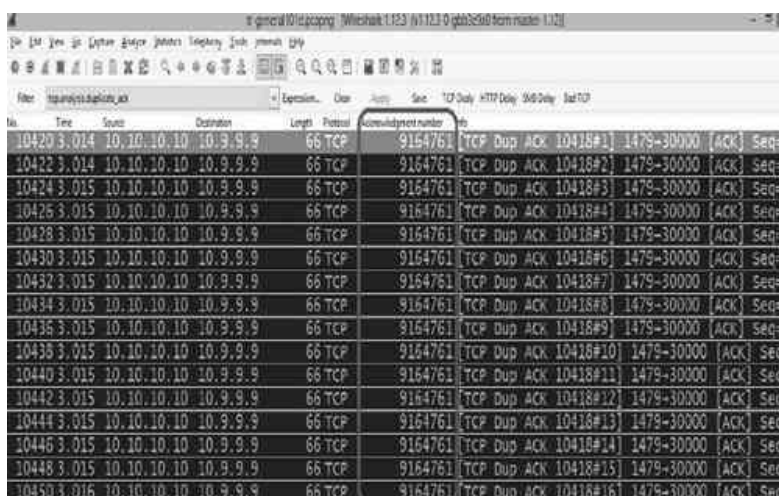
Topic 147: Wireshark Lab 50

In this topic, we use a Filter to count Duplicate ACKs in Wireshark.

Step 1: Open *tr-general101d.pcapng*.

Step 2: Enter the filter **tcp.analysis.duplicate_ack** in the display filter area and then Click **Apply**. From the Status Bar, we see that Wireshark has detected 1,019 Duplicate ACKs.

Step 3: Expand **TCP header** of a Packet. Click on the **Acknowledgment Number** field and select **Apply as Column**. Name it **ACK#** by right-clicking on this column and selecting **Edit Column Details**. On scrolling through the trace file, you will observe that these Duplicate ACKs are requests for a single missing packet whose sequence number is 9,164,761.



The screenshot shows a Wireshark packet capture of a TCP connection. The display filter is set to 'tcp.analysis.duplicate_ack'. The packet list shows a series of duplicate ACKs for sequence 9164761. The packet details pane shows the TCP header with the Acknowledgment Number field highlighted. The packet bytes pane shows the raw data of the packet.

No.	Time	Source	Destination	Length	Protocol	Acknowledgment Number	Info
10420	3.014	10.10.10.10	10.9.9.9	66	TCP	9164761	[TCP Dup ACK 10418#1] 1479-30000 [ACK] Seq=
10422	3.014	10.10.10.10	10.9.9.9	66	TCP	9164761	[TCP Dup ACK 10418#2] 1479-30000 [ACK] Seq=
10424	3.015	10.10.10.10	10.9.9.9	66	TCP	9164761	[TCP Dup ACK 10418#3] 1479-30000 [ACK] Seq=
10426	3.015	10.10.10.10	10.9.9.9	66	TCP	9164761	[TCP Dup ACK 10418#4] 1479-30000 [ACK] Seq=
10428	3.015	10.10.10.10	10.9.9.9	66	TCP	9164761	[TCP Dup ACK 10418#5] 1479-30000 [ACK] Seq=
10430	3.015	10.10.10.10	10.9.9.9	66	TCP	9164761	[TCP Dup ACK 10418#6] 1479-30000 [ACK] Seq=
10432	3.015	10.10.10.10	10.9.9.9	66	TCP	9164761	[TCP Dup ACK 10418#7] 1479-30000 [ACK] Seq=
10434	3.015	10.10.10.10	10.9.9.9	66	TCP	9164761	[TCP Dup ACK 10418#8] 1479-30000 [ACK] Seq=
10436	3.015	10.10.10.10	10.9.9.9	66	TCP	9164761	[TCP Dup ACK 10418#9] 1479-30000 [ACK] Seq=
10438	3.015	10.10.10.10	10.9.9.9	66	TCP	9164761	[TCP Dup ACK 10418#10] 1479-30000 [ACK] Seq=
10440	3.015	10.10.10.10	10.9.9.9	66	TCP	9164761	[TCP Dup ACK 10418#11] 1479-30000 [ACK] Seq=
10442	3.015	10.10.10.10	10.9.9.9	66	TCP	9164761	[TCP Dup ACK 10418#12] 1479-30000 [ACK] Seq=
10444	3.015	10.10.10.10	10.9.9.9	66	TCP	9164761	[TCP Dup ACK 10418#13] 1479-30000 [ACK] Seq=
10446	3.015	10.10.10.10	10.9.9.9	66	TCP	9164761	[TCP Dup ACK 10418#14] 1479-30000 [ACK] Seq=
10448	3.015	10.10.10.10	10.9.9.9	66	TCP	9164761	[TCP Dup ACK 10418#15] 1479-30000 [ACK] Seq=
10450	3.016	10.10.10.10	10.9.9.9	66	TCP	9164761	[TCP Dup ACK 10418#16] 1479-30000 [ACK] Seq=

You can observe that there are many Duplicate ACKs that are requests for a single missing packet whose sequence number is 9,164,761. The host continues sending Duplicate ACKs until missing sequence number is resolved.

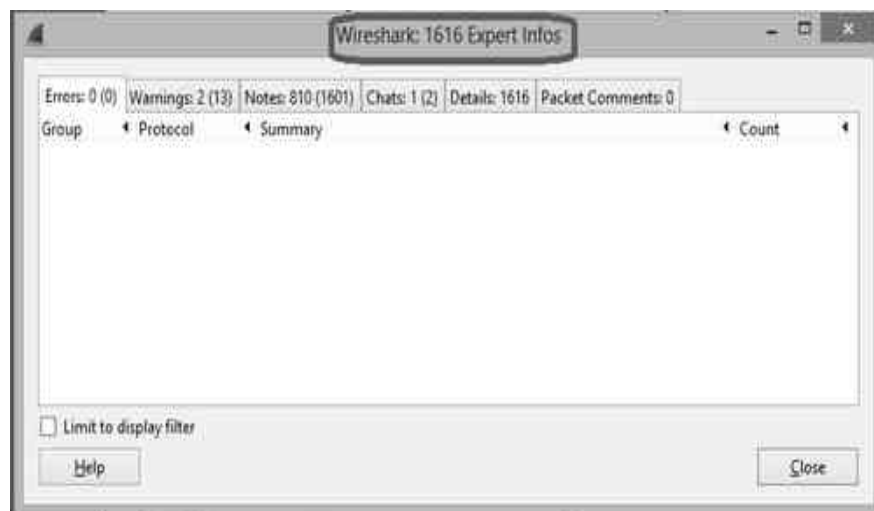
Step 4: To hide the **ACK#** column from view, right-click the **ACK#** column and select **Hide Column**. Click **Clear** to remove the display filter.

Topic 148: Wireshark Lab 51

This topic finds Duplicate ACKs using Expert Infos in Wireshark.

Step 1: Open *tr-general101d.pcapng*.

Step 2: Click the **Expert Infos** button on the Status Bar.



Step 3: Let's examine problems in this trace file by clicking the **Notes** tab. Count the Duplicate ACKs. Wireshark does not count all Duplicate ACKs together, it groups Duplicate ACKs based on their number. In a trace file, assume packet loss occurs twice. Both times the original ACK, Duplicate ACK#1, Duplicate ACK#2, and Duplicate ACK#3 were sent. Wireshark will list 3 Duplicate ACKs with an indication that each occurred twice.

Wireshark: 1616 Expert Infos

Errors: 0 (0) | Warnings: 2 (13) | Notes: 810 (1601) | Chats: 1 (2) | Details: 1616 | Packet Comments: 0

Group	Protocol	Summary	Count
Sequence	TCP	Duplicate ACK (#1)	5
Sequence	TCP	Duplicate ACK (#2)	4
Sequence	TCP	Duplicate ACK (#3)	4
Sequence	TCP	Duplicate ACK (#4)	4
Sequence	TCP	Duplicate ACK (#5)	4
Sequence	TCP	Duplicate ACK (#6)	4
Sequence	TCP	Duplicate ACK (#7)	4
Sequence	TCP	Duplicate ACK (#8)	4
Sequence	TCP	Duplicate ACK (#9)	4
Sequence	TCP	Duplicate ACK (#10)	4
Sequence	TCP	Duplicate ACK (#11)	4
Sequence	TCP	Duplicate ACK (#12)	4
Sequence	TCP	Duplicate ACK (#13)	4

☐ Limit to display filter

Help Close

Duplicate ACK (#1) occurs 5 times i.e., the Fast Recovery process was launched 5 times. The 4 Duplicate ACK (#2) indications in one case receiver only had to send the first Duplicate ACK and then it recovered.

Wireshark: 1616 Expert Infos

Errors: 0 (0) | Warnings: 2 (13) | Notes: 810 (1601) | Chats: 1 (2) | Details: 1616 | Packet Comments: 0

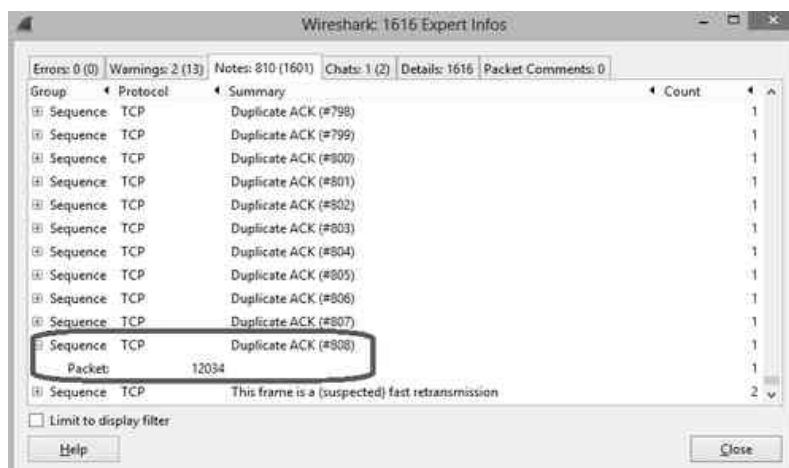
Group	Protocol	Summary	Count
Sequence	TCP	Duplicate ACK (#1)	5
Packet:		10429	1
Packet:		12039	1
Packet:		13700	1
Packet:		15845	1
Packet:		32017	1
Sequence	TCP	Duplicate ACK (#2)	4
Packet:		10422	1
Packet:		12041	1
Packet:		15702	1
Packet:		15847	1
Sequence	TCP	Duplicate ACK (#3)	4
Sequence	TCP	Duplicate ACK (#4)	4

☐ Limit to display filter

Help Close

Probably, the one out-of-order packet situation in the trace file occurs near Packet 3,217.

Step 4: Scroll to the end of the **Notes** section.



The TCP receiver kept on requesting for a missing packet 809 times. This means a packet got lost and the recovery took a significant amount of time. This can be due to path latency or a network disconnect. Finally, close the Expert Infos window.

Topic 149: Wireshark Lab 52

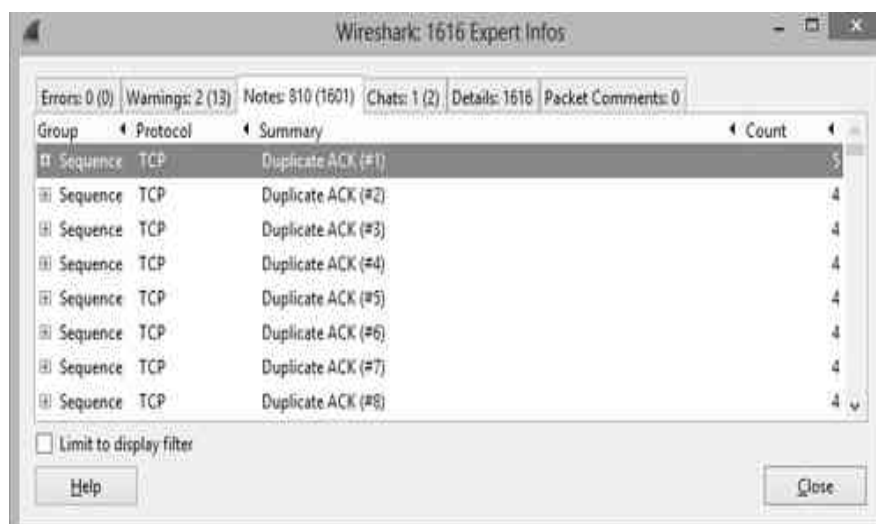
In this topic, we determine if Selective ACK (SACK) is in use by examining Duplicate ACKs in Wireshark.

When two TCP hosts are communicating and are using SACK, only the missing packets are retransmitted. If SACK is not in use many unnecessary retransmissions will be generated. The sender retransmits every data packet starting from the missing sequence number. If the TCP handshake packets are not captured, we can examine duplicate ACKs to determine if SACK is enabled on the connection.

Step 1: Open *tr-general101d.pcapng*.

Step 2: Click the **Expert Infos** button on the Status Bar.

Step 3: Duplicate ACKs can be located by clicking on the **Notes** tab.



Step 4: Expand the **Duplicate ACK (#1)** line and click on the first packet listed, whose number is 10,420. Go to Wireshark to examine that packet.



Step 5: SACK Left Edge (SLE) and SACK Right Edge (SRE) information is listed in the **Info** column. You can also expand packet 10420 and look for the SLE and RLE values in the Options area of the TCP header.

```

* Frame 10420: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface
* Ethernet II, Src: Cisco_00:00:00 (00:0c:ce:00:00:00), Dst: Cisco_00:00:01 (00:0c:c
* Internet Protocol Version 4, Src: 10.10.10.10 (10.10.10.10), Dst: 10.9.9.9 (10.9.9
* Transmission Control Protocol, Src Port: 1479 (1479), Dst Port: 30000 (30000), Seq
  Source Port: 1479 (1479)
  Destination Port: 30000 (30000)
  [Stream index: 0]
  [TCP Segment Len: 0]
  Sequence number: 1 (relative sequence number)
  Acknowledgment number: 9164761 (relative ack number)
  Header Length: 32 bytes
  [Calculated window size: 32768]
  [Window size scaling factor: -1 (unknown)]
  * Checksum: 0x9b37 [validation disabled]
  Urgent pointer: 0
  * Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), SACK
    * No-Operation (NOP)
    * No-Operation (NOP)
    * SACK: 9175321-9177961
      Kind: SACK (5)
      Length: 10
      left edge = 9175321 (relative)
      right edge = 9177961 (relative)
      [TCP SACK Count: 1]
  * [SEQ/ACK analysis]

```

Topic 150: Out-of-Order Packets and their Causes

This topic describes out-of-order Packets and their causes.

On a receiving host, the transmission control protocol (TCP) layer cannot pass the received data up to the application layer until all the bytes are in the correct order.

Let's assume a scenario where two packets arrive at a host in reverse order. The packets both arrive within 1 ms. This situation is unlikely to cause a problem. Therefore, we may expect that performance will not be affected if there is a little time between the expected arrival and the actual arrival of out-of-order packets. Wireshark labels a packet out-of-order if it

- (a) Contains data,
- (b) Does not advance the sequence number value, and
- (c) Arrives within 3 ms of the highest sequence number seen.

Causes of Out-of-Order Packets

- a stream using multiple different speed paths to reach the target,
- a poorly configured queuing along a path or,
- Asymmetric routing.

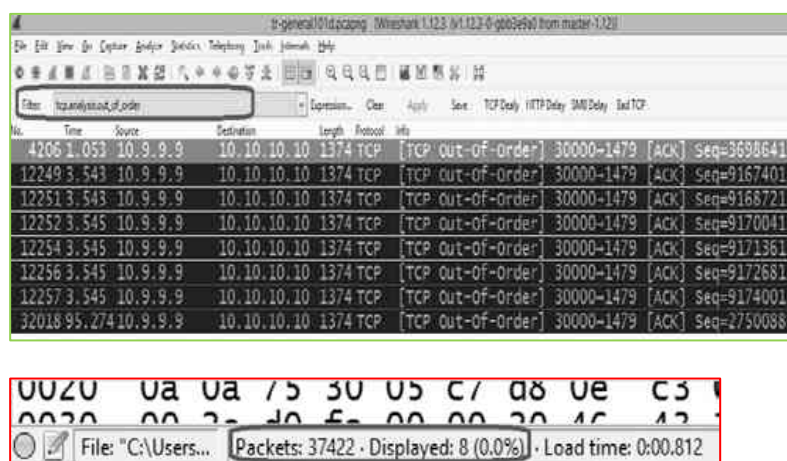
A queuing device that does not forward packets in a first-in/first-out (FIFO) order can cause packets to arrive a host in an out-of-order fashion.

Topic 151: Wireshark Lab 53

This topic uses a filter to count out-of-order packets in Wireshark.

Step 1: Open *tr-general101d.pcapng*.

Step 2: Enter the expression **tcp.analysis.out_of_order** in the display filter area. Click **Apply**.



We find that Wireshark has detected 8 Out-of-Order packets as indicated by the Status Bar. On examining the packet numbers, we can have an idea that packets

4206 and 32018 are out-of-order packets, while remaining 6 packets belong to a group of out-of-order packets in close proximity.

```

Internet Protocol Version 4, Src: 10.9.9.9 (10.9.9.9), Dst: 10.10.10.10 (10.10.10.10)
Transmission Control Protocol, Src Port: 30000 (30000), Dst Port: 1479 (1479), Seq: 3698641, A
  Source Port: 30000 (30000)
  Destination Port: 1479 (1479)
  [Stream index: 0]
  [TCP Segment Len: 1320]
  Sequence number: 3698641 (relative sequence number)
  [Next sequence number: 3699961 (relative sequence number)]
  Acknowledgment number: 1 (relative ack number)
  Header Length: 20 bytes
  ... 0000 0001 0000 = Flags: 0x010 (ACK)
  window size value: 46
  [Calculated window size: 46]
  [Window size scaling factor: -1 (unknown)]
  Checksum: 0xd0fa [validation disabled]
  Urgent pointer: 0
  [SEQ/ACK analysis]
  [Bytes in flight: 2640]
  [TCP Analysis Flags]
  [Expert Info (Warn/Sequence): this frame is a (suspected) out-of-order segment]
  [Timestamp:

```

If you find multiple Out-of-Order packets in close proximity, then it is likely that a set of packets have been lost. These packets are Retransmissions that arrived within 3 ms.

Step 3: When you are done, click the **Clear** button to remove the filter.

Topic 152: Wireshark Lab 54

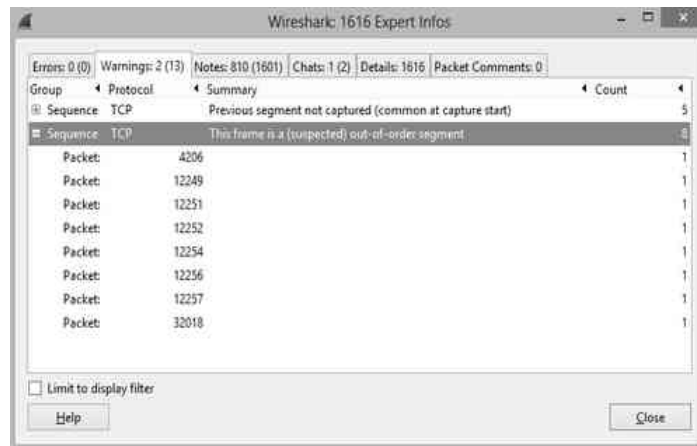
In this topic, we find out-of-order packets with Expert Infos in Wireshark.

Step 1: Open *tr-general101d.pcapng*.

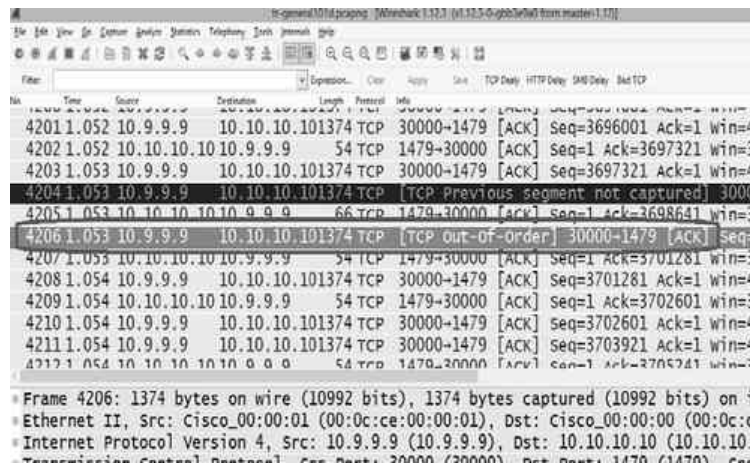
Step 2: On the Status Bar, click the **Expert Infos** button.



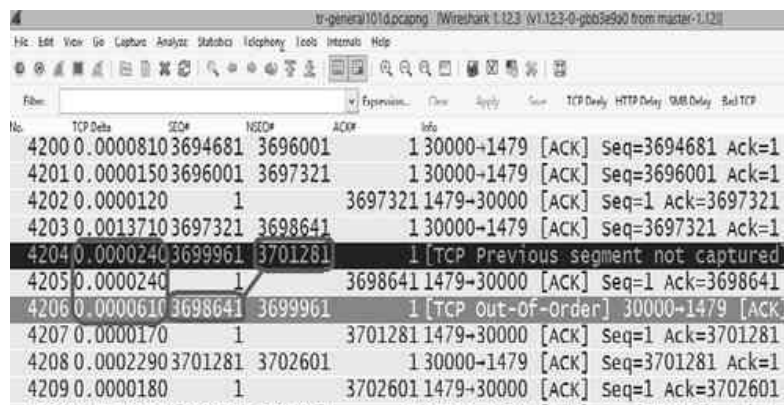
Step 3: Click the **Warnings** tab. Expand the **Out-of-Order** segment section.



