WireShark in-depth Tutorial

Wireshark is an open-source and free network traffic inspection tool. It captures and displays packets in real-time for offline analysis in a human-readable format with microscopic details. It requires some sound knowledge of basic networking and is considered an essential tool for system administrators and network security experts.

Wireshark is the de-facto go-to tool for several network problems that vary from network troubleshooting, security issue examination, inspecting network traffic of a suspicious application, debugging protocol implementations, along with network protocol learning purposes, etc.

The Wireshark project was initiated in 1998. Thanks to the global networking expert's voluntary contribution, it continues to make updates for new technologies and encryption standards. Hence, it's by far one of the best packet analyzer tools and is utilized as a standard commercial tool by various government agencies, educational institutes, and non-profit organizations.

The Wireshark tool is composed of a rich set of features. Some of them are the following:

- Multiplatform: it is available for Unix, Mac, and Window systems.
- It captures packets from various network media, i.e., Wireless LAN, Ethernet, USB, Bluetooth, etc.
- It opens packet files captured by other programs such as Oracle snoop and atmsnoop, Nmap, tcpdump, Microsoft Network Monitor, SNORT, and many others.
- It saves and exports captured packet data in various formats (CSV, XML, plaintext, etc.).
- It provides description support for protocols including SSL, WPA/WPA2, IPsec, and many others.
- It includes capture and display filters.

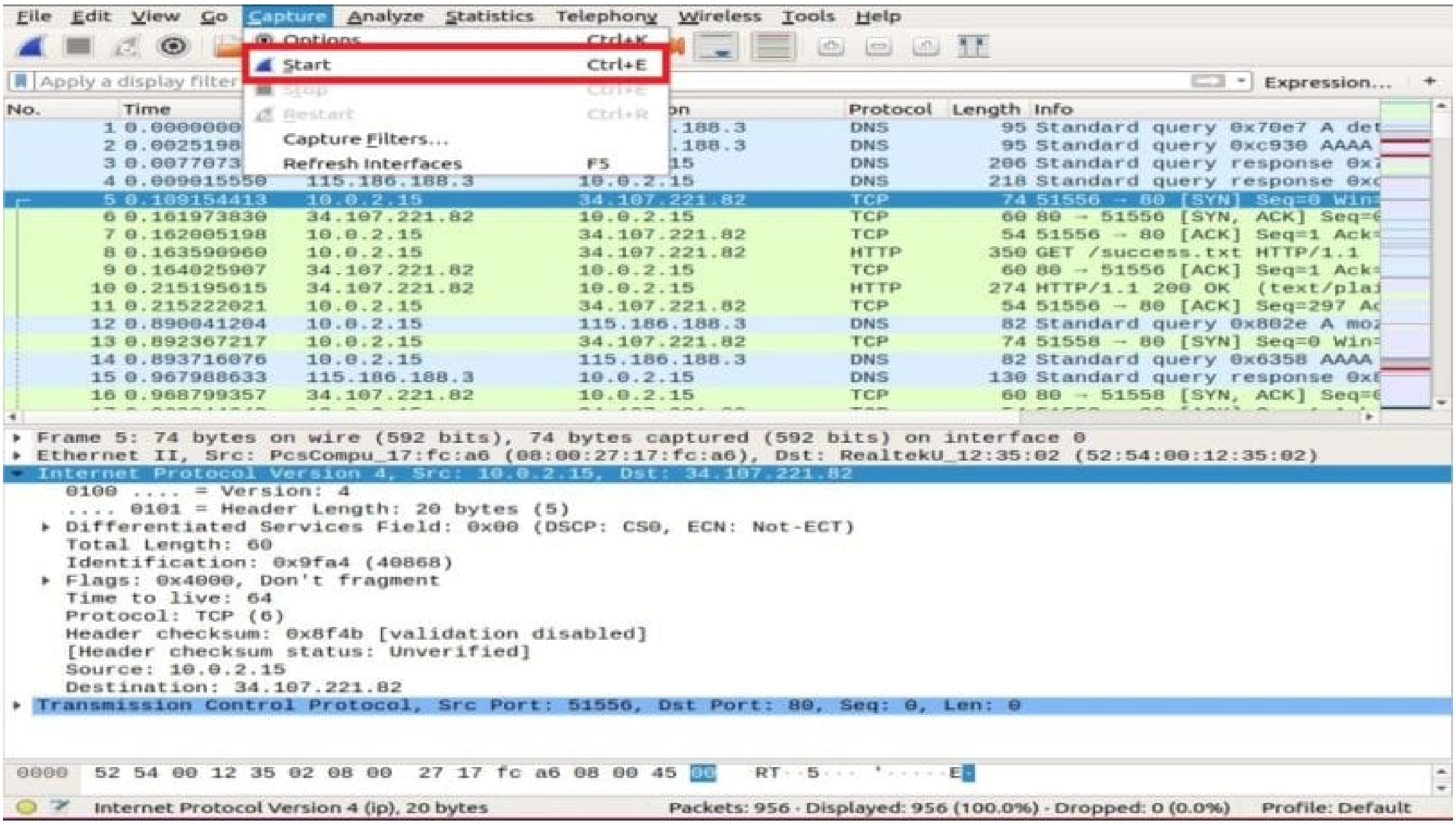
However, Wireshark won’t warn you of any malicious activity. It will only help you inspect and identify what is happening on your network. Moreover, it will only analyze network protocol/activities and won’t perform any other activity like sending/intercepting packets.