Click on the first entry i.e. Packet 4,206 and then click the **Close** button and then go to the main Wireshark window.



Step 4: If you have previously created, **SEQ#**, **NEXTSEQ#** and **ACK#** columns, then right-click on any column heading and display them. Otherwise, create them. Select **Sequence number** field in the TCP header of a packet, and then **Apply as a Column**. Rename the column as **SEQ#** by right-clicking on the column and selecting **Edit Column Details**.



We can see that **SEQ#** of Packet 4,206 is lower than the **NSEQ#** in Packet 4,204. Also, the out-of-order packet arrived 85 microseconds, which is less than 3 ms after the previous packet from 10.9.9.9.

Topic 153: Causes of Fast Retransmissions

In this topic, we describe fast retransmissions and their causes. Fast retransmissions are triggered, when three identical ACKs (the original ACK and two Duplicate ACKs) arrive at a receiving host. Fast Retransmissions are a sign that packet loss has occurred and are part of the Fast Recovery process.

Characteristics of Fast Retransmissions

The segment contains data or has the SYN or FIN bits set to 1. The segment does not advance the sequence number. At least 2 Duplicate ACKs were coming from the reverse direction. The Sequence Number field value matches Acknowledgment Number field value in the preceding Duplicate ACKs. The packet arrived within 20 ms of the last Duplicate ACK.

Causes of Fast Retransmissions

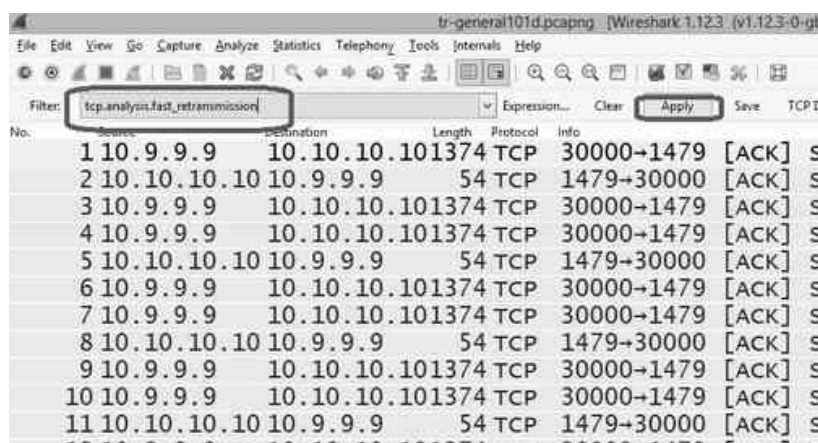
When a host believes there is a packet loss and supports Fast Recovery, then it generates at least four identical ACKs. In addition to the original ACK, at least three Duplicate ACKs are sent towards the sender requesting the missing segment. Packets are typically lost at infrastructure devices.

Topic 154: Wireshark Lab 55

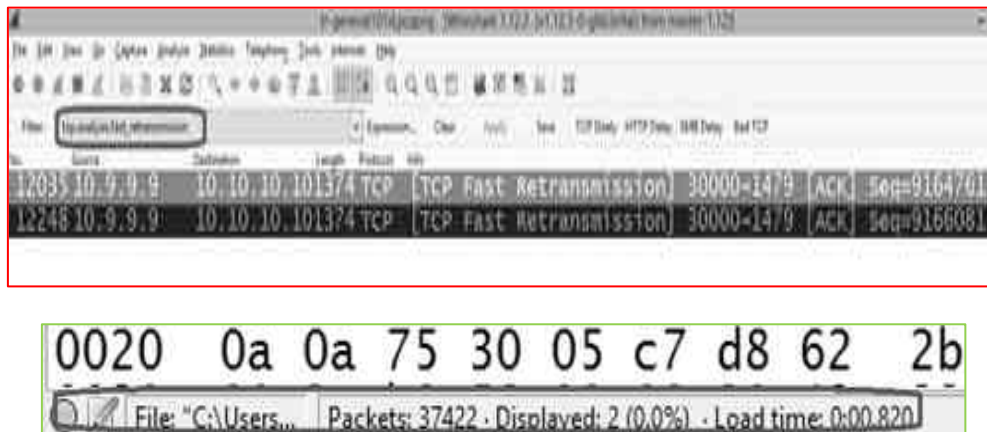
This topic uses a filter to count fast retransmission packets in Wireshark.

Step 1: Open *tr-general101d.pcapng*.

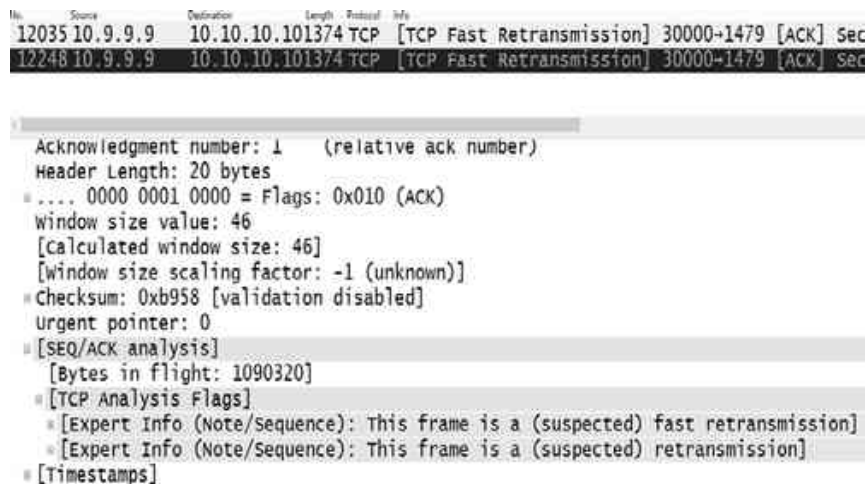
Step 2: Enter the filter **tcp.analysis.fast_retransmission** in the display filter area. Click **Apply**.



In this trace file, Wireshark has detected two Fast Retransmissions as indicated by the Status Bar.



Step 3: Expand the **[SEQ/ACK analysis]** section on one of the Fast Retransmissions. Wireshark colors this area cyan. Color of the Expert Infos Notes is Cyan.



There are two Expert Infos indications marked by Wireshark on this one packet. Frame is marked to be a Fast Retransmission and a Retransmission. Fast Retransmission is a Retransmission.

Step 4: Click the **Clear** button to remove the filter as you finish your analysis.

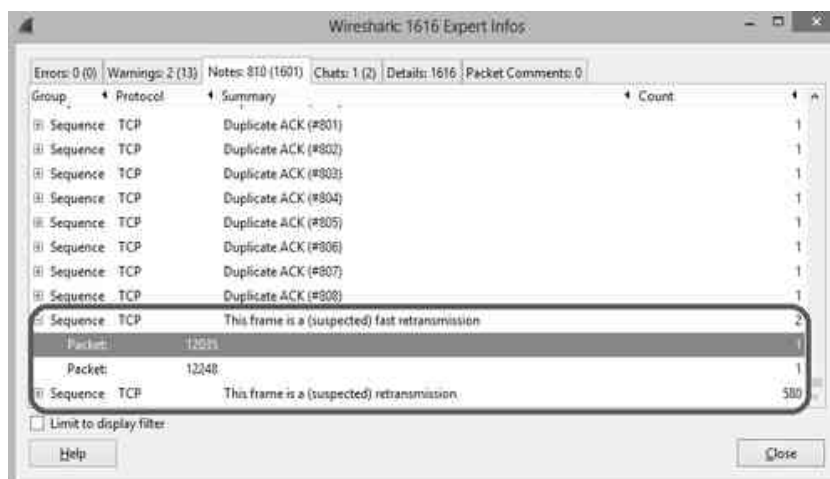
Topic 155: Wireshark Lab 56

In this topic, we find fast retransmission packets with Expert Infos in Wireshark.

Step 1: Open *tr-general101d.pcapng*.

Step 2: Click the **Expert Infos** button on the Status Bar.

Step 3: Click the **Notes** tab. Scroll through the list. As you find the Fast Retransmissions section, expand it and click on the first entry, **Packet 12,035**.



We can observe that there are 808 Duplicate ACKs before the Fast Retransmission packet. Return to the main Wireshark window after clicking the **Close** button.

Step 4: If you have created **SEQ#**, **NEXTSEQ#** and **ACK#** columns, then you can display them by right-clicking on a column and selecting them from Displayed Columns.

The image shows the main Wireshark packet list window. It displays a list of network packets. The columns shown are No., Time, TCP Delta, Seq, Next Seq, Ack, Source, Destination, Length, Protocol, and Info. The packets are numbered 12030 through 12039. The 'TCP Delta' column shows values like 0.0000100, 0.0000290, 0.0000090, 0.0026980, 0.0000150, 0.0000330, 0.0002040, 0.0000110, 0.0000190, and 0.0000090. The 'Seq' column shows values like 1, 10252441, 10253781, 10253761, 10255081, 9164761, 10255081, 10256401, 10257721, and 9166081. The 'Next Seq' column shows values like 1, 10253781, 10255081, 10255081, 10255081, 9166081, 10256401, 10257721, 10257721, and 9166081. The 'Ack' column shows values like 1, 1, 1, 1, 1, 1, 1, 1, 1, and 1. The 'Source' column shows IP addresses like 10.10.10.10, 10.9.9.9, 10.10.10.10, 10.9.9.9, 10.10.10.10, 10.9.9.9, 10.10.10.10, 10.9.9.9, 10.10.10.10, and 10.9.9.9. The 'Destination' column shows IP addresses like 10.10.10.10, 10.9.9.9, 10.10.10.10, 10.9.9.9, 10.10.10.10, 10.9.9.9, 10.10.10.10, 10.9.9.9, 10.10.10.10, and 10.9.9.9. The 'Length' column shows values like 74, 74, 74, 74, 74, 74, 74, 74, 74, and 74. The 'Protocol' column shows 'TCP'. The 'Info' column shows details like '[TCP Dup ACK 10418#8]', '[ACK] Seq=30000-1479', '[TCP Fast Retransmission]', and '[TCP Dup ACK 12037#1]'. A red arrow points to the packet with sequence number 9164761.

From the TCP Delta column, we can see that the Fast Retransmission occurred within 20 ms of the last Duplicate ACK. If you consider time, you will find that only about 465 ms have passed between when packet with sequence number 9,164,761 went missing and when Fast Retransmission was initiated. Although there were 808 Duplicate ACKs, but this ½-second delay would be unnoticeable by the user.

Topic 156: Causes of Retransmissions

This topic describes retransmissions and their causes.

Wireshark considers a packet to be a Retransmission if the segment contains data or has the SYN or FIN bits set to 1. Sequence number is not advanced by the segment.

Duplicate ACKs have not triggered Retransmission. The segment arrives > 3 ms later than the previous packet with a higher sequence number. A Retransmission Time Out (RTO) at a sender triggers Standard Retransmissions. The RTO timer is used to ensure data delivery continues even if the TCP peer stops communicating (with ACKs). Each TCP host calculates and maintains a RTO timer. This timer value is based on the round trip time learned through previous data transmissions and related acknowledgments. The RTO value consistently changes through the conversation.

What Causes Retransmissions?

A TCP host begins counting down the RTO, when it sends a data packet. The sender retransmits the unacknowledged data packet, if the RTO timer expires without receiving an ACK for the data packet. The sender has no idea whether the original packet was lost or the acknowledgment was lost. The sender just knows that there some problem as it did not receive an ACK within the RTO.

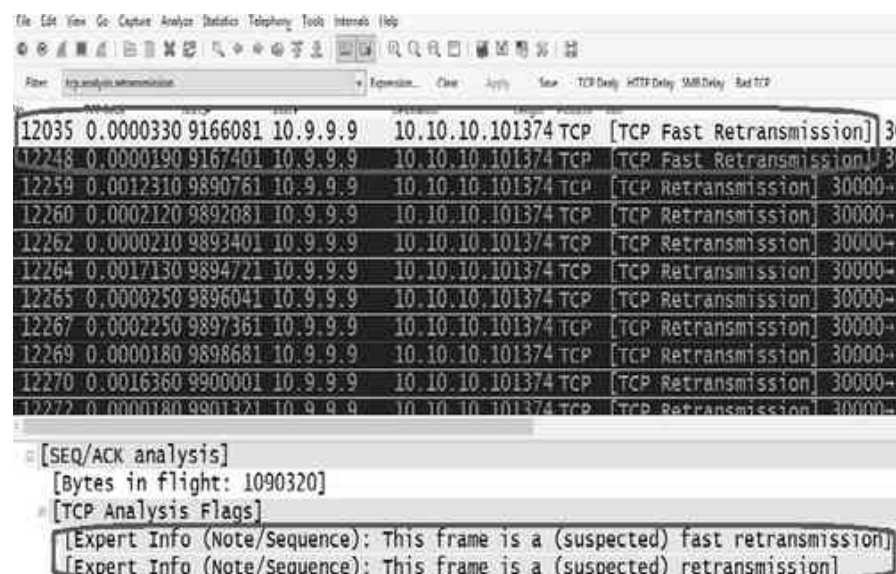
Topic 157: Wireshark Lab 57

This topic uses a filter to count retransmission packets in Wireshark.

Applying **tcp.analysis.retransmission** in Wireshark as a filter for Retransmissions will also display Fast Retransmissions. You need to exclude Fast Retransmissions in the filter, if you are interested in seeing packets that are only labeled as standard Retransmissions.

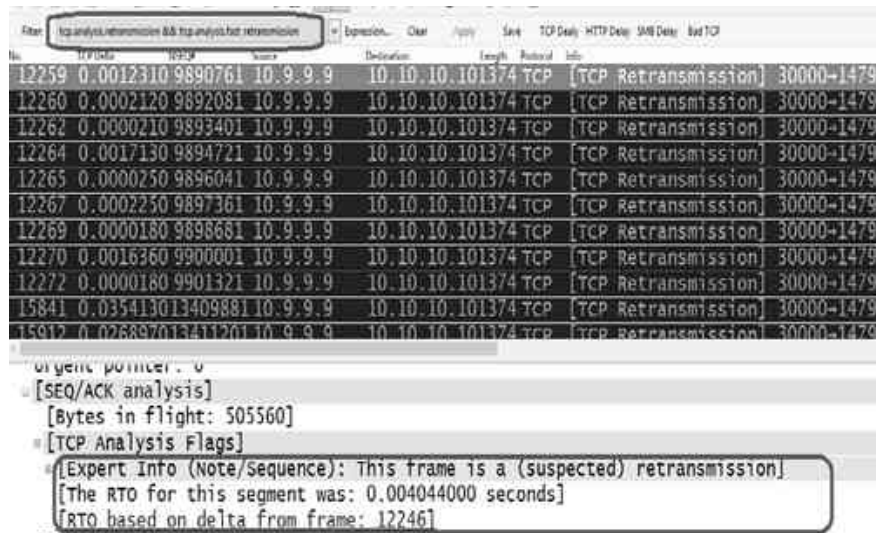
Step 1: Open *tr-general101d.pcapng*.

Step 2: Enter the filter **tcp.analysis.retransmission** in the display filter area. Click **Apply**.



From the Status Bar, we can see that Wireshark has detected 580 Retransmissions in the trace file. This also includes two Fast Retransmissions.

Step 3: Update the filter to the following: **tcp.analysis.retransmission && !tcp.analysis.fast_retransmission**. Click **Apply**.



There are 578 packets that match this filter. So the two Fast Retransmissions have been removed from view.

Step 4: Click the **Clear** button to remove the filter.

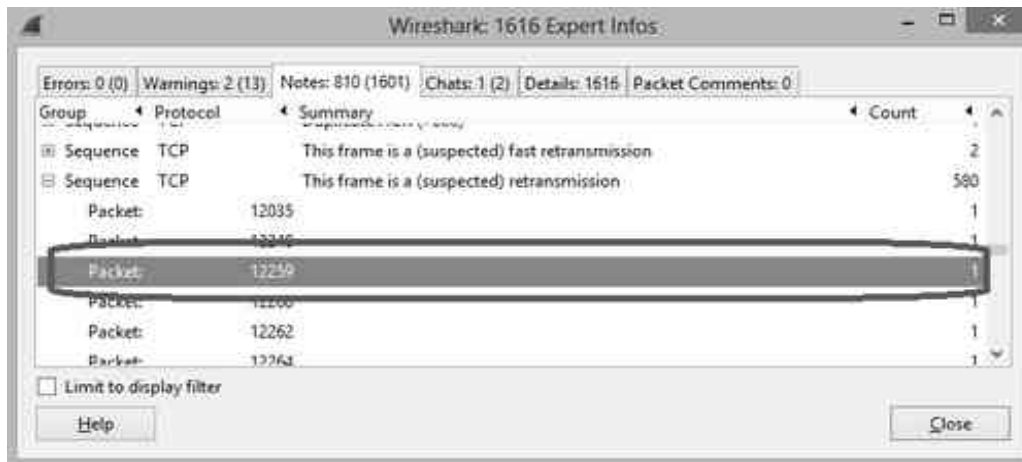
Topic 158: Wireshark Lab 58

This topic finds retransmission packets with Expert infos in Wireshark.

Step 1: Open **tr-general101d.pcapng**.

Step 2: On the Status Bar, Click the **Expert Infos** button.

Step 3: Click the **Notes** tab. Scroll through list. Expand the **Retransmissions** section and click on **Packet 12,259**. This will take you to the Wireshark Main window. Click the **Close**.



No.	Source	Destination	Length	Protocol	Info
12244	10.9.9.9	10.10.10.101374	TCP	50000-1479	[ACK] Seq=10392361 Ack=1 W
12245	10.10.10.10	10.9.9.9	74	TCP	[TCP Dup ACK 12037#10] 1479-30000 [A
12246	10.9.9.9	10.10.10.101374	TCP	30000-1479	[PSH, ACK] Seq=10393681 Ac
12247	10.10.10.10	10.9.9.9	74	TCP	[TCP Dup ACK 12037#10] 1479-30000 [A
12248	10.9.9.9	10.10.10.101374	TCP		[TCP Fast Retransmission] 30000-1479
12249	10.9.9.9	10.10.10.101374	TCP		[TCP Out-Of-Order] 30000-1479 [ACK] S
12250	10.10.10.10	10.9.9.9	74	TCP	1479-30000 [ACK] Seq= Ack=9168721 wi
12251	10.9.9.9	10.10.10.101374	TCP		[TCP Out-Of-Order] 30000-1479 [ACK] S
12252	10.9.9.9	10.10.10.101374	TCP		[TCP Out-Of-Order] 30000-1479 [ACK] S
12253	10.10.10.10	10.9.9.9	74	TCP	1479-30000 [ACK] Seq= Ack=9171361 wi
12254	10.9.9.9	10.10.10.101374	TCP		[TCP Out-Of-Order] 30000-1479 [ACK] S
12255	10.10.10.10	10.9.9.9	74	TCP	1479-30000 [ACK] Seq= Ack=9172681 wi
12256	10.9.9.9	10.10.10.101374	TCP		[TCP Out-Of-Order] 30000-1479 [ACK] S
12257	10.9.9.9	10.10.10.101374	TCP		[TCP Out-Of-Order] 30000-1479 [ACK] S
12258	10.10.10.10	10.9.9.9	66	TCP	1479-30000 [ACK] Seq= Ack=9889441 wi
12259	10.9.9.9	10.10.10.101374	TCP		[TCP Retransmission] 30000-1479 [ACK]

Step 4: If you have previously created, **SEQ#**, **NEXTSEQ#** and **ACK#** columns, then right-click on any column heading and display them. Otherwise, create them. Select **Sequence number** field in the TCP header of a packet, and then **Apply as a Column**. Rename the column as **SEQ#** by right-clicking on the column and selecting **Edit Column Details**. Scroll up and you will find Packet 12,246 is a data packet from 10.9.9.9, which has the highest Sequence No. field value at this point of the trace file. Right-click on **Packet 12,246** and select **Set Time Reference (toggle)**.

The image shows a Wireshark packet capture window with several annotations highlighting specific TCP events:

- PKT12249: Out-of-Order within 3 ms of highest SEQ#** (points to packet 12249)
- PKT12246: Highest SEQ# from 10.9.9.9** (points to packet 12246)
- PKT12248: Fast Retransmission within 20 ms of Duplicate ACK** (points to packet 12248)
- PKT12259: Retransmission did not advance SEQ#, no related Duplicate ACKs; not within 3ms** (points to packet 12259)

No.	Time	Source	Destination	Protocol	Seq#	NextSeq#	Info
12243	3.541	10.9.9.9	10.10.10.10	TCP	10391041	10392361	ndmps > dberegiste
12244	3.541	10.9.9.9	10.10.10.10	TCP	10392361	10393681	ndmps > dberegiste
12245	3.541	10.10.10.10	10.9.9.9	TCP	9166001	9166101	TCP Dup ACK 12031
12246	3.541	10.10.10.10	10.9.9.9	TCP	9166001	9166101	TCP Dup ACK 12031
12247	0.000	10.10.10.10	10.9.9.9	TCP	9166001	9166101	TCP Dup ACK 12031
12248	0.000	10.10.10.10	10.9.9.9	TCP	9166001	9166101	TCP Fast Retransmission
12249	0.000	10.10.10.10	10.9.9.9	TCP	9166001	9166101	TCP Out-of-order
12250	0.000	10.9.9.9	10.10.10.10	TCP	1	9166721	dberegiste > ndmp
12251	0.000	10.10.10.10	10.9.9.9	TCP	9166721	9170041	TCP Out-of-order
12252	0.002	10.10.10.10	10.9.9.9	TCP	9170041	9171361	TCP Out-of-order
12253	0.002	10.9.9.9	10.10.10.10	TCP	1	9171361	dberegiste > ndmp
12254	0.002	10.10.10.10	10.9.9.9	TCP	9171361	9172681	TCP Out-of-order
12255	0.002	10.10.10.10	10.9.9.9	TCP	1	9172681	dberegiste > ndmp
12256	0.002	10.10.10.10	10.9.9.9	TCP	9172681	9174001	TCP Out-of-order
12257	0.002	10.10.10.10	10.9.9.9	TCP	9174001	9175321	TCP Out-of-order
12258	0.002	10.10.10.10	10.9.9.9	TCP	1	9889441	dberegiste > ndmp
12259	0.004	10.10.10.10	10.9.9.9	TCP	9889441	9890761	TCP Retransmission
12260	0.004	10.10.10.10	10.9.9.9	TCP	9890761	9892081	TCP Retransmission
12261	0.004	10.10.10.10	10.9.9.9	TCP	1	9892081	dberegiste > ndmp
12262	0.004	10.10.10.10	10.9.9.9	TCP	9892081	9893401	TCP Retransmission
12263	0.004	10.10.10.10	10.9.9.9	TCP	1	9893401	dberegiste > ndmp

Step 5: Time Reference, can be removed by right-clicking on **Packet 12,246**, and selecting **Set Time Reference (toggle)**.