This article provides an in-depth tutorial that begins with the basics (i.e., filtering, Wireshark network layers, etc.) and takes you into the depth of traffic analysis.

Wireshark Filters

Wireshark comes with powerful filter engines, Capture Filters and Display Filters, to remove noise from the network or already captured traffic. These filters narrow down the unrequired traffic and display only the packets that you want to see. This feature helps network administrators to troubleshoot the problems at hand.

Before going into the details of filters. In case you are wondering how to capture the network traffic without any filter, you can either press Ctrl+E or go to the Capture option on the Wireshark interface and click Start.



```
No.   Time                                    on           Protocol Length Info
    1 0.0000000                                 .188.3      DNS         95 Standard query 0x70e7 A det
    2 0.0025198                                 .188.3      DNS         95 Standard query 0xc930 AAAA
    3 0.0077073                                 15          DNS        206 Standard query response 0x7
    4 0.009015550  115.186.188.3               10.0.2.15     DNS        218 Standard query response 0xc
    5 0.109154413  10.0.2.15                   34.107.221.82 TCP         74 51556 → 80 [SYN] Seq=0 Win=
    6 0.161973830  34.107.221.82               10.0.2.15     TCP         60 80 → 51556 [SYN, ACK] Seq=0
    7 0.162005198  10.0.2.15                   34.107.221.82 TCP         54 51556 → 80 [ACK] Seq=1 Ack=
    8 0.163590960  10.0.2.15                   34.107.221.82 HTTP       350 GET /success.txt HTTP/1.1
    9 0.164025907  34.107.221.82               10.0.2.15     TCP         60 80 → 51556 [ACK] Seq=1 Ack=
   10 0.215195615  34.107.221.82               10.0.2.15     HTTP       274 HTTP/1.1 200 OK  (text/plai
   11 0.215222021  10.0.2.15                   34.107.221.82 TCP         54 51556 → 80 [ACK] Seq=297 Ac
   12 0.890041204  10.0.2.15                   115.186.188.3 DNS         82 Standard query 0x802e A moz
   13 0.892367217  10.0.2.15                   34.107.221.82 TCP         74 51558 → 80 [SYN] Seq=0 Win=
   14 0.893716076  10.0.2.15                   115.186.188.3 DNS         82 Standard query 0x6358 AAAA
   15 0.967988633  115.186.188.3               10.0.2.15     DNS        130 Standard query response 0xE
   16 0.968799357  34.107.221.82               10.0.2.15     TCP         60 80 → 51558 [SYN, ACK] Seq=0
```

```
▸ Frame 5: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
▸ Ethernet II, Src: PcsCompu_17:fc:a6 (08:00:27:17:fc:a6), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
▾ Internet Protocol Version 4, Src: 10.0.2.15, Dst: 34.107.221.82
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 60
    Identification: 0x9fa4 (40868)
  ▸ Flags: 0x4000, Don't fragment
    Time to live: 64
    Protocol: TCP (6)
    Header checksum: 0x8f4b [validation disabled]
    [Header checksum status: Unverified]
    Source: 10.0.2.15
    Destination: 34.107.221.82
▸ Transmission Control Protocol, Src Port: 51556, Dst Port: 80, Seq: 0, Len: 0

0000  52 54 00 12 35 02 08 00  27 17 fc a6 08 00 45 00   RT··5···'·····E·

Internet Protocol Version 4 (ip), 20 bytes      Packets: 956 · Displayed: 956 (100.0%) · Dropped: 0 (0.0%)      Profile: Default
```