Topic 159: Causes of ACKed Unseen Segments

In this topic, we explain ACKed Unseen segments and their causes.

What is ACKed Unseen Segment?

When Wireshark sees an ACK, but it did not see the data packet that is being acknowledged, it calls it an ACK Unseen Segment. For each host, Wireshark keeps a track of sequence number, next sequence number, and acknowledgment number values. Wireshark also keeps a track of a "Maximum Sequence Number to be ACKed" (*maxseqtobeacked*) value. This value gets updated as data is received. If Wireshark sees an ACK to acknowledge a data packet that has a higher Sequence Number field value than *maxseqtobeacked*, then it shows that it has missed the data packet that is being ACKed. Let's assume that Wireshark is running on a client which is communicating with a server. An ACK being sent to acknowledge all sequence numbers up to 9,380 has been captured.

Causes of ACKed Unseen Segment:

A) Problems occurring during the capture process can cause ACKed Unseen Segments. If the switch is oversubscribed when you are using switch port spanning, then it is dropping packets that should have been forwarded to Wireshark.

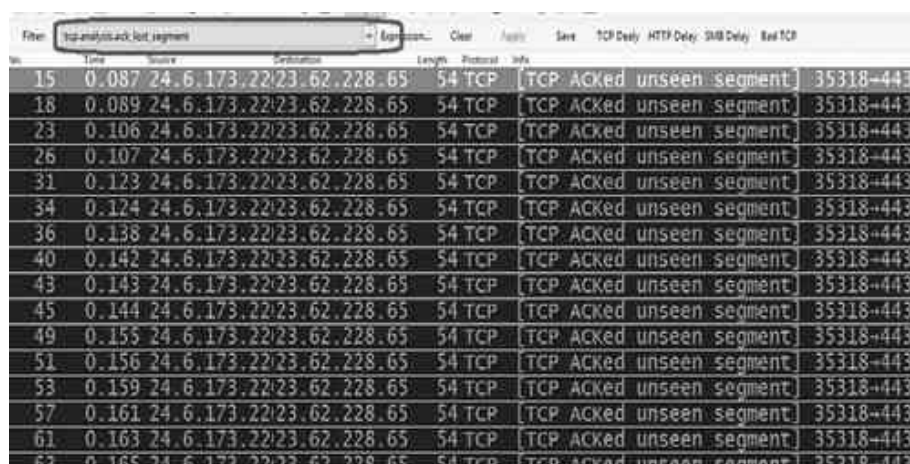
B) Asymmetric routing can cause ACKed Unseen Segments. With asymmetric routing, data may be flowing along one path on the network while ACKs flow along another path. Wireshark was able to capture ACKs but not the data.

Topic 160: Wireshark Lab 59

In this topic, we use a filter to count ACKed Unseen Segment warnings in Wireshark.

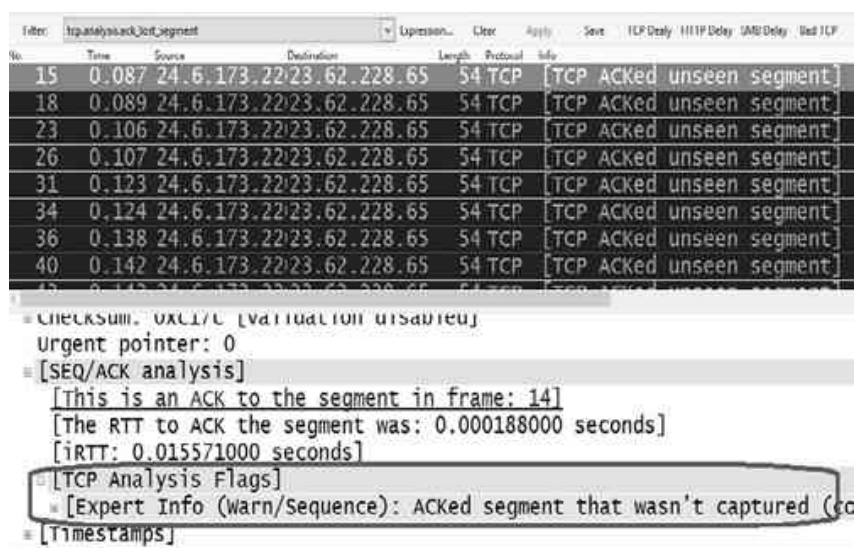
Step 1: Open *tr-badcapture.pcapng*.

Step 2: Enter the filter expression: `tcp.analysis.ack_lost_segment` in the display filter area. Click **Apply**.



No.	Time	Source	Destination	Length	Protocol	Info
15	0.087	24.6.173.22	23.62.228.65	54	TCP	[TCP ACKed unseen segment] 35318→443
18	0.089	24.6.173.22	23.62.228.65	54	TCP	[TCP ACKed unseen segment] 35318→443
23	0.106	24.6.173.22	23.62.228.65	54	TCP	[TCP ACKed unseen segment] 35318→443
26	0.107	24.6.173.22	23.62.228.65	54	TCP	[TCP ACKed unseen segment] 35318→443
31	0.123	24.6.173.22	23.62.228.65	54	TCP	[TCP ACKed unseen segment] 35318→443
34	0.124	24.6.173.22	23.62.228.65	54	TCP	[TCP ACKed unseen segment] 35318→443
36	0.138	24.6.173.22	23.62.228.65	54	TCP	[TCP ACKed unseen segment] 35318→443
40	0.142	24.6.173.22	23.62.228.65	54	TCP	[TCP ACKed unseen segment] 35318→443
43	0.143	24.6.173.22	23.62.228.65	54	TCP	[TCP ACKed unseen segment] 35318→443
45	0.144	24.6.173.22	23.62.228.65	54	TCP	[TCP ACKed unseen segment] 35318→443
49	0.155	24.6.173.22	23.62.228.65	54	TCP	[TCP ACKed unseen segment] 35318→443
51	0.156	24.6.173.22	23.62.228.65	54	TCP	[TCP ACKed unseen segment] 35318→443
53	0.159	24.6.173.22	23.62.228.65	54	TCP	[TCP ACKed unseen segment] 35318→443
57	0.161	24.6.173.22	23.62.228.65	54	TCP	[TCP ACKed unseen segment] 35318→443
61	0.163	24.6.173.22	23.62.228.65	54	TCP	[TCP ACKed unseen segment] 35318→443
63	0.165	24.6.173.22	23.62.228.65	54	TCP	[TCP ACKed unseen segment] 35318→443

Look at the Status Bar. You will see that Wireshark has detected 24 ACKed Unseen Segments.



No.	Time	Source	Destination	Length	Protocol	Info
15	0.087	24.6.173.22	23.62.228.65	54	TCP	[TCP ACKed unseen segment] 35318→443
18	0.089	24.6.173.22	23.62.228.65	54	TCP	[TCP ACKed unseen segment] 35318→443
23	0.106	24.6.173.22	23.62.228.65	54	TCP	[TCP ACKed unseen segment] 35318→443
26	0.107	24.6.173.22	23.62.228.65	54	TCP	[TCP ACKed unseen segment] 35318→443
31	0.123	24.6.173.22	23.62.228.65	54	TCP	[TCP ACKed unseen segment] 35318→443
34	0.124	24.6.173.22	23.62.228.65	54	TCP	[TCP ACKed unseen segment] 35318→443
36	0.138	24.6.173.22	23.62.228.65	54	TCP	[TCP ACKed unseen segment] 35318→443
40	0.142	24.6.173.22	23.62.228.65	54	TCP	[TCP ACKed unseen segment] 35318→443
43	0.143	24.6.173.22	23.62.228.65	54	TCP	[TCP ACKed unseen segment] 35318→443

= Checksum: 0x1111 (validation disabled)
 Urgent pointer: 0
 = [SEQ/ACK analysis]
 [This is an ACK to the segment in frame: 14]
 [The RTT to ACK the segment was: 0.000188000 seconds]
 [RTT: 0.015571000 seconds]
 = [TCP Analysis Flags]
 [Expert Info (Warn/Sequence): ACKed segment that wasn't captured (cc
 = [Timestamps]

Step 3: Click the **Clear** button so that the filter expression is removed.

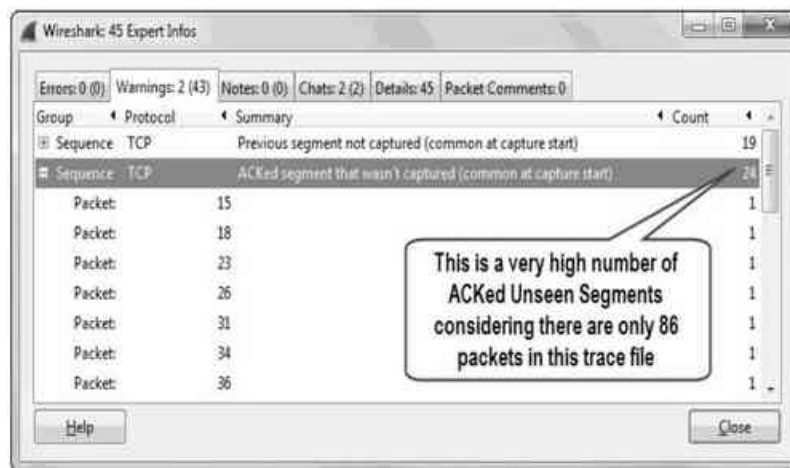
Topic 161: Wireshark Lab 60

In this topic, we find ACKed Unseen Segment indications using Expert Infos in Wireshark.

Step 1: Open *tr-badcapture.pcapng*.

Step 2: Click the **Expert Infos** button on the Status Bar.

Step 3: Click the **Warnings** tab. Expand the section with title **ACKed segment that wasn't captured (common at capture start)**.



Step 4: Click on the first entry, **Packet 15** and then click the **Close** button. Let's go to the main Wireshark window. Display the **SEQ#**, **NEXTSEQ#** and **ACK#** columns.

No.	Time	Seq#	NextSeq#	ACK#	Source	Destination	Length	Protocol	Info
11	0.071	2310	5970	809	23.62.228.6	24.6.173.220	1514	TLSv	Ignored Unknown Record
12	0.071	3976	5436	809	23.62.228.6	24.6.173.220	1514	TLSv	Ignored Unknown Record
13	0.071	809		5436	24.6.173.22	23.62.228.65	54	TCP	35318-443 [ACK] Seq=809
14	0.087	6896	8356	809	23.62.228.6	24.6.173.220	1514	TLSv	[TCP Previous segment not captured]
15	0.087	809		8356	24.6.173.22	23.62.228.65	54	TCP	[TCP ACKed unseen segment]
16	0.088	8356	9816	809	23.62.228.6	24.6.173.220	1514	TLSv	Ignored Unknown Record
17	0.089	11276	12736	809	23.62.228.6	24.6.173.220	1514	TLSv	[TCP Previous segment not captured]
18	0.089	809		12736	24.6.173.22	23.62.228.65	54	TCP	[TCP ACKed unseen segment]
19	0.104	12736	14196	809	23.62.228.6	24.6.173.220	1514	TLSv	Ignored Unknown Record
20	0.105	14196	15656	809	23.62.228.6	24.6.173.220	1514	TLSv	Ignored Unknown Record
21	0.105	15656	17116	809	23.62.228.6	24.6.173.220	1514	TLSv	Ignored Unknown Record
22	0.105	809		17116	24.6.173.22	23.62.228.65	54	TCP	35318-443 [ACK] Seq=809
23	0.106	809		20036	24.6.173.22	23.62.228.65	54	TCP	[TCP ACKed unseen segment]
24	0.106	20036	21496	809	23.62.228.6	24.6.173.220	1514	TLSv	[TCP Previous segment not captured]
25	0.106	21496	22956	809	23.62.228.6	24.6.173.220	1514	TLSv	Ignored Unknown Record
26	0.107	809		22956	24.6.173.22	23.62.228.65	54	TCP	[TCP ACKed unseen segment]
27	0.121	22956	24416	809	23.62.228.6	24.6.173.220	1514	TLSv	Ignored Unknown Record

As you examine the SEQ# values, there are jumps in values. But no Retransmissions, Duplicate ACKs, or Fast Retransmissions can be observed. As far as the TCP peers are concerned, no packets have been lost. The spanned switch is not sending all the traffic down the port to which Wireshark is connected.

Topic 162: Causes of Keep Alives

In this topic, we explain Keep Alives and their causes.

Keep Alives are used refer to TCP Keep Alive Probe packets. These packets are used to detect dead connections, detect dead TCP peers, and prevent a connection from terminating when idle. Keep Alives are implemented as ACK packets that are either empty or contain 1 byte of data. The sequence number value in a Keep-Alive packet is 1 less than the next expected sequence number. TCP SYN, FIN and RST packets cannot be considered to be Keep Alives.

How Wireshark detects Keep-Alives?

Wireshark can easily detect Keep Alives because it keeps a track of the Sequence Number field values in all TCP streams. The Notes column of Wireshark's Expert Info window provides information about Keep Alives/Keep Alive ACKs.

What Causes Keep Alives?

Applications written to use Keep Alives generate Keep Alives. Three parameters are defined for Keep Alives:

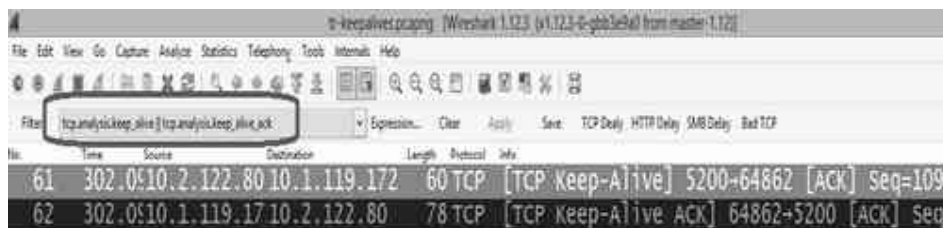
- 1) Keep Alive Time:** The amount of time between the last data packet and the first Keep Alive probe.
- 2) Keep Alive Interval:** Interval between Keep Alive Probes.
- 3) Keep Alive Probes:** Number of unacknowledged Keep Alive Probes that should be sent before considering the connection to be dead.

Topic 163: Wireshark Lab 61

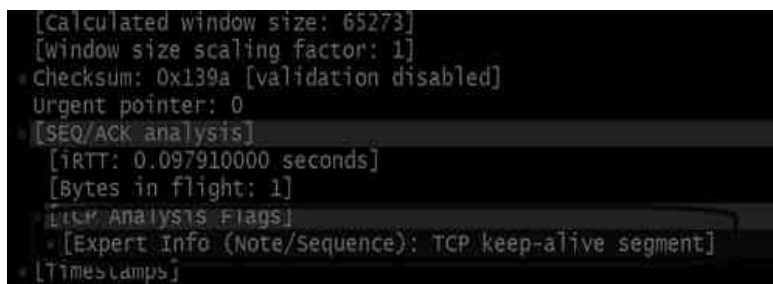
This topic uses a filter to count Keep Alive/Keep Alive ACK Packets in Wireshark.

Step 1: Open *tr-keepalives.pcapng*.

Step 2: Enter the filter `tcp.analysis.keep_alive || tcp.analysis.keep_alive_ack` in the display filter area. Click **Apply**.



We can see that only two packets match the filter.



Step 3: Display **TCP Delta** column if it is hidden. Otherwise, create it. If you look at the TCP Delta before Packet 61, then you can see that 10.2.122.80 has a 300 second Keep Alive Time value. Packet 62 is the Keep Alive ACK packet.



Step 4: Remove the filter by clicking the **Clear** button.

Topic 164: Wireshark Lab 62

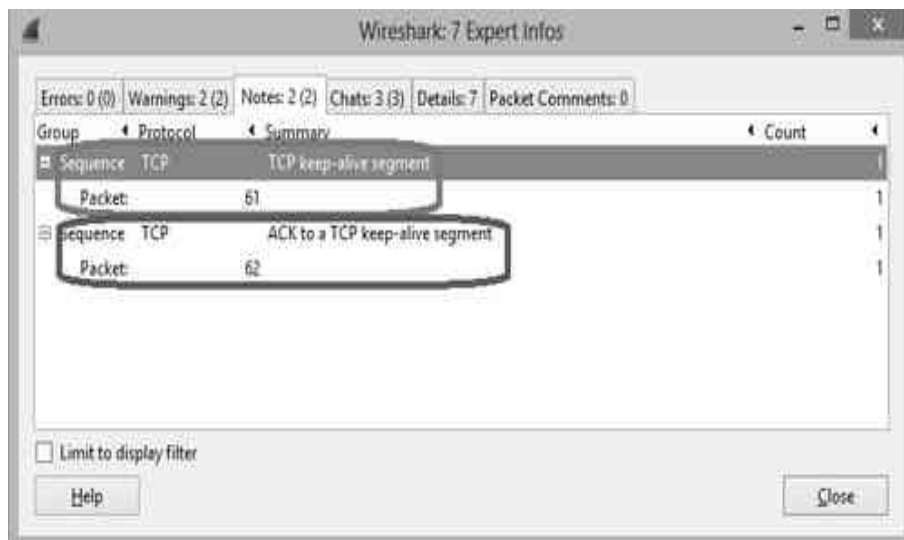
In this topic, we find Keep Alive/Keep Alive ACK Packets with Expert Infos in Wireshark.

Keep Alives and Keep Alive ACKs are typically used to check for dead TCP peers and avoid time out of idle connections. They are not indications of problems.

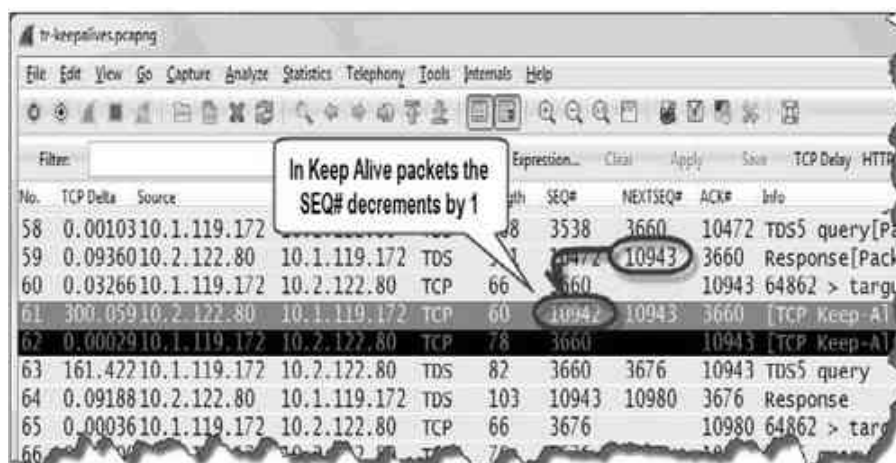
Step 1: Open *tr-keepalives.pcapng*.

Step 2: Click the **Expert Infos** button on the Status Bar.

Step 3: Click the **Notes** tab. You can see **Keep Alives** and **Keep Alive ACKs**. Expand the **Keep Alive** and **Keep Alive ACK** lines. You can see Packets 61 and 62 listed.



Step 4: Click on the **Keep Alive** entry (Packet 61). This will take you to the main Wireshark window. Click the **Close** button to close the Expert Infos window. Display the **SEQ#**, **NEXTSEQ#** and **ACK#** columns. For Packet 61, you can find that Sequence Number field value has been decremented by 1 from 10,943 to 10,942.

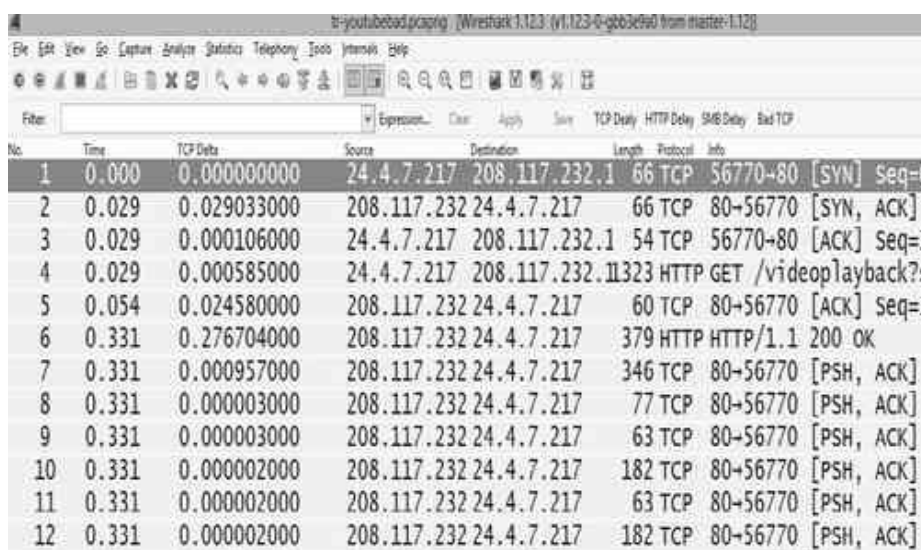


Topic 165: Wireshark Lab 63

In this topic, we identify Keep Alive Packets used in Zero Window Conditions in Wireshark.

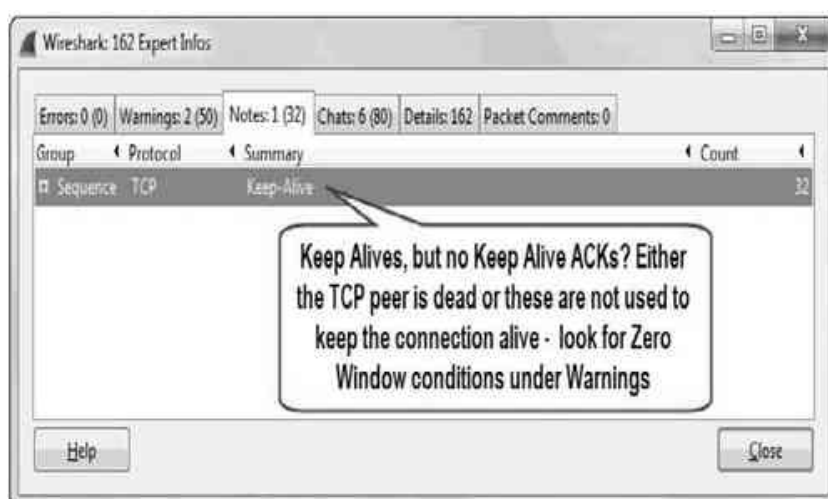
The two sides of a TCP conversation advertise their receive buffer space in the Window Size Value field. It is a 2-byte field whose maximum value can be 65,535 bytes. Over today's high-speed links, if a receiving application is slow in pulling data out of the receive buffer, it's not unusual that the advertised Window Size value can drop to zero – Zero Window Condition. Hosts can send TCP Keep Alives in place of Window Zero Probes. When a TCP host sends a Keep Alive to a peer that is advertising a Zero Window condition. You will not see Keep Alive ACK responses.

Step 1: Open *tr-youtubebad.pcapng*.



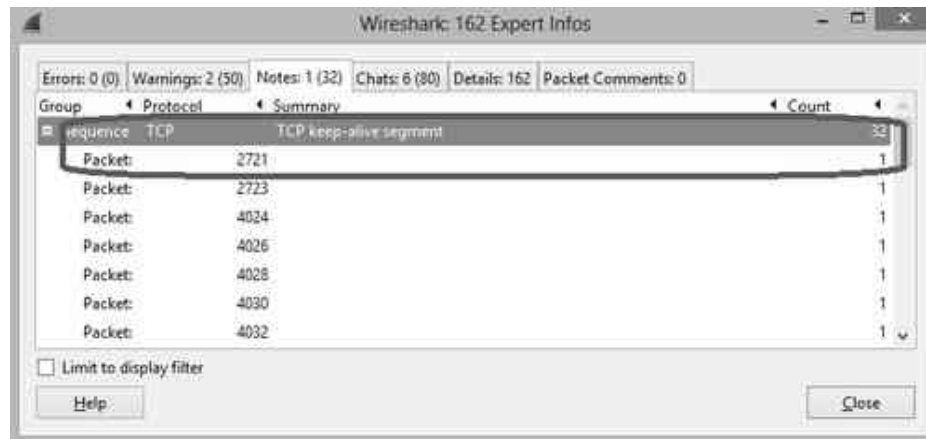
No.	Time	TCP Data	Source	Destination	Length	Protocol	Info
1	0.000	0.000000000	24.4.7.217	208.117.232.1	66	TCP	56770→80 [SYN] Seq=0
2	0.029	0.029033000	208.117.232	24.4.7.217	66	TCP	80→56770 [SYN, ACK]
3	0.029	0.000106000	24.4.7.217	208.117.232.1	54	TCP	56770→80 [ACK] Seq=1
4	0.029	0.000585000	24.4.7.217	208.117.232.1	323	HTTP	GET /videoplayback?e
5	0.054	0.024580000	208.117.232	24.4.7.217	60	TCP	80→56770 [ACK] Seq=1
6	0.331	0.276704000	208.117.232	24.4.7.217	379	HTTP	HTTP/1.1 200 OK
7	0.331	0.000957000	208.117.232	24.4.7.217	346	TCP	80→56770 [PSH, ACK]
8	0.331	0.000003000	208.117.232	24.4.7.217	77	TCP	80→56770 [PSH, ACK]
9	0.331	0.000003000	208.117.232	24.4.7.217	63	TCP	80→56770 [PSH, ACK]
10	0.331	0.000002000	208.117.232	24.4.7.217	182	TCP	80→56770 [PSH, ACK]
11	0.331	0.000002000	208.117.232	24.4.7.217	63	TCP	80→56770 [PSH, ACK]
12	0.331	0.000002000	208.117.232	24.4.7.217	182	TCP	80→56770 [PSH, ACK]

Step 2: Click the **Expert Infos** button on the Status Bar and then click the **Notes** tab.



During a successful keep alive process, hosts exchange Keep Alives and Keep Alive ACKs. In this case, either the peer is dead or this is not a standard keep alive process.

Step 3: If you expand the **Keep-Alive** section, you will find **Packet 2,721** the first Keep Alive listed. Click on it and then click the **Close** button to return to the main Wireshark window.



No.	Time	TCP Data	Source	Destination	Length	Protocol	Info
2716	2.485	0.000004000	208.117.232.24	24.4.7.217	1514	TCP	80→56770 [ACK] Seq=...
2717	2.485	0.000003000	208.117.232.24	24.4.7.217	1514	TCP	80→56770 [ACK] Seq=...
2718	2.485	0.000097000	24.4.7.217	208.117.232.1	54	TCP	56770→80 [ACK] Seq=...
2719	2.486	0.000769000	208.117.232.24	24.4.7.217	1514	TCP	[TCP window Full]
2720	2.688	0.202321000	24.4.7.217	208.117.232.1	54	TCP	[TCP ZeroWindow]
2721	3.095	0.407261000	208.117.232.24	24.4.7.217	60	TCP	[TCP Keep-Alive]
2722	3.095	0.000049000	24.4.7.217	208.117.232.1	54	TCP	[TCP ZeroWindow]
2723	3.882	0.786594000	208.117.232.24	24.4.7.217	60	TCP	[TCP Keep-Alive]
2724	3.882	0.000048000	24.4.7.217	208.117.232.1	54	TCP	[TCP ZeroWindow]
2725	4.094	0.211823000	24.4.7.217	208.117.232.1	54	TCP	[TCP window Update]
2726	4.114	0.020629000	208.117.232.24	24.4.7.217	1514	TCP	80→56770 [ACK] Seq=...
2727	4.115	0.000830000	208.117.232.24	24.4.7.217	1514	TCP	80→56770 [ACK] Seq=...

You can see that the first Keep Alive occurs almost 400 ms after a packet from the peer. It is not used to check for a dead peer. It is used to determine if Window Update (Window Size value has increased) has taken place.

Topic 166: Causes of Reused Ports

In this topic, we explain reused ports and their causes.

Sometimes Reused Ports are not a problem. Other times Reused Ports can cause significant delays in communications. Mostly, we can reuse port numbers without any problem as long as the previous TCP connection is terminated. Let's assume a client establishes a TCP connection with a server using source port number 1026 and destination port number 80. After a while, without terminating the previous connection, it tries to establish another new connection with same port number. In this case, if the first connection had been terminated, then reused ports would have not been a problem. Reused Ports become issue, when previous connection has not terminated (through TCP FINs/RSTs).

When Wireshark detects a previous SYN packet using the same IP address/port number combination in a trace file, then it marks SYN packets with the Reused Ports Expert Analysis definition.

What Causes Reused Ports?

Reused ports can be seen if

- A) An application defines a static source port number, or
- B) An application uses a very small range of source port numbers.
- C) A host is going through a lot of source port numbers very quickly.

Topic 167: Wireshark Lab 74

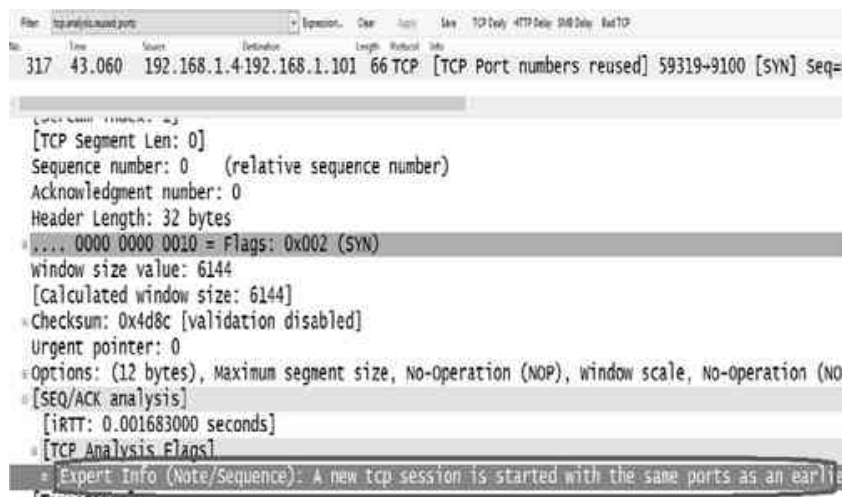
This topic uses a filter to count reused port Packets in Wireshark.

Step 1: Open *tr-reusedports.pcapng*.

Step 2: Type the filter **tcp.analysis.reused_ports** in the display filter area. Click **Apply**. In this trace file, Wireshark has detected one Reused Port as indicated by Status Bar.



Step 3: Expand the **[SEQ/ACK analysis]** section of the reused port packet whose number is 317. This area is colored cyan. Wireshark associates cyan with Expert Infos **Notes**.



Step 4: Let's remove filter by clicking the **Clear** button.

Topic 168: Wireshark Lab 75

This topic finds Reused Ports with Expert Infos in Wireshark.

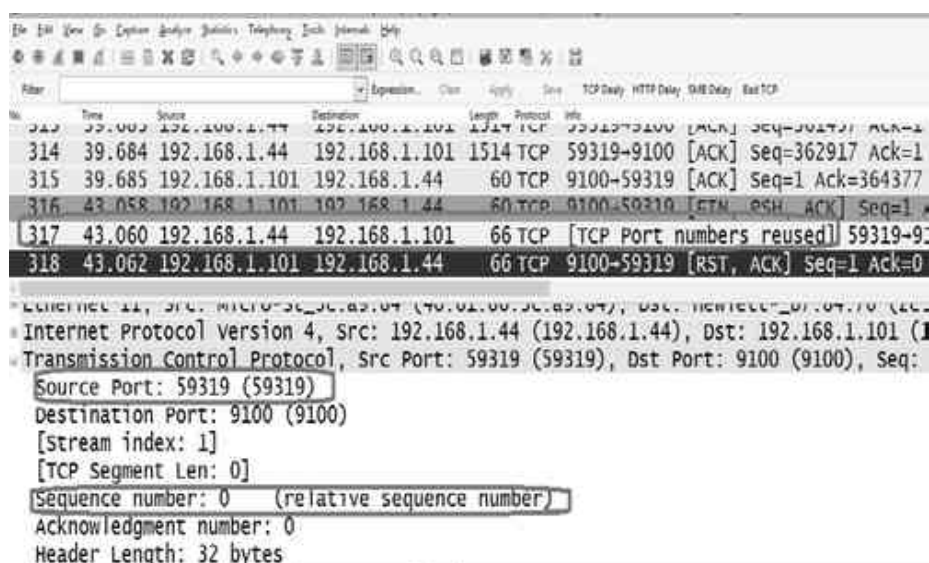
Step 1: Open *tr-reusedports.pcapng*.

Step 2: Click the **Expert Infos** button on the Status Bar.

Step 3: Click the **Notes** tab. Expand the **A new tcp session is started with the same ports as an earlier session in this trace** section. Click on Packet 317. Close the Expert Infos window by clicking **Close**. Wireshark will take you to main Wireshark window.



Step 4: As expected, the Reused Ports packet shown is a SYN packet. This packet uses IP address 192.168.1.44 and port 59319.



Step 5: The sequence number field can be used to determine whether the SYN belongs to a unique connection request, a retransmission or a reused port. Wireshark applies a relative sequence number to TCP. In the Packet Details pane of any packet, right-click on the **TCP header**. Select **Protocol Preferences** and toggle **Relative sequence numbers** off.

Step 6: To view the sequence number value of SYN packets. Enter **tcp.flags==0x0002** in the display filter area and click **Apply**. You will find only 2 packets.

Step 7: Display your **SEQ#** column to view the sequence numbers quickly. The sequence numbers are different. Packet 317 would be a retransmission if they were the same.



No.	Time	Source	Destination	Length	Protocol	Info
1	0.000	1851490611	192.168.1.44	66	TCP	59319→9100 [SYN] Seq=1851
317	43.060	1147483647	192.168.1.44	66	TCP	[TCP Port numbers reused]

Step 8: To turn the **Relative sequence numbers** preference setting on, click on a **TCP header** in the Packet Details pane, select **Protocol Preferences**.

Topic 169: Causes of Checksum Errors

This topic explains checksum errors and their causes.

A checksum is used to detect errors (e.g., flipped bits) in a transmitted segment. In TCP/IP model, UDP, TCP, IPv4 and link-layer protocols check against errors using a checksum. By default, Wireshark does not perform TCP, UDP or IPv4 checksum validation. Wireshark's Checksum validation processes can be enabled/disabled easily using the Preference settings. To enable checksum validation for IPv4, TCP and UDP, select the **Preferences** button on the Main Toolbar. Expand the Protocols section and select **IPv4**. Check the **Validate the IPv4 checksum if possible** preference setting.



Causes of Checksum Errors

A faulty Network Interface Card (NIC) or any device that alters the content of the packets along the path. On a capturing device, task Offloading. By task offload, we mean that numerous processes are offloaded to a network interface card to free up a host's CPU for other tasks. Assume you are capturing your own traffic to and from your machine on which Wireshark is running. You would see Checksum Errors on all outbound traffic if

- (a) Task offloading is enabled on that host, and
- (b) Wireshark checksum validation processes are enabled.

Topic 170: Wireshark Lab 76

This topic detects checksum errors with Expert infos in Wireshark.

Step 1: Open *tr-checksums.pcapng*.

Step 2: To enable checksum validation for IPv4, TCP and UDP, select the **Preferences** button on the Main Toolbar.

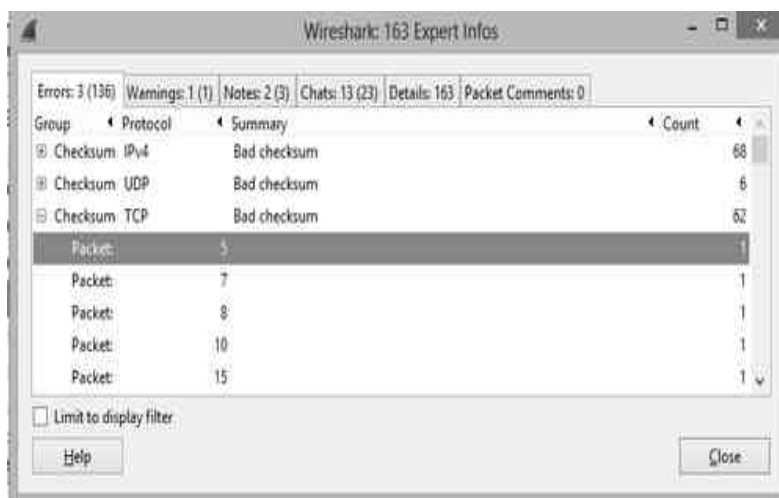
Step 3: Expand the Protocols section and select **IPv4**. Check the **Validate the IPv4 checksum if possible** preference setting.

Step 4: Repeat step 3 for **TCP**.

Step 5: Repeat step 3 for **UDP**. Click **OK** to save these preference settings. You will see that the coloring in the Packet List pane changes. Packets from 192.168.1.72 have a black background and red foreground.

Step 6: Click the **Expert Infos** button on the Status Bar.

Step 7: Go to **Errors** tab. You can see 68 bad IPv4 checksums, 6 bad UDP checksums and 62 bad TCP checksums. Expand the **TCP Bad Checksum** section.



Step 8: **Packet 5** is the first TCP Bad Checksum entry. Click on it. This will take you to the main Wireshark window. You can see that the packets from 192.168.1.72 when arrived at the target do not contain error. The target acknowledged receipt of the packets, processed them, and replied to the requests. Host with 192.168.1.72 is configured to use task offloading.

Step 9: Disable, IPv4, UDP and TCP checksum validation settings.

Topic 171: Wireshark Lab 77

This topic uses DNS errors Filter in Wireshark.

Two most common DNS errors are 1) a server failure (Reply Code 2), and 2) a name error, listed as Non-Existent Domain, (Reply Code 3).

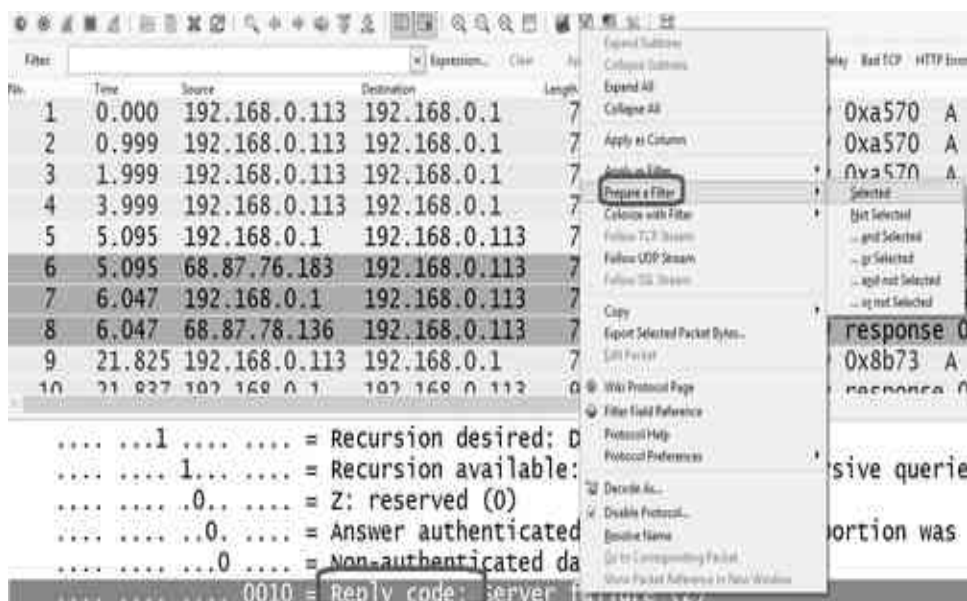
To quickly identify DNS errors in a trace file, let's create a button.

Step 1: Open *tr-dnserrors.pcapng*.

Step 2: **Packet 5** is the first DNS response. Click on it. From the Info column, we can see that it is a Server Failure reply.

Step 3: In **Packet 5**, right-click on the **Domain Name System (response)** line and select **Expand Subtrees**. You will find the DNS Reply Code field inside the Flags section.

Step 4: Right-click on the **Reply code** field and select **Prepare a Filter | Selected**.

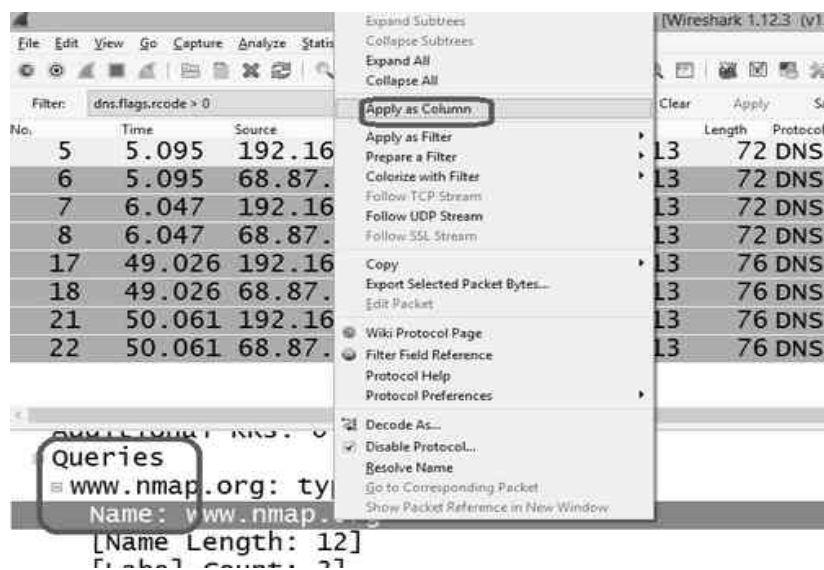


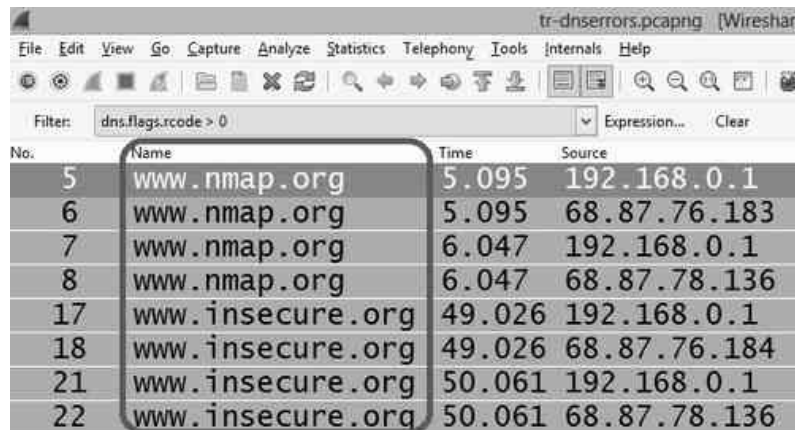
Using this filter will locate Server Failures. As we want to create a button to display DNS errors. All such packets will have values larger than 0 in the **dns.flags.rcode** field.

Step 5: Type **dns.flags.rcode > 0** and click the **Save** button. Rename the button's name to **DNS Errors** and click **OK**.

Step 6: Click the new **DNS Errors** button. You will find that there are eight DNS error responses in this trace file.

Step 7: Expand the **Queries** section in Packet Details pane of a response. Right click on the **Name** field and **Apply as Column**. This will determine what name(s) generated these responses





No.	Name	Time	Source
5	www.nmap.org	5.095	192.168.0.1
6	www.nmap.org	5.095	68.87.76.183
7	www.nmap.org	6.047	192.168.0.1
8	www.nmap.org	6.047	68.87.78.136
17	www.insecure.org	49.026	192.168.0.1
18	www.insecure.org	49.026	68.87.76.184
21	www.insecure.org	50.061	192.168.0.1
22	www.insecure.org	50.061	68.87.78.136

The errors can be because of a server upstream from our local server is not responding to recursive DNS queries.

Step 8: Click the **Clear** button to remove filter.

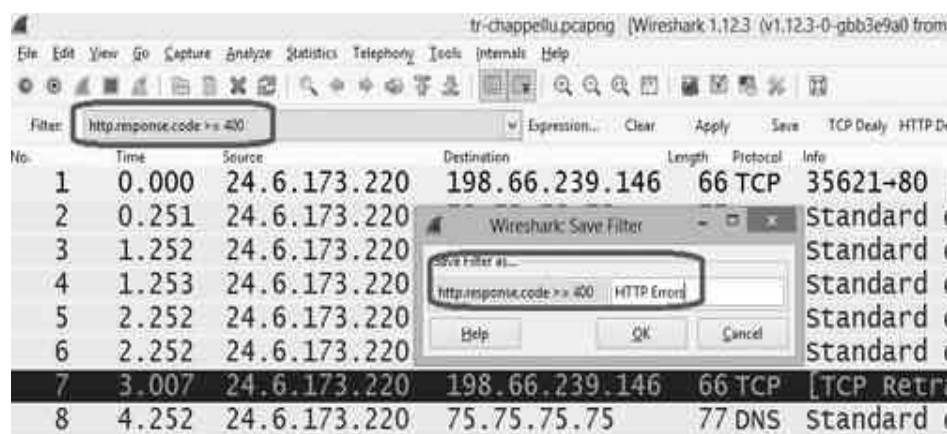
Topic 172: Wireshark Lab 78

This topic uses an HTTP errors filter in Wireshark.

We will create a button, which when clicked will quickly identify HTTP errors in your trace files.

Step 1: Open *tr-chappellu.pcapng*.

Step 2: Type **http.response.code >= 400** in the display filter area and click the **Save** button. Let's name the button **HTTP Errors** and click **OK**.



No.	Time	Source	Destination	Length	Protocol	Info
1	0.000	24.6.173.220	198.66.239.146	66	TCP	35621->80
2	0.251	24.6.173.220				Standard
3	1.252	24.6.173.220				Standard
4	1.253	24.6.173.220				Standard
5	2.252	24.6.173.220				Standard
6	2.252	24.6.173.220				Standard
7	3.007	24.6.173.220	198.66.239.146	66	TCP	[TCP Retra
8	4.252	24.6.173.220	75.75.75.75	77	DNS	Standard

Step 3: Click the newly created **HTTP Errors** button. This will identify the two HTTP error responses in this trace file. The codes are 404 “Not Found” errors.



Step 4: Right-click on **Packet 61** and select **Follow TCP Stream**. This will allow us to figure out what item was not found on the server. **Follow TCP Stream** applies a filter based on the Stream Index number. Also, a window is opened that depicts the conversation without headers.



Step 5: Click **Clear** to remove the filter.

Figures and Material used for Part3 (from Topics 173 to 192) have been adapted from A. Jesin's "Packet Tracer Network Simulator", 2014.

Topic 173: Introduction to Packet Tracer

This topic provides an introduction to Packet Tracer.

Packet Tracer is a network simulator from Cisco. It allows you to get a taste of everything in Cisco without needing to purchase real hardware. With Packet Tracer, complex topologies with tens of Cisco devices can be designed and troubleshot. You can also watch the packets moving between them. A wide range of Cisco switches and routers, several end devices such as PCs and servers and wireless devices from Linksys, are available in Packet Tracer.

Layer	Cisco Packet Tracer Supported Protocols
Application	FTP, SMTP, POP3, HTTP, TFTP, Telnet, SSH, DNS, DHCP, NTP, SNMP, AAA, ISR VOIP, SCCP config and calls, ISR command support, Call Manager Express
Transport	TCP and UDP, TCP Nagle Algorithm & IP Fragmentation, RTP
Network	BGP, IPv4, ICMP, ARP, IPv6, ICMPv6, IPSec, RIP v1/v2/ng, Multi-Area OSPF, EIGRP, Static Routing, Route Redistribution, Multilayer Switching, L3 QoS, NAT, CBAL, Zone-based policy firewall and Intrusion Protection System on the ISR, GRE VPN, IPSec VPN
Network Access/Interface	Ethernet (802.3), 802.11, HDLC, Frame Relay, PPP, PPPoE, STP, RSTP, VTP, DTP, CDP, 802.1q, PAgP, L2 QoS, SLARP, Simple WEP, WPA, EAP

Packet Tracer Workspaces

Two workspaces are supported; logical and physical. In the logical workspace, we can build logical network topologies by placing, connecting, and clustering virtual network devices. The physical workspace provides a sense of scale and placement in how network devices such as routers, switches, and hosts would look in a real environment.

Packet Tracer Modes

It supports two operating modes—real-time mode and simulation mode. In real-time mode, all network activities take place with immediate real-time response. The simulation mode allows a user to control time intervals, and the propagation of data across a network.

Installation of Packet Tracer

You can download the latest version of Packet Tracer for free from www.netacad.com/about-networking-academy/packet-tracer.

Topic 174: Packet Tracer's Interface Overview

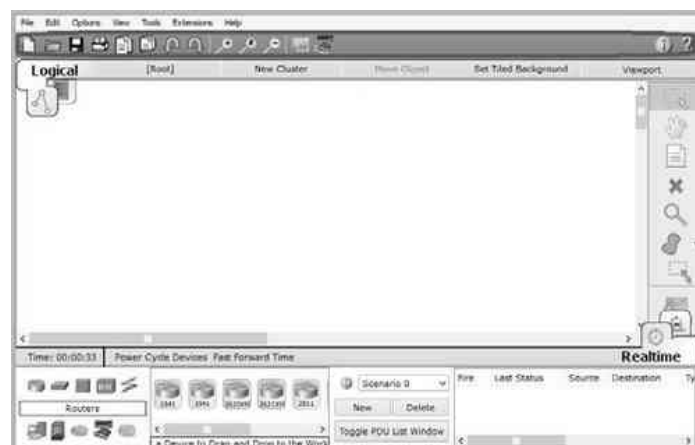
This topic has a look at the Interface of Packet tracer.

The layout of Packet Tracer can be divided into several components.

Menu bar: This is a commonly found menu. It is used to open, close, print, save, change preferences, and so on.



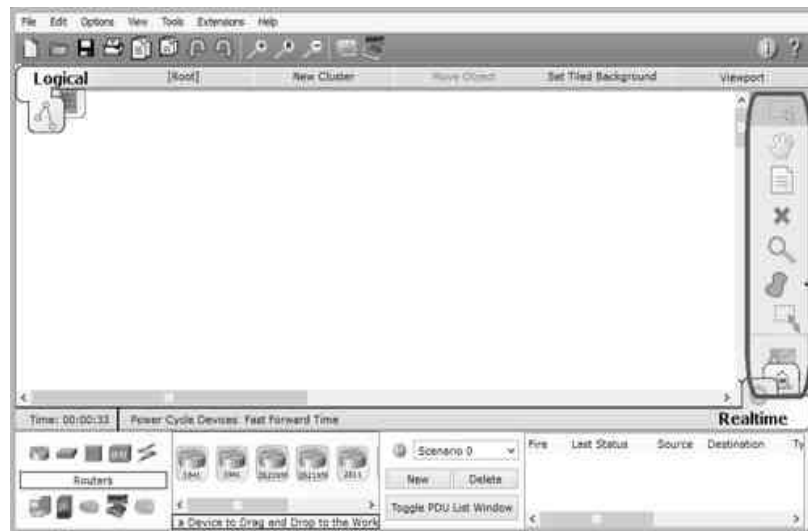
Main toolbar: This bar contains shortcut icons to menu options that are frequently accessed. For example, open, save, zoom, undo, and redo.



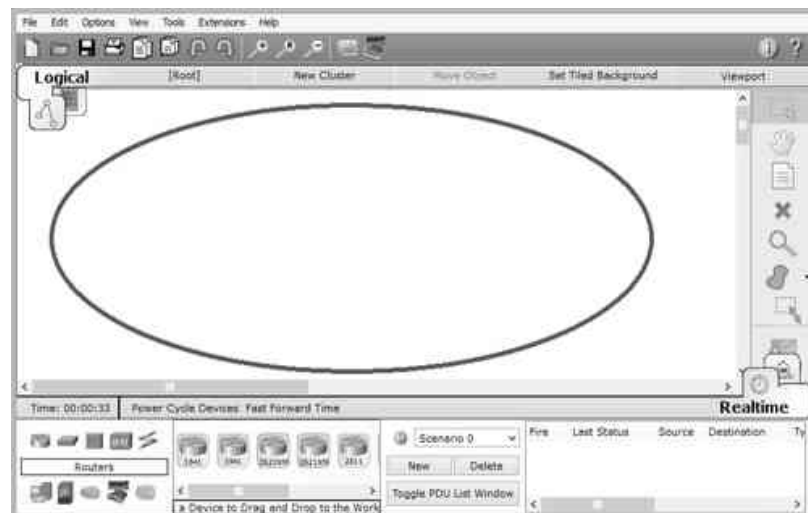
Physical/Logical workspace tabs: These tabs allow you to choose between the Logical and Physical work areas.



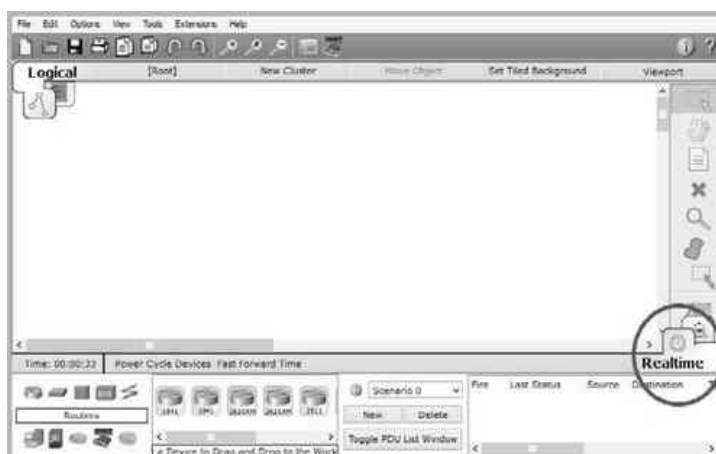
Common tools bar: With help of this toolbar, we can manipulate topologies. For example, select, move layout, place note, delete, resize shape, add simple/complex Protocol data unit (PDU).



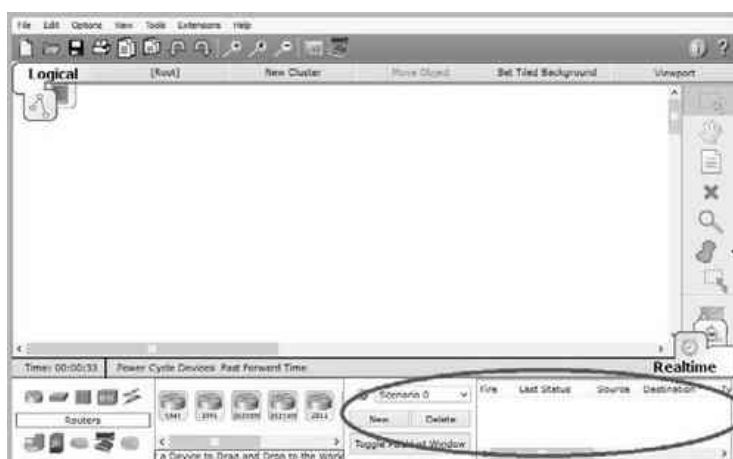
Workspace: Here, we create topologies and display simulations.



Real-time/Simulation tabs: With these tabs, the operating mode can be switched between the real and simulation modes. Buttons are also provided to control the time.



User-created packet box: From this area, you can create highly-customized packets in order to test your topology.



Network component box: All of the network and end devices available in Packet Tracer are listed in this component. It can be sub-divided into two areas:



Device-type selection box: Device categories are given here.

Device-specific selection box: Given a device category, this box displays different device models within that category.

Topic 175: Creating a Simple topology

In this topic, we create our first simple topology in Packet Tracer.

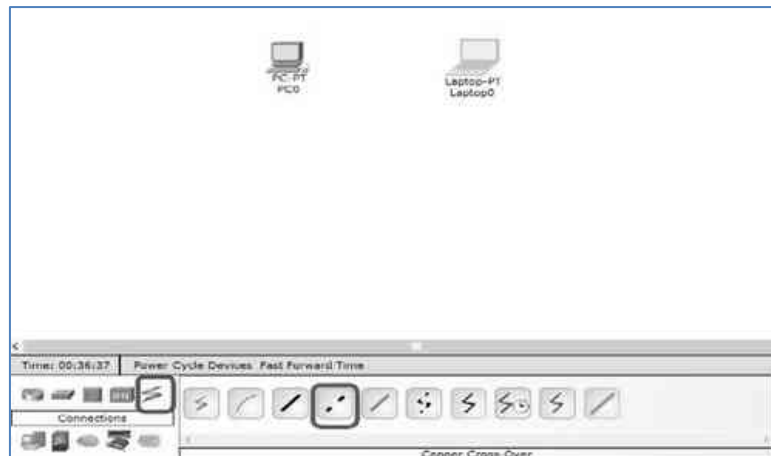
Step 1: Run Packet Tracer.

Step 2: Click **End Devices** in **Device-type selection box**. This will select the end devices category in **Device-specific selection box**.



Drag-and-drop a **Generic** PC icon into the Workspace. Do the same for a **Generic** laptop icon.

Step 3: Click on **Connections** in **Device-type selection box**. Then, click on **Copper Cross-Over** in **Device-specific selection box**.

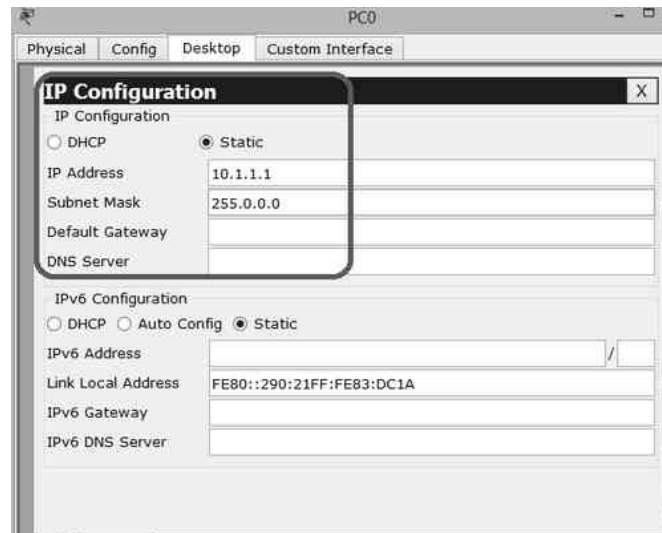


Now, click on **PC0**, and select **FastEthernet**. Click on **Laptop0** and select **FastEthernet**. When the link is up, the color of link status LED becomes green.

Step 4: Click on the PC, and go to the **Desktop** tab.

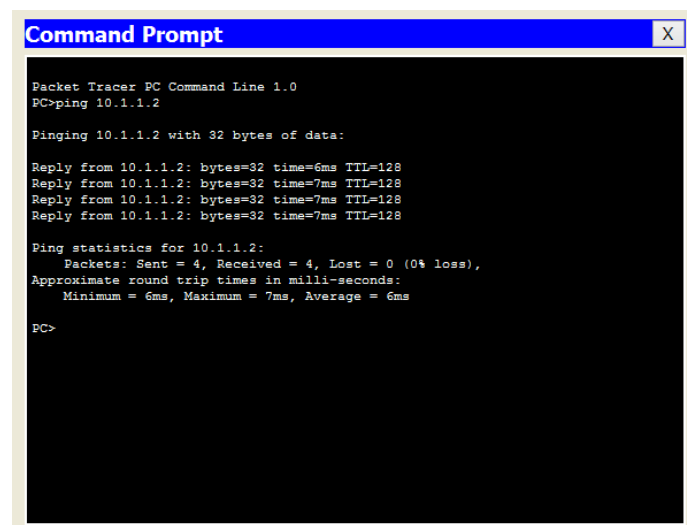


Click on **IP Configuration**, and enter an IP address and subnet mask. No information required for a default gateway and DNS server. Close the window



Step 5: Repeat step 4 for the laptop. Make sure that both of the IP addresses are in the same subnet.

Step 6: Click on the laptop, open the **command prompt**, and ping the IP address of the device at the other (PC) end to check connectivity.



Step 7: This topology can be saved by going to **File | Save As**. The topology with the devices in the same state will be saved with a **.pkt**.

Topic 176: Introduction to Cisco and PT devices

This topic describes Cisco and PT devices available in Packet Tracer.

Network devices form the core of networking. Routers, switches are used to Interconnect end devices such as PCs, laptops, servers.

Routers in Packet Tracer: To examine the Cisco and **Generic** routers. Select **Routers** in the **device-type selection box**, then, go to the **device-specific selection box**.

Cisco 1841: An **Integrated Service Router (ISR)**. Contains two Fast Ethernet ports, two slots for **High Speed WAN Interface Cards (HWICs)**, and one slot for **Advanced Integration Module (AIM)**.

Cisco 1941: Similar to **Cisco 1841**. But runs on Cisco IOS Version 15. There are two ports that operate at Gigabit Ethernet speeds.

Cisco 2620XM: A multiservice router. It contains one Fast Ethernet port, two slots for WAN interface cards, and one slot for AIM.

Cisco 2621XM: Similar to Cisco 2620XM, except that this router has two Fast Ethernet ports.

Cisco 2811: An ISR router with two Fast Ethernet ports, four WIC slots, and a dual slot for AIM.

Cisco 2901: Two Gigabit Ethernet ports, four WIC slots, and two **Digital Signal Processor (DSP)** slots. Cisco IOS Version 15 runs on it.

Cisco 2911: Three Gigabit Ethernet ports and all the other features of **Cisco 2901**.

Generic Router: contains 10 slots, and is a custom router running on Cisco IOS.

Switches in Packet Tracer: To examine the Cisco and **Generic** switches. Select **Switch** in the **device-type selection box**, then, go to the **device-specific selection box**.

Cisco 2950-24: a managed switch. There are 24 Fast Ethernet ports.

Cisco 2950T-24: It has two Gigabit Ethernet ports in addition to the 24 Fast Ethernet ports.

Cisco 2960-24TT: It contains 24 ports and has **Small Form-factor Pluggable (SFP)** modules.

Cisco 3560-24PS: It is a layer 3 switch and performs routing and switching. The suffix **PS** means it supports **Power over Ethernet (PoE)**. It can power up IP phones without using power adapters.

Bridge PT: It is a device used to segment a network with only two ports.

Generic Switch PT: It is a customizable Packet-Tracer switch with 10 slots and several modules and runs on Cisco IOS.

Hub PT: It is a Packet Tracer device with 10 slots, and is an oldest way to connect end devices.

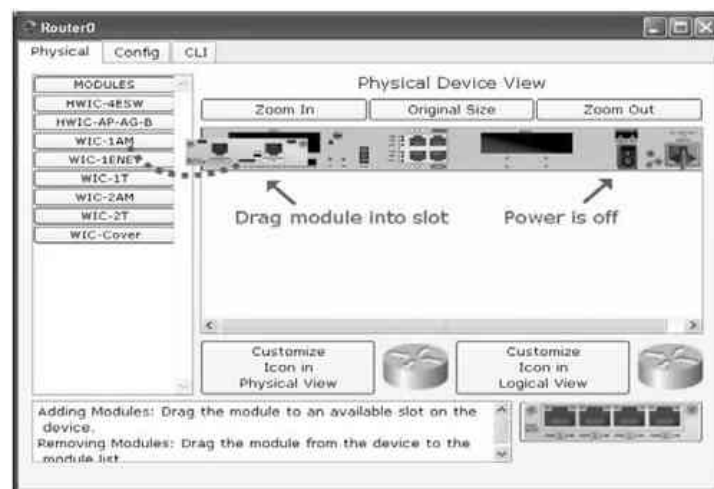
Repeater PT: It is used to boost the signal on a wire when the distance between two points is high.

Coaxial Splitter PT: It is used to split a single coaxial connector into two. It contains three coaxial ports.

Topic 177: Customizing Devices with Modules

In this topic, we customize devices with modules and learn naming conventions in PT.

A piece of hardware that contains several device interfaces e.g. **HWIC-4ESW** (High-Speed WAN Interface Card) is a module having four Ethernet (10 MBps) ports. There is a power switch located on the right-hand side of each device. If you see a green LED on the power switch, then it indicates that device is powered up. If you want to add a module, you need to click on this switch to turn the device off. This is similar to a real router/switch. Drag a module from the modules list and drop it onto an empty slot. The module will automatically return to the module list, in case it does not fit into that slot.



A module can be removed by first, powering off the device and then, dragging it from the slot back to the module list.

Naming convention for Interfaces:

Router Interfaces can be identified by their names. The interfaces can be broadly grouped as:

Copper Ethernet Interface: is a normal LAN interface, employs RJ-45 connector with a copper cable. **Ethernet** (10 MBps), **FastEthernet** (100 MBps), and **GigabitEthernet** (1000 MBps). A number followed by **E**, **FE**, **CE**, **CFE**, or **CGE** is used to identify modules with **Ethernet** interfaces. **SW** is used for those modules, which provide switching features when used on routers. **Examples:**

- **HWIC-4ESW** (four Ethernet switching ports)
- **WIC-1ENET** (single Ethernet port)
- **NM-1E** (single Ethernet port)
- **NM-1FE-TX** (single Fast Ethernet port)

Fiber Ethernet Interface: similar to **Copper Ethernet Interface** but employs fiber cables. **Examples:**

- **NM-1FE-FX** (single Fast Ethernet fiber media).

Serial Interface: synchronous modules are identified by **T** while asynchronous – **A/S**.

Examples:

- **WIC-1T** (a single synchronous serial port)

Modem Interface: these interfaces consist of RJ11 ports with analog telephone cables and are identified by **AM**.

Examples:

- **WIC-8AM** (eight RJ11 ports)

If you have created a device that you have customized with a set of modules, you can save it for future use. 1- Drag-and-drop a network device into the work area. 2- Turn the device off. 3-Add modules to it. 4- Click **Tools | Custom Devices Dialog**. 5- Click **Select**, and then click on the device that was just customized. 6- Provide a name, and then click on **Add** and **Save**. The device is saved with a .ptd extension.

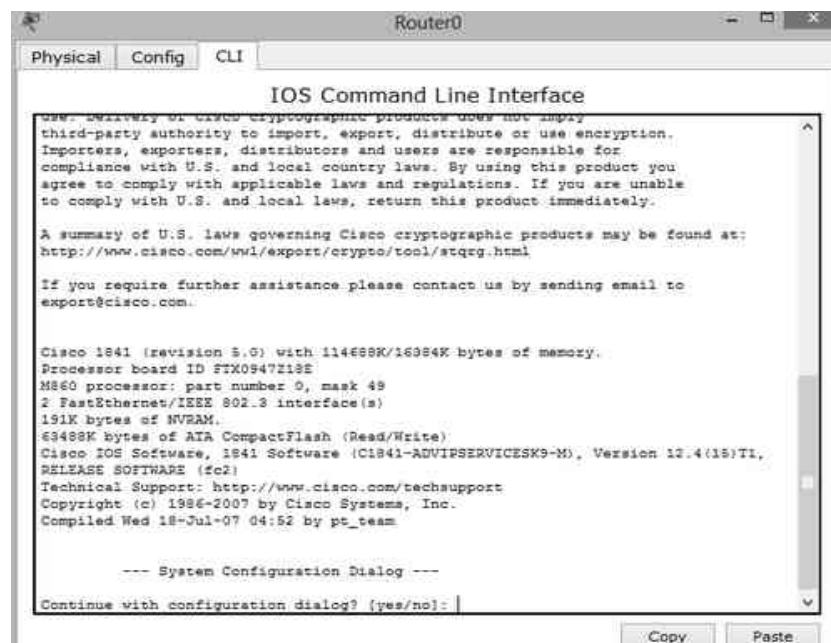
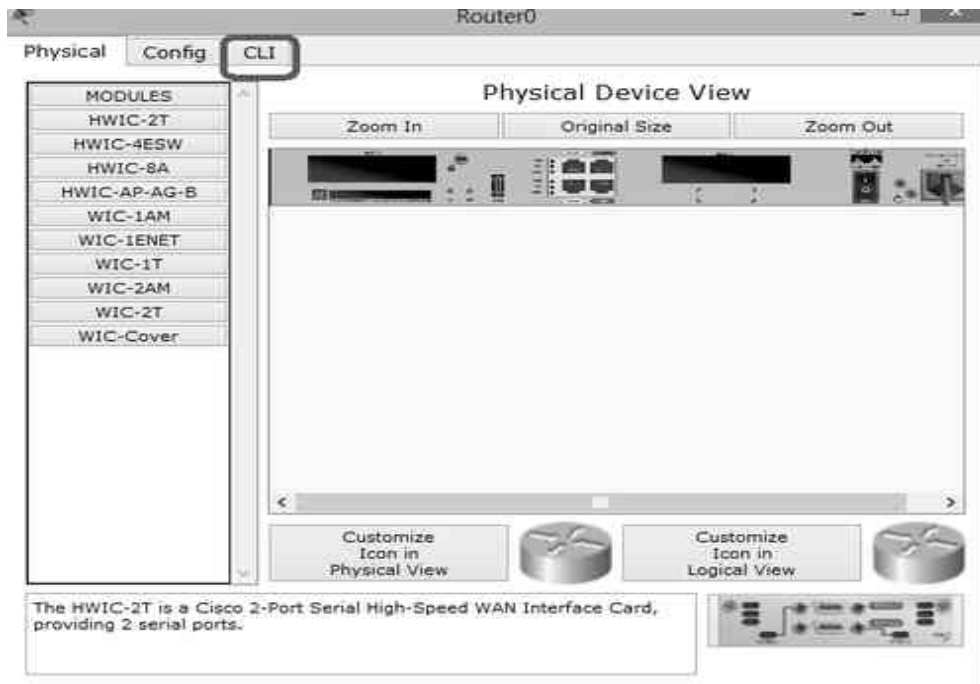
Topic 178: Accessing CLI of a Device

In this topic, we describe how to access Command Line Interface (CLI) of a device in Packet Tracer.

There are two ways to access the Command-line Interface of a device in Packet Tracer:

The CLI Tab: It can be used to configure devices e.g. you can assign IP addresses to router interfaces. You can use Paste and Copy buttons to copy text to and from the

command line. Click on a network device, go to the **CLI** tab to access the Cisco IOS command line interface, and you can enter commands as on a real device (router).



You should refer to the HELP files which lists the IOS commands supported by Packet Tracer.

The Console port: No difference with respect to CLI tab method. The use of console port makes the topology look similar to the real world. Let's illustrate with an example.

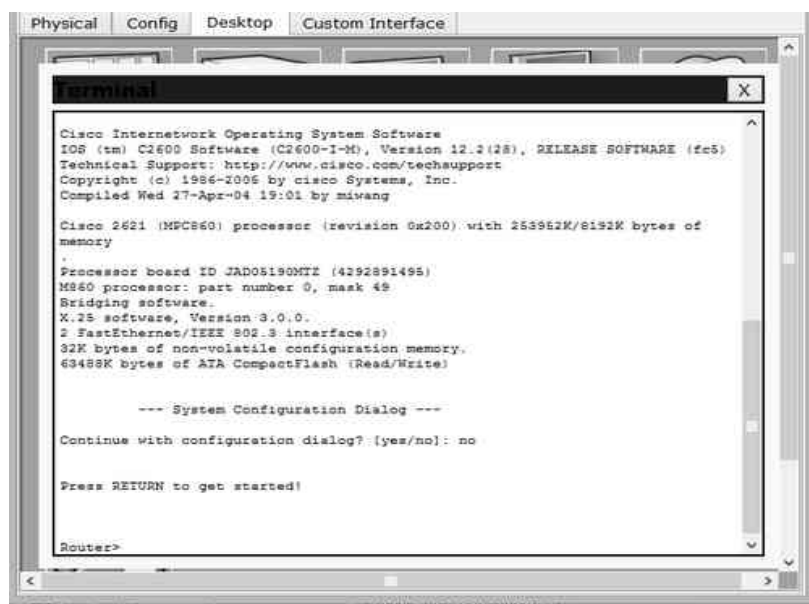
Step 1: Click **End Devices** in **Device-type selection box** and then drag and drop a PC or laptop from **Device-specific selection box** to the workspace.

Step 2: In a similar fashion, add a router, let's say **2621 XM**.

Step 3: Click **Connections** in **Device-type selection box** and then select console port from **Device-specific selection box** to the workspace.

Step 4: Connect the console cable of the network device to the RS-232 port of the PC/laptop.

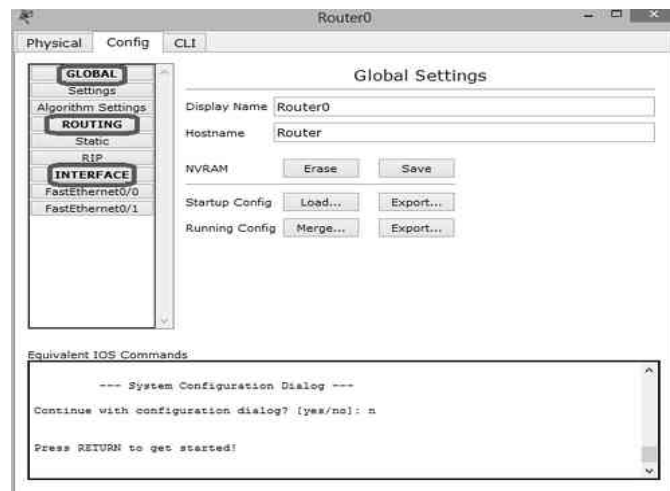
Step 5: Click the PC/laptop, go to the **Desktop** tab, and open **Terminal**. Click **OK** with the default settings to view the console. The router's console through its terminal is displayed:



Topic 179: Configuring Devices with Config Tab

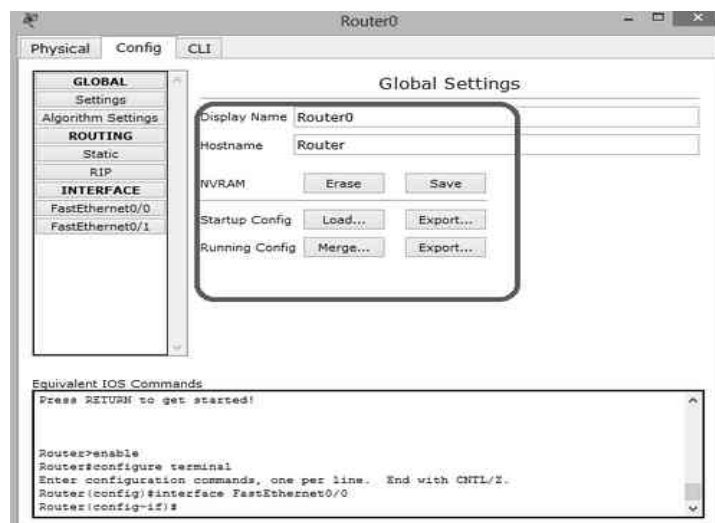
In this topic, we configure routers/switches using the config tab in Packet Tracer.

You can configure Cisco routers and switches without using a single command. This becomes possible by using a **Config** tab. The **Config** tab contains a Graphical User Interface (GUI) options for the most common configurations. As you fiddle with a GUI, the equivalent Cisco IOS commands are also displayed.



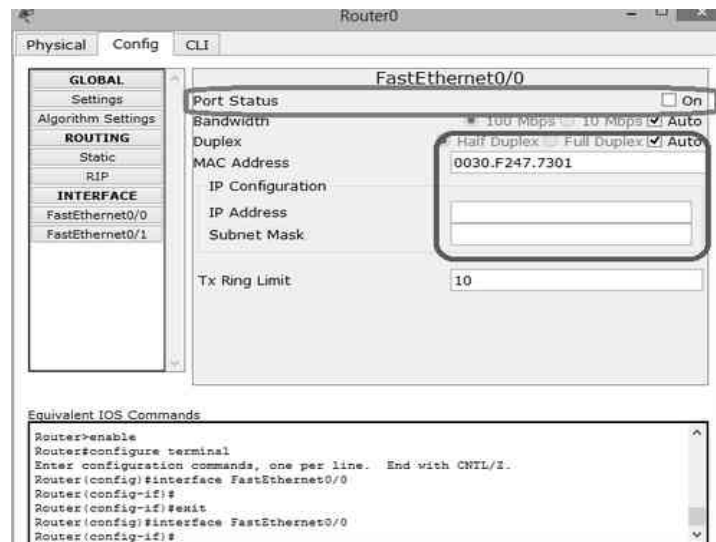
There are 3 general levels of configurations: Global, Routing, and Interface.

Global Settings: You can change the router's name. This can also be done directly in the workspace. The configuration file of a device can also be saved, erased, or exported.

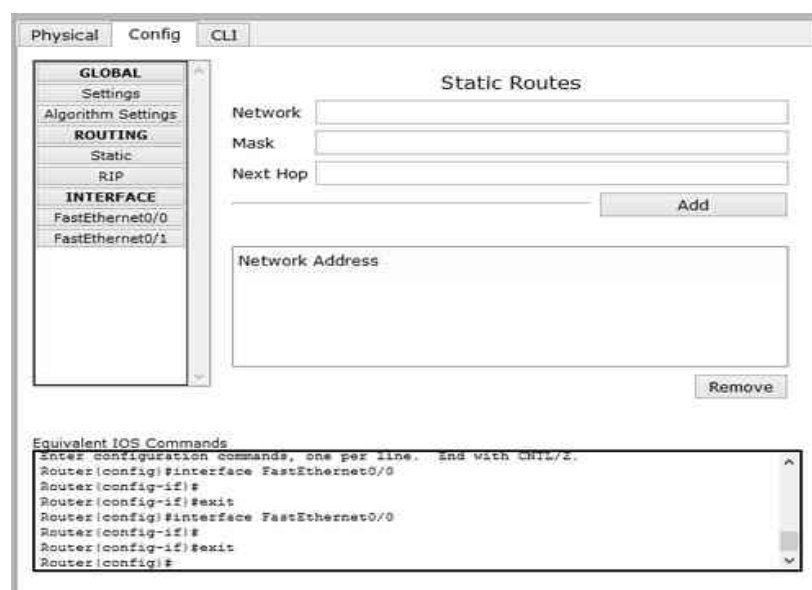


The **Global's Algorithm Settings** are meant for advanced users. You can minutely tweak your device to see how it responds to certain situations.

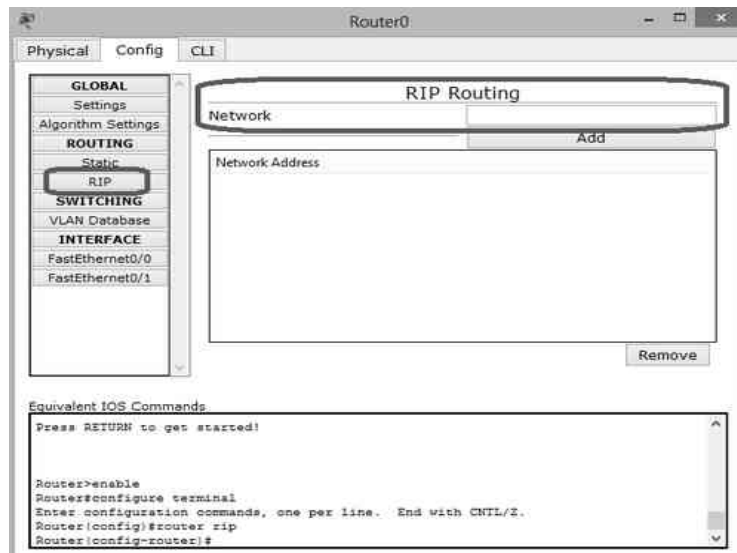
Interface Settings: The IP, MAC addresses and subnet mask can be entered here. To view the port status, you need to check the "On" of "Port status".



Routing Settings: Here, there are two options for configuring **static** and **dynamic** routing. For static routing configuration, you can enter network address, net mask, and its next hop address, and then click on **Add**.



Dynamic routing such as Routing Information Protocol (RIP) can be configured by adding only the network IP.



VLAN database: Available on a switch. It allows you to create and remove VLANs.

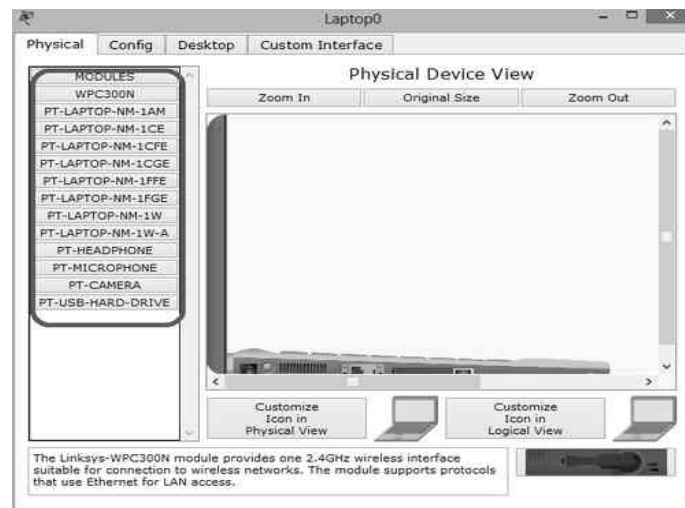
Topic 180: Generic IP End Devices in PT

This topic describes Generic IP End Devices in Packet Tracer.

Network devices such as switches, routers are the core of a network. End devices (PCs, servers) are the ones that use this core. PCs, laptops, tablets, personal digital assistant (PDAs), and a TV a wide range of end devices are available in PT. We can group them

- A) clients
- B) server
- C) Other devices

A) Clients: Desktops and laptops: As far as usability is concerned, there is no difference between them. Most usable and highly configurable client devices. Next, we describe those modules which are available for desktops and laptops. Addition and removal of a module require a device to be switched off.



Linksys-WMP300N: Through this wireless interface, you can configure a WLAN.

PC-HOST-NM-1AM: It can be used as a modem as it provides RJ11 interface.

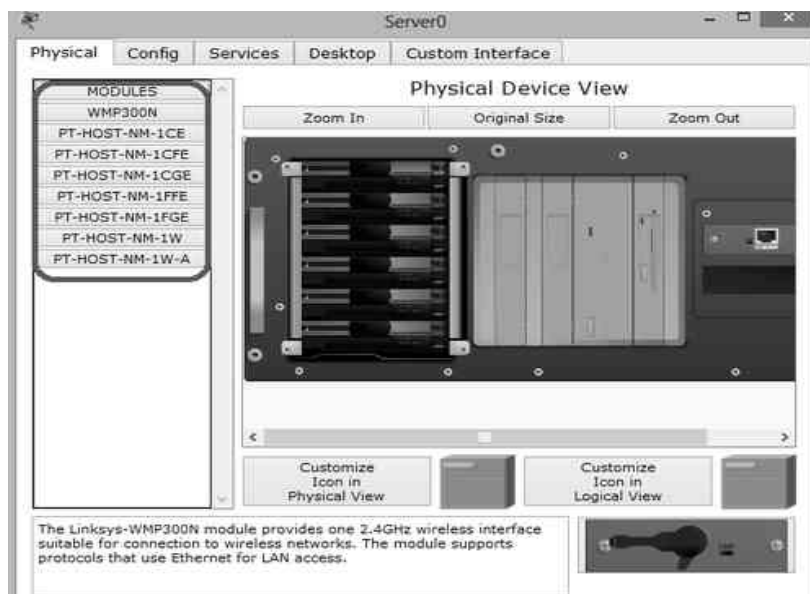
PC-HOST-NM-1CE, PC-HOST-NM-1CFE, PC-HOST-NM-1CGE: Through these modules, **Ethernet**, **FastEthernet**, and **GigabitEthernet** connections are provided respectively.

PC-HOST-NM-1FFE, PC-HOST-NM-1FGE: Fiber version of the previous module.

PC-HOST-NM-1W, PC-HOST-NM-1W-A: Wireless interface for WLAN. The first has a frequency of 2.4 GHz and the second has 5GHz.

PC-HEADPHONE, PC-MICROPHONE, PC-CAMERA, PC-USB-HARD-DRIVE: No functionality associated with them.

B) Servers: Space for two network interfaces. Modules available are the same as PC modules, except **PC-HOST-NM-1AM** module.



Let's look at services that are available for servers. **HTTP service:** Both HTTP and HTTPS (HTTP employing Secure Sockets Layer (SSL) protocol) protocols can be supported by a web server. You can also create and edit static HTML pages. **DHCP service:** can assign IP addresses to routers. You can create and edit DHCP pools of IP addresses. The default pool is called **serverPool**, which cannot be removed or edited. **DNS service:** resolves domain names to IP addresses. **AAA service:** Authentication, Authorization, and Accounting and supports RADIUS and TACACS authentication protocols. **NTP:** Network Time Protocol ensures that the clocks of all devices are synchronized properly. **EMAIL services:** **SMTP** and **POP3** services are supported. **FTP Services:** Users can be created and permissions can be granted to them. **Firewall:** You can configure rules based on source/destination IP addresses and source/destination port numbers. Connections can be allowed or denied.

Topic 181: Configuring End Devices in PT

This topic configures End devices in Packet Tracer.

Click on an end device in the workspace, then go to the **Desktop** tab. A lot of utilities are available here. These can be used for testing and debugging the network.

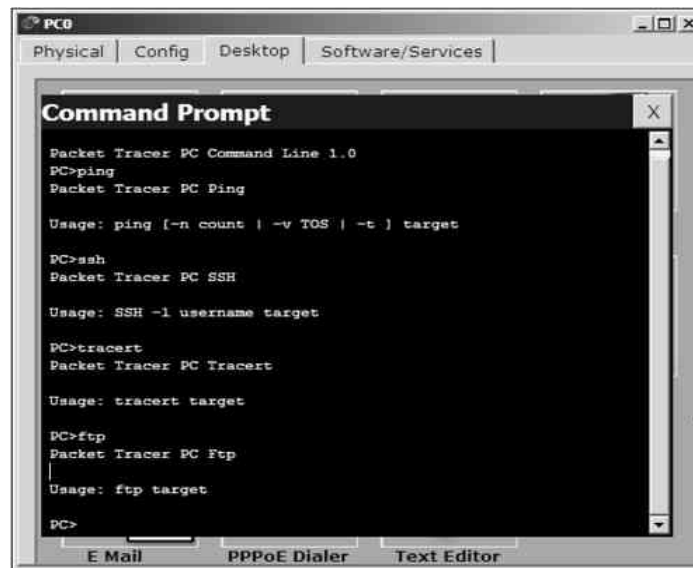


IP Configuration: With this utility, you can assign a dynamic or static IP address to an end device. If you choose to enter a static IP address, **Subnet Mask** field gets filled according to the class of the IP address. In case you want your devices to obtain IP dynamically, then choose **DHCP** here. Configure **DHCP** on **Server-PT**.

Dial-up: End devices such as **PC-PT** and **Laptop-PT** have the **PC-HOST-NM-1AM**. This utility allows to simulate a modem dialer.

Terminal: This utility can be used for accessing the CLI through the console port. **Server-PT** device does not enjoy this utility as it does not have an RS-232 interface.

Command Prompt: It simulates the command line offered by Windows OS. Commands such as **arp**, **delete**, **dir**, **ftp**, **help**, **ipconfig**, **netstat**, **nslookup**, **ping**, **ssh**, **telnet** **tracert**.



Web Browser: It can be used with a **Server-PT** configured with HTTP. It provides only **back**, **forward**, **go**, and **stop** buttons. Cache or history are not available.

PC Wireless: Using this utility, you can display information such as signal strength of a **Linksys-WMP300N** module.

VPN: Virtual private network (VPN) is used to create connection for secure communication. This works when a router has been configured as a VPN server.

Traffic Generator: With this utility, you can create customized packets and send them at periodic intervals. Similar to the **Add Simple Protocol Data Unit (PDU)** and **Add Complex PDU** tools.

MIB Browser: The Management Information Base (**MIB**) Browser utility allows you to generate Simple Network Management Protocol (**SNMP**) requests.

Cisco IP Communicator: A computer can be turned into an IP phone with the help of this utility. It is not available on the **Server-PT**.

Email: You can send and receive emails with the help of this utility.

Text Editor: You can use this utility to create, edit, and save text files.

Topic 182: Packet Tracer's Simulation Mode

This topic discusses simulation mode of Packet Tracer.

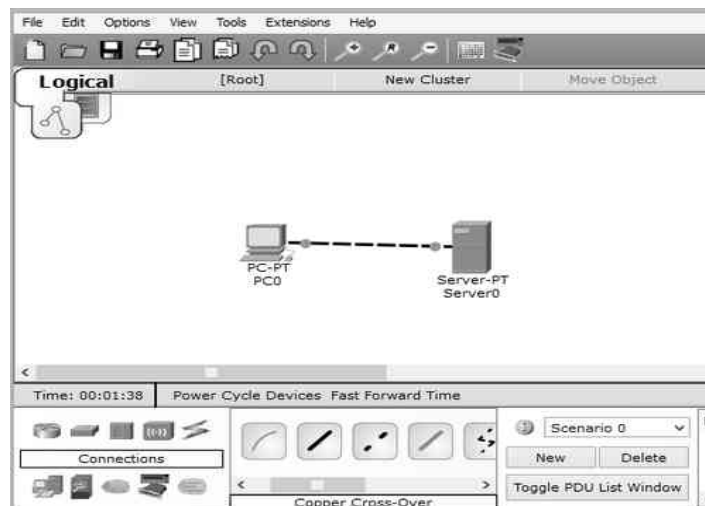
In Packet Tracer's simulation mode, you can observe packets flowing from one device to another. Also, when you can click on a packet, you can see detailed information about the TCP/IP layers.

Step 1: Open Packet Tracer

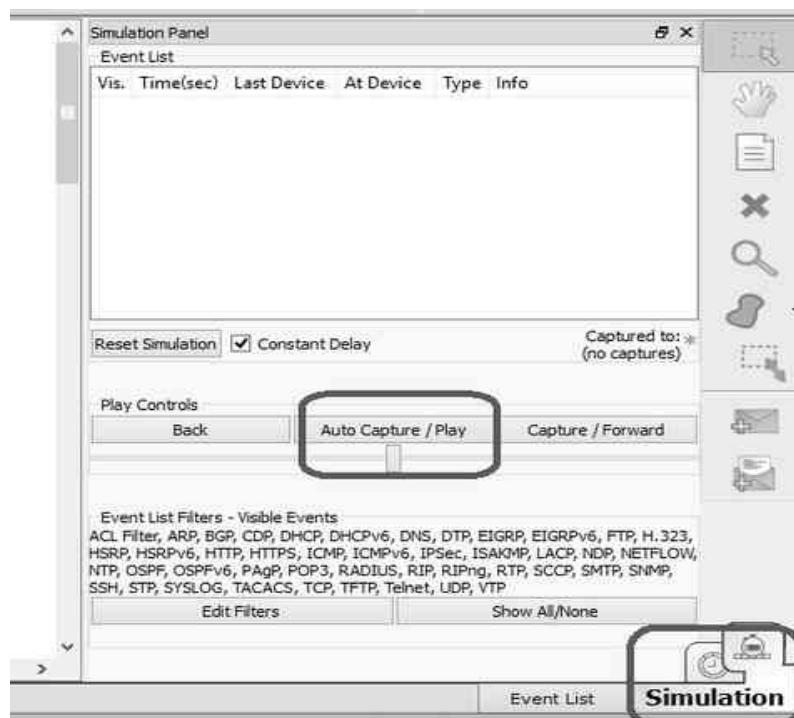
Step 2: Add a PC and a server to the workspace.

Step 3: Connect two devices with a copper crossover cable.

Step 4: Assign IP addresses. **PC:** 192.168.0.1/255.255.255.0 and **Server:** 192.168.0.2/255.255.255.0. Both of them are in the same subnet.

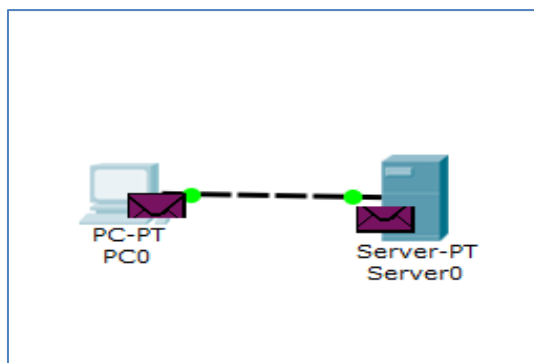


Step 5: Click on the **real time/simulation** tab and switch to the simulation mode.

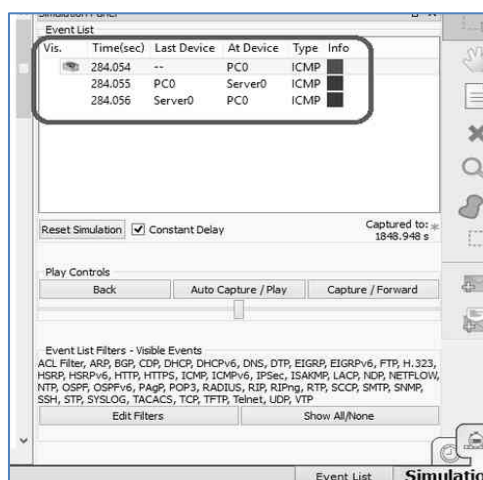


Step 6: Click on the **Auto Capture / Play** button. Packet capture begins.

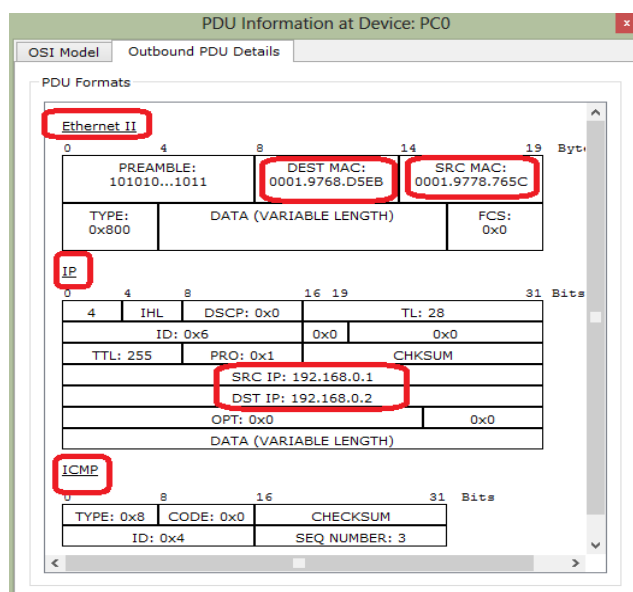
Step 7: From the **common tools** bar, click on the closed envelope icon. Click on the PC first and then on the server.



In the event list, you will observe three entries: creation of an ICMP packet, ICMP echo sent, and ICMP reply received.



To view a packet's TCP/IP layers information, click on a packet (the envelope icon). The **Outbound PDU Details** tab lists each layer's information.



The simulation mode has control buttons:

Back: Each time you click on it, the simulation moves one step back.

Auto Capture / Play: The network traffic will be continuously captured when it is pressed.

Capture/Forward: Each time, you want to move the packet from one place to another you have to press this button.

Topic 183: Connecting Devices and Link Status

This topic discusses how to connect devices and to check the link status in Packet Tracer.

In Packet Tracer, there are a number of cables available to connect devices. Click **Connections** icon from the **device-type selection box**. Now, you can view the various cables in **device-specific selection box**.

Automatically choose connection type: If you have no idea which cable to use, this option automatically allows two devices to get connected with the best cable.

Console: The console port of a network device can be connected to the RS-232 port on a PC/laptop. This allows you to view the network device's console from a PC/laptop.

Copper straight-through: It is a standard Ethernet cable that connects devices operating in different layers of the OSI model. For example, hub to router, switch to PC.

Copper cross-over: This Ethernet cable connects devices such as hub to hub, PC to PC, PC to router, and PC to printer i.e. devices operating in the same OSI layer.

Fiber: Connects Fast Ethernet and Gigabit Ethernet ports of a fiber port.

Phone: RJ11 cable. It connects the analog phone to a VoIP phone. Also, modem interface of routers can be connected.

Serial DCE and DTE: Serial cables connect routers together. There is a clock symbol on the **Data Circuit-terminating Equipment (DCE)** end. If you have selected Serial **Data Terminal Equipment (DTE)**, the DTE end will be the first device connected with this cable and the DCE end will be the next. In case of the Serial DCE cable, just the opposite happens.

Link status: Once you have connected devices together, you will find a light, at each end of the cable. The light indicates the state of the connection.

- **Bright green:** When the physical link is up, it is indicated by bright green. Status of the line protocol is not indicated by it.
- **Blinking green:** Link activity is represented by it.
- **Red:** In case a physical link is down, it is indicated by red. This can be due to incorrect cables or by a port being administratively shut down.
- **Amber:** When the port of a switch is running the **Spanning Tree Protocol (STP)** algorithm to detect layer 2 loops, the light becomes amber.

Topic 184: Testing Connectivity with PDUs

In this topic, we test connectivity with Protocol Data Units (PDUs) in Packet Tracer.

When you have created a topology in Packet Tracer, then you would like to test connectivity between devices. Connectivity can be tested by: using either simple or complex Protocol Data Units (PDUs), or pinging devices from their command-line interface. For large topologies, the PDU option is quicker.

Simple PDU

The **Add Simple PDU** tool relies on **Internet Control Message Protocol (ICMP)**. Let's create a simple topology to demonstrate the working of simple PDU.

Step 1: Add a PC and a server to the workspace and connect them using a copper crossover cable.

Step 2: Click on PC and use IP configuration. Type: 192.168.0.1/255.255.255.0. Do the same for the server and type: 192.168.0.2/255.255.255.0.

Step 3: Go to the common tools bar, click on the closed **envelope icon** or simply press key **P**. Click the envelope symbol on PC first and then on the server. Then, look at the **User Created Packet** box.

Fire	Last Status	Source	Destination	Type	Color	Time (sec)	Periodic	Num.	Edit
	Successful	PC0	Server0	ICMP		0.000	N	0	(edit)

Complex PDU

We use the previous example to understand working of Complex PDU.

Step 1: Click on the open envelope icon or press **C**.

Step 2: Click on the PC. This opens the **Create Complex PDU** dialog box.

Select **application** as HTTP, fill the **Destination IP address** = 192.168.0.2, **Starting Source Port**, (let's say 12345), and **Time** = 0, and then click on the **Create PDU** button.

Step 3: Now, click on the server. Examine the **user-created packet box**.

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num
	Successful	PC0	192.168.0.2	TCP		0.000	N	0

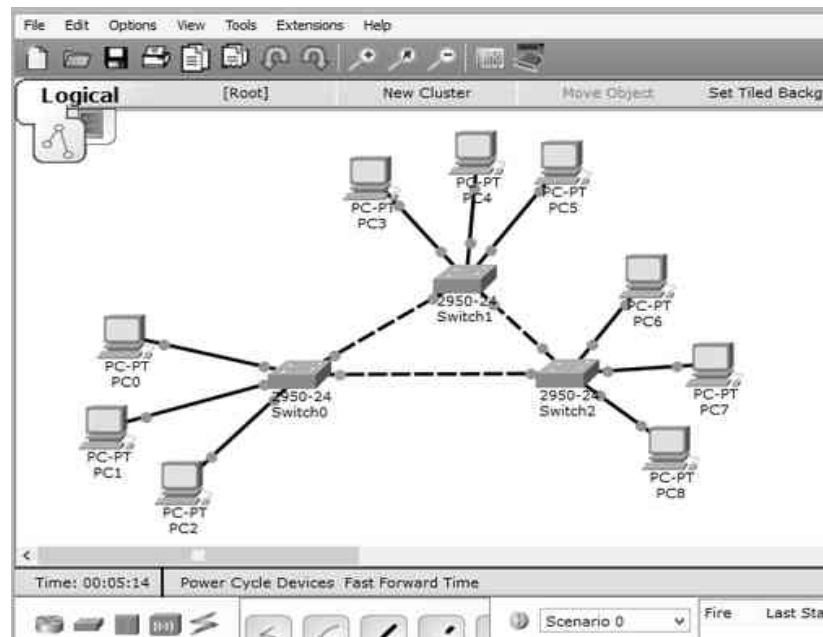
The entry indicates a successful TCP three-way handshake.

Topic 185: Clustering a Topology

This topic discusses how to cluster a topology in Packet Tracer.

When large topologies are created, understanding them becomes difficult. Clustering combines several devices that you choose into a single cloud icon. Upon double-clicking the cluster, it will get expanded and will display the devices normally.

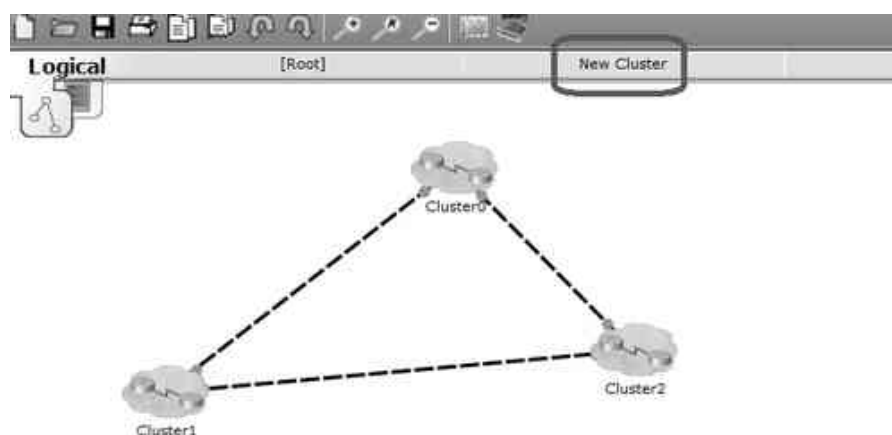
Step 1: Let's create a topology that consists of three switches and nine PCs. Select **End Devices** in **device-type selection box**. From the **Device-specific selection box**, drag-and-drop 9 PCs in the workspace. Select **switches** in **device-type selection box**. From the **Device-specific selection box**, drag-and-drop three 2950-24 switches in workspace. Select **Connections** in **device-type selection box**. From the **Device-specific selection box**, drag-and-drop Copper Straight-through cable in workspace. Connect a PC with a switch. Use copper cross-over cables to connect the switches.



Step 2: Combine PC0, PC1, PC2 and Switch0. Select PC0, PC1, PC2, and Switch0 by clicking on whitespace. Click on the **New Cluster** button on the top-right corner.

Step 3: Form a group of PC3, PC4, PC5 and Switch1. Click on whitespace next to PC3. Drag your mouse to select PC3, PC4, PC5, and Switch1. Click on the **New Cluster** button on the top-right corner.

Step 4: Repeat the same procedure done in **Steps 2 and 3** for combining PC6, PC7, PC8 and Switch2.



Double-clicking on a cluster expands it and displays only the devices within it.

Topic 186: Creating Cities, Offices & Wiring Closets

This topic discusses how to create cities, offices, and wire closets in Packet Tracer.

Packet Tracer can simulate the required environment logically and physically. In the physical workspace, it gives logical topology a physical dimension. Thus, making logical topologies more realistic. There are 4 environments available in the physical workspace:

- Intercity,
- City,
- Building, and
- Wiring closet.

1-Intercity: Being the largest environment, it consists of cities. You can create cities, buildings, and wiring closets in this layer.

2-Cities: Buildings and wiring closets are part of it. The default city name is **Home City**. You can drag and place cities anywhere on the intercity map.

3-Buildings: It contains only wiring closets. **Corporate Office** is default name.

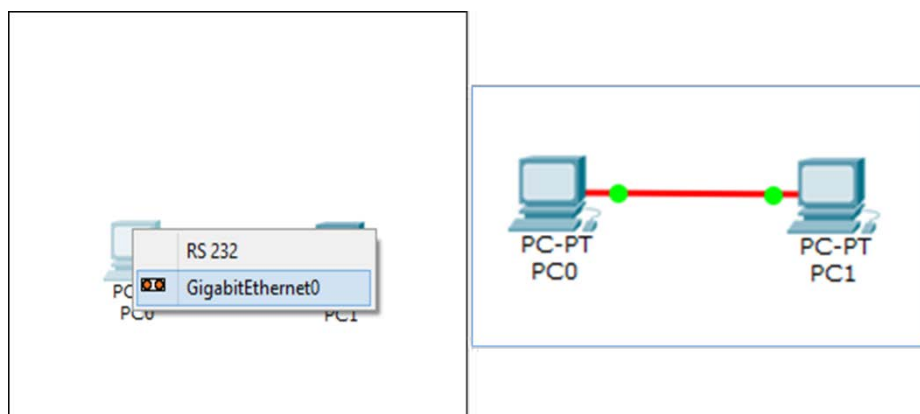
4-Wiring closet: This layer contains only devices. Default name is **Main Wiring Closet**.

Moving devices physically: Packet Tracer places all devices that are used in the logical workspace in Main Wiring Closet. Now we learn how to move them.

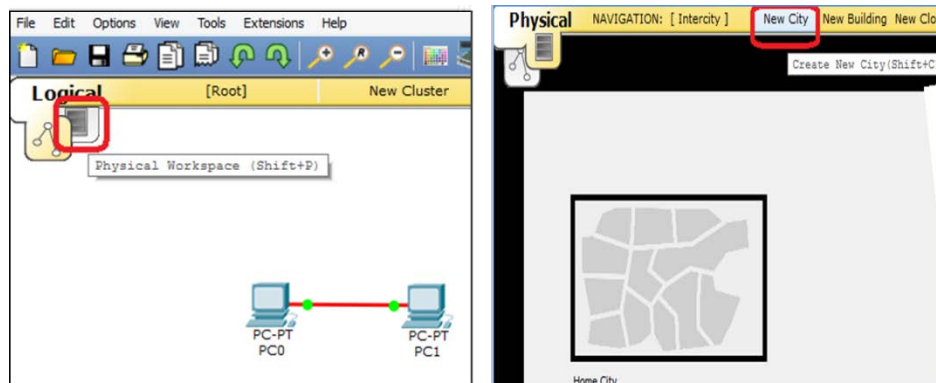
Step 1: Create a topology consisting of two PCs in the logical workspace.

Step 2: As Ethernet has distance restrictions, switch off both the PCs and replace their default modules with **PT-HOST-NM-1FGE**.

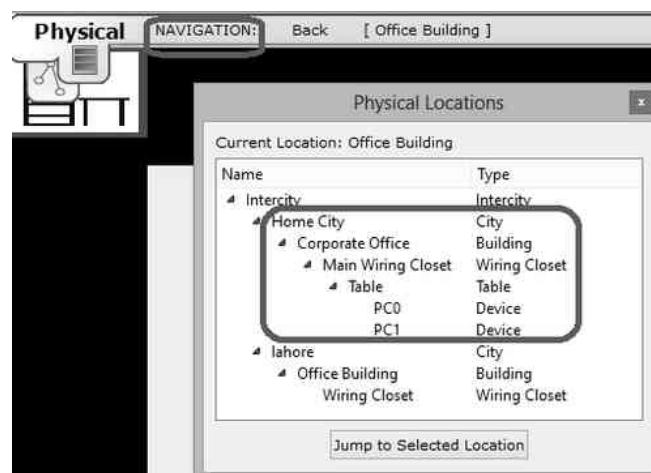
Step 3: Connect both of the PCs with a fiber cable and assign IP addresses.



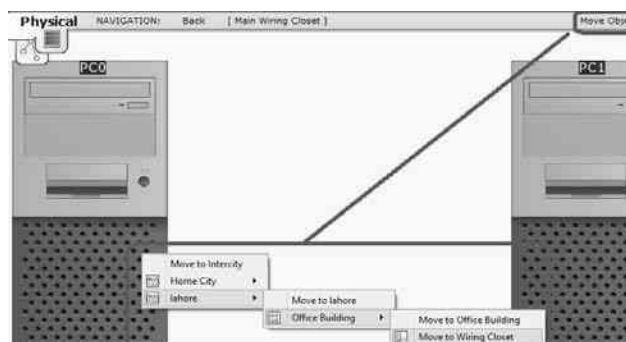
Step 4: Switch to the physical view, and click on the **New City** button. Rename it Lahore. Click on this city and create a new building, and then create a new wiring closet within this building.

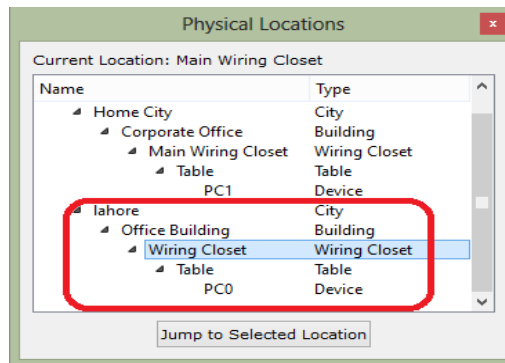


Step 5: Use the **NAVIGATION** button and go to **Home City | Corporate Office | Main Wiring Closet**. Both the PCs we inserted in the logical workspace are located here. Jump to their location.



Step 6: Use the **Move Object** button and move one of the PCs to **Lahore | Office Building | Wiring Closet**.

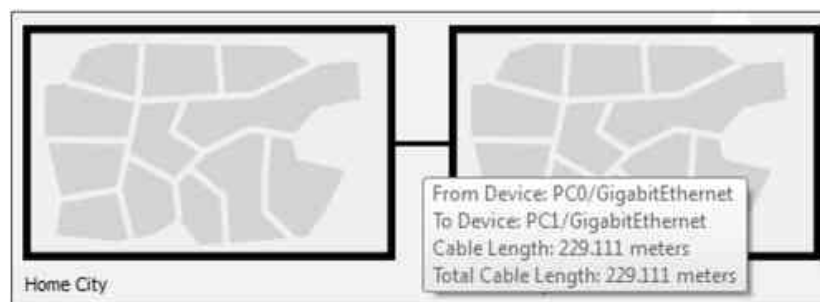




Topic 187: Managing Cables and Distances

In this topic, we discuss how to manage cables and distances in physical space of Packet Tracer.

Assume that we have created two cities in Packet Tracer. Each consists of a PC. Both PCs are connected via fiber. In physical view, we can measure a cable's distance by placing the pointer on the cable.



The length of Standard copper Ethernet cables can extend up to 100 meters. Let's test that the connection comes down due to the distance.

Step 1: Create a topology consisting of two PCs in the logical workspace.

Step 2: Connect both of the PCs with a copper cable and assign IP addresses.

Step 3: Switch to the physical view, and click on the **New City** button. Rename it Lahore. Click on this city and create a new building, and then create a new wiring closet within this building.

Step 4: Use the **NAVIGATION** button and go to **Home City | Corporate Office | Main Wiring Closet**. Both the PCs we inserted in the logical workspace are located here. Jump to their location.

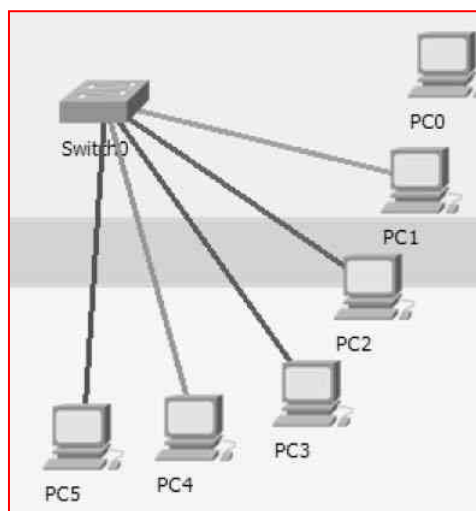
Step 5: Use the **Move Object** button and move one of the PCs to **Lahore | Office Building | Wiring Closet**.

Step 6: Check the distance between them. In case the distance is less than 100 meters, move them further apart, so that the distance becomes greater than 100 meters.

Step 7: When you go to the logical view, you will find that the connection comes down due to the distance as indicated by red color of the link status.

Cable manipulation

As number of devices increase, it becomes confusing to see which cable connects to what. This can be reduced by adding colors to cables. In the physical view, click on a wire and then, choose **Color Cable**. Pick a color from the Select Color dialog box.



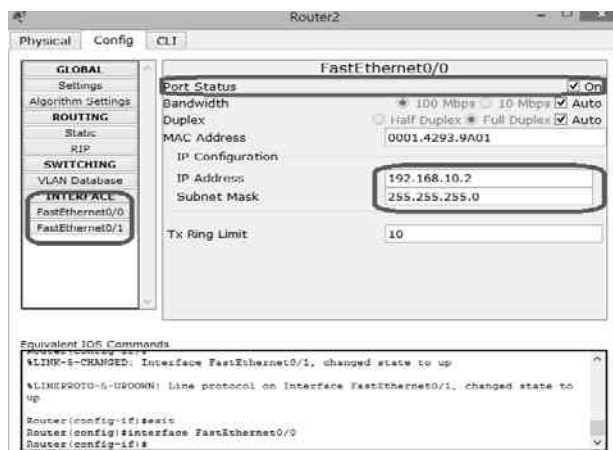
Topic 188: Static Routing with GUI

This topic configures static routing with GUI in Packet Tracer.

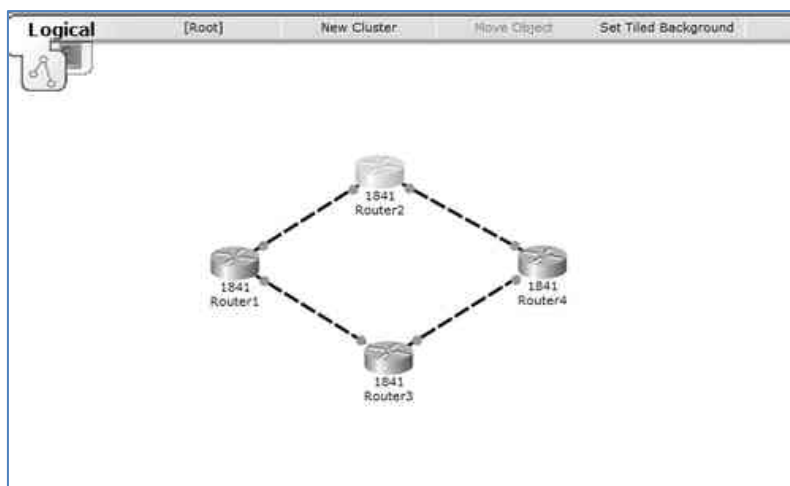
In **static routing algorithms**, routes change very slowly over time, often as a result of human intervention. A human manually edits a router's forwarding table. Let's learn the usage of the GUI feature with the help of a network consisting of four routers in a ring topology, with no PCs or loopback interfaces.

Step 1: Drag and drop 4 routers in the workspace.

Step 2: Click on a router icon, go to the **Config** tab, select an interface, and configure the IP address. Check the **On** checkbox. This will bring the port status up.



Router	Interface	IP Address
R1	FastEthernet0/0	192.168.10.1
	FastEthernet0/1	192.168.20.1
R2	FastEthernet0/0	192.168.10.2
	FastEthernet0/1	192.168.30.1
R3	FastEthernet0/0	192.168.20.2
	FastEthernet0/1	192.168.40.1
R4	FastEthernet0/0	192.168.30.2
	FastEthernet0/1	192.168.40.2



Step 3: Now go to the **ROUTING** section, and click on **Static**. The settings that we use for configuring static routing using the GUI are shown next:

Device	Network/Mask	Next Hop
R1	192.168.30.0 / 255.255.255.0	192.168.10.2
	192.168.40.0 / 255.255.255.0	192.168.20.2
R2	192.168.20.0 / 255.255.255.0	192.168.10.1
	192.168.40.0 / 255.255.255.0	192.168.30.2
R3	192.168.10.0 / 255.255.255.0	192.168.20.1
	192.168.30.0 / 255.255.255.0	192.168.40.2
R4	192.168.10.0 / 255.255.255.0	192.168.30.1
	192.168.20.0 / 255.255.255.0	192.168.40.1

Step 4: Test the connectivity between all of the routers with the help of **simple PDU**. Click on a router first and then click on another router. Look at the **User Created Packet** box.

Fire	Last Status	Source	Destination	Type	Color	Time(sec)
	Successful	Router4	Router2	ICMP		0.000
	Successful	Router2	Router4	ICMP		0.000
	Successful	Router1	Router2	ICMP		0.000

Step 5: Let's view the routing table of a router. Go to the **Common tools bar**, click on the inspect icon. Select a router and click on it. Then select **Routing Table**.

Type	Network	Port	Next Hop IP	Metric
C	192.168.10.0/24	FastEthernet0/0	---	0/0
C	192.168.20.0/24	FastEthernet0/1	---	0/0
S	192.168.30.0/24	---	192.168.10.2	1/0
S	192.168.40.0/24	---	192.168.20.2	1/0

Topic 189: Static Routing with CLI

This topic configures static routing with CLI in Packet Tracer.

To learn the usage of the CLI feature, assume a network consisting of four routers in a ring topology, with no PCs or loopback interfaces.

Step 1: Drag and drop 4 routers in the workspace.

Step 2: Click on a router icon, go to the **CLI** tab. As the device boots up, then you will see the prompt.

Step 3: Assign IP addresses to R1's interfaces:

- **R1(config)#interface FastEthernet0/0**
- **R1(config-if)#ip address 192.168.10.1 255.255.255.0**
- **R1(config-if)#no shutdown**
- **R1(config-if)#exit**
- **R1(config)#interface FastEthernet0/1**
- **R1(config-if)#ip address 192.168.20.1 255.255.255.0**
- **R1(config-if)#no shutdown**
- **R1(config-if)#exit**

Repeat Step 3 for the remaining routers with the following configurations for their interfaces:

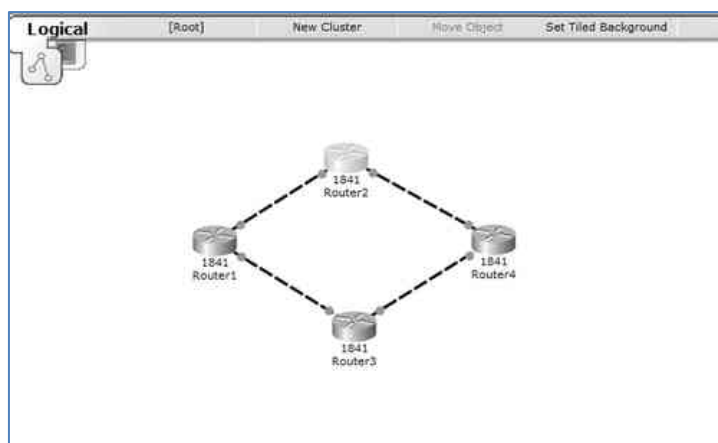
Router	Interface	IP Address
R1	FastEthernet0/0	192.168.10.1
	FastEthernet0/1	192.168.20.1
R2	FastEthernet0/0	192.168.10.2
	FastEthernet0/1	192.168.30.1
R3	FastEthernet0/0	192.168.20.2
	FastEthernet0/1	192.168.40.1
R4	FastEthernet0/0	192.168.30.2
	FastEthernet0/1	192.168.40.2

Step 4: Configure static routing:

- **R1(config)#ip route 192.168.30.0 255.255.255.0 192.168.10.2**
- **R1(config)#ip route 192.168.40.0 255.255.255.0 192.168.20.2**

Repeat **Step 4** for the remaining routers using the configuration shown next:

Device	Network/Mask	Next Hop
R1	192.168.30.0 / 255.255.255.0	192.168.10.2
	192.168.40.0 / 255.255.255.0	192.168.20.2
R2	192.168.20.0 / 255.255.255.0	192.168.10.1
	192.168.40.0 / 255.255.255.0	192.168.30.2
R3	192.168.10.0 / 255.255.255.0	192.168.20.1
	192.168.30.0 / 255.255.255.0	192.168.40.2
R4	192.168.10.0 / 255.255.255.0	192.168.30.1
	192.168.20.0 / 255.255.255.0	192.168.40.1



Step 5: Test the connectivity between all of the routers with the help of **simple PDU**. Click on a router first and then click on another router. Look at the **User Created Packet** box.

Fire	Last Status	Source	Destination	Type	Color	Time(sec)
	Successful	Router4	Router2	ICMP		0.000
	Successful	Router2	Router4	ICMP		0.000
	Successful	Router1	Router2	ICMP		0.000

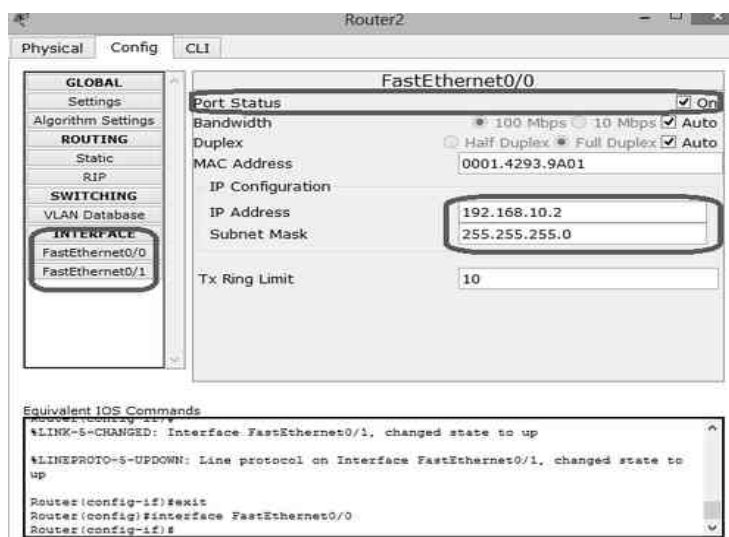
Topic 190: Configuring RIP with GUI

This topic configures RIP with GUI in Packet Tracer.

Dynamic Routing Protocols: A) Form "neighbor ship" with other routers. B) Send them the directly-connected routes and other received routes. Updates are also sent out by routers, when a topology change occurs. A GUI to configure a dynamic routing protocol called **Routing Information Protocol (RIP)** is available in Packet Tracer. Let's learn the usage of the GUI feature with the help of a network consisting of four routers in a ring topology, with no PCs or loopback interfaces.

Step 1: Drag and drop 4 routers in the workspace.

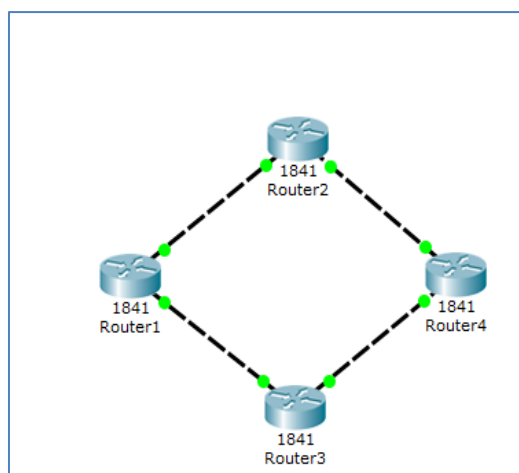
Step 2: Click on a router icon, go to the **Config** tab, select an interface, and configure the IP address.



Router	Interface	IP Address
R1	FastEthernet0/0	192.168.10.1
	FastEthernet0/1	192.168.20.1
R2	FastEthernet0/0	192.168.10.2
	FastEthernet0/1	192.168.30.1
R3	FastEthernet0/0	192.168.20.2
	FastEthernet0/1	192.168.40.1
R4	FastEthernet0/0	192.168.30.2
	FastEthernet0/1	192.168.40.2

Step 3: Click on **RIP**. Enter **Network** IP of its own interfaces.

Device	RIP Network
R1	192.168.10.0
	192.168.20.0
R2	192.168.10.0
	192.168.30.0
R3	192.168.20.0
	192.168.40.0
R4	192.168.30.0
	192.168.40.0



Step 4: Once the topology is configured, use the simple PDU to check for connectivity.

Fire	Last Status	Source	Destination	Type	Color	Time(sec)
	Successful	Router1	Router4	ICMP		0.000
	Successful	Router2	Router3	ICMP		0.000

Step 5: Use the delete tool and remove one link. Let's say we remove link between **R1** and **R2**. Use the simulation mode and test connectivity with the simple PDU. The packet takes the alternate, longer route and succeeds in reaching the destination.

Topic 191: Configuring RIP with CLI

In this topic, we configure RIP with CLI in Packet Tracer.

Let's assume a network consisting of four routers in a ring topology, with no PCs or loopback interfaces.

Step 1: Drag and drop 4 routers in the workspace.

Step 2: Click on a router icon, go to the **CLI** tab. As the device boots up, then you will see the prompt.

Step 3: Assign IP addresses to R1's interfaces:

- **R1(config)#interface FastEthernet0/0**
- **R1(config-if)#ip address 192.168.10.1 255.255.255.0**

- R1(config-if)#no shutdown
- R1(config-if)#exit
- R1(config)#interface FastEthernet0/1
- R1(config-if)#ip address 192.168.20.1 255.255.255.0
- R1(config-if)#no shutdown
- R1(config-if)#exit

Repeat Step 3 for the remaining routers with the following configurations for their interfaces:

Router	Interface	IP Address
R1	FastEthernet0/0	192.168.10.1
	FastEthernet0/1	192.168.20.1
R2	FastEthernet0/0	192.168.10.2
	FastEthernet0/1	192.168.30.1
R3	FastEthernet0/0	192.168.20.2
	FastEthernet0/1	192.168.40.1
R4	FastEthernet0/0	192.168.30.2
	FastEthernet0/1	192.168.40.2

Step 4: Enter into the config mode of RIP:

- R1(config)#router rip

Step 5: Enter the network IP addresses:

- R1(config-router)#network 192.168.10.0
- R1(config-router)#network 192.168.20.0

Step 6: Test the connectivity between all of the routers with the help of **simple PDU**. Click on a router first and then click on another router. Look at the **User Created Packet** box.

Fire	Last Status	Source	Destination	Type	Color	Time(sec)
	Successful	Router1	Router4	ICMP		0.000
	Successful	Router2	Router3	ICMP		0.000

Topic 192: Load Sharing

In this topic, we look at how load sharing works in Packet Tracer.

When a source router has multiple paths to a target path, then it can load balance traffic across them. If multiple paths exist from a source to a network destination with the same metric, then RIP automatically balances the traffic between them. Assume a network that consists of four routers in a ring topology with no PCs.

Step 1: Drag and drop 4 routers in the workspace.

Step 2: Configure the IP address.

Router	Interface	IP Address
R1	FastEthernet0/0	192.168.10.1
	FastEthernet0/1	192.168.20.1
R2	FastEthernet0/0	192.168.10.2
	FastEthernet0/1	192.168.30.1
R3	FastEthernet0/0	192.168.20.2
	FastEthernet0/1	192.168.40.1
R4	FastEthernet0/0	192.168.30.2
	FastEthernet0/1	192.168.40.2

Step 3: Configure **Network** IP of its own interfaces.

Device	RIP Network
R1	192.168.10.0
	192.168.20.0
R2	192.168.10.0
	192.168.30.0
R3	192.168.20.0
	192.168.40.0
R4	192.168.30.0
	192.168.40.0

Step 4: On **Router 4**, let's add a loopback interface (a virtual interface that works like a real interface and needs IP address). Go to the **CLI** tab of **R4** and enter:

- **R4(config)#interface loopback 0**

- **R4(config-if)#ip address 192.168.100.0 255.255.255.0**

Step 5: Go to the RIP config mode and enter the network IP for this loopback interface.

- **R4(config)#router rip**
- **R4(router-if)#network 192.168.100.0 .**

Step 6: Create a complex PDU that is sent every two seconds.

Source Settings

Source Device: Router4

Outgoing Port: FastEthernet0/0 ☒ Auto Select Port

PDU Settings

Select Application: PING

Destination IP Address: 192.168.100.1

TTL: 32

TOS: 0

Sequence Number: 0

Size: 0

Simulation Settings

☐ One Shot Time: Seconds

☒ Periodic Interval: 2 Seconds

Create PDU

Step 7: Turn on the simulation mode. You will find that the first packet takes the **R1-R2-R4** route while the second takes the **R1-R3-R4** route.