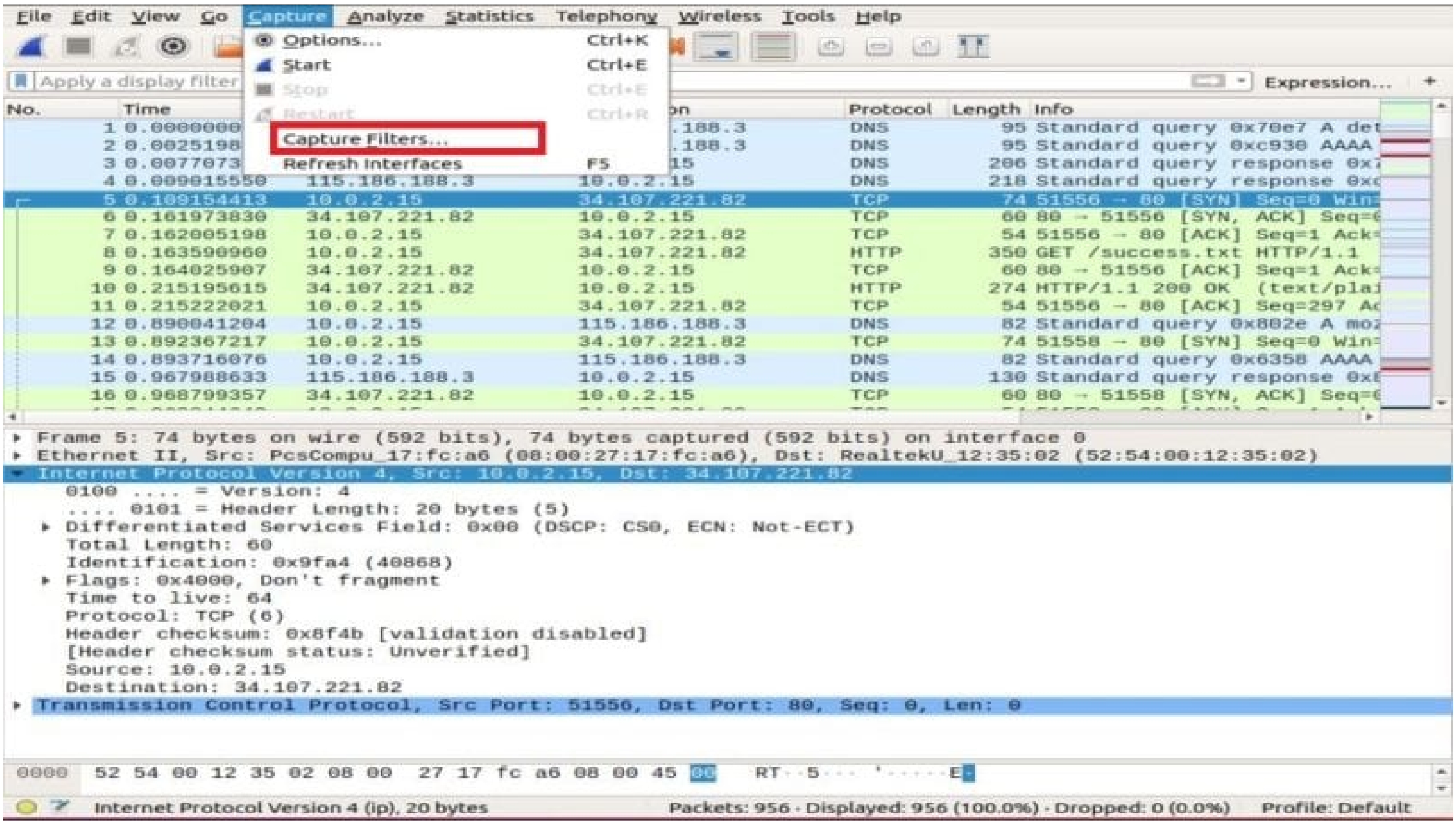
Now, let’s dig deep into the available filters.

Capture Filter

Wireshark provides support in reducing the size of a raw packet capture by allowing you to use a Capture Filter. But it only captures the packet traffic that matches the filter and disregards the rest of it. This feature helps you monitor and analyze the traffic of a specific application using the network.

Do not confuse this filter with display filters. **It's not a display filter**. This filter appears at the main window that is needed to set before starting packet capture. Moreover, you cannot modify this filter during the capture.

You can go to the **Capture** option of the interface and select **Capture Filters**.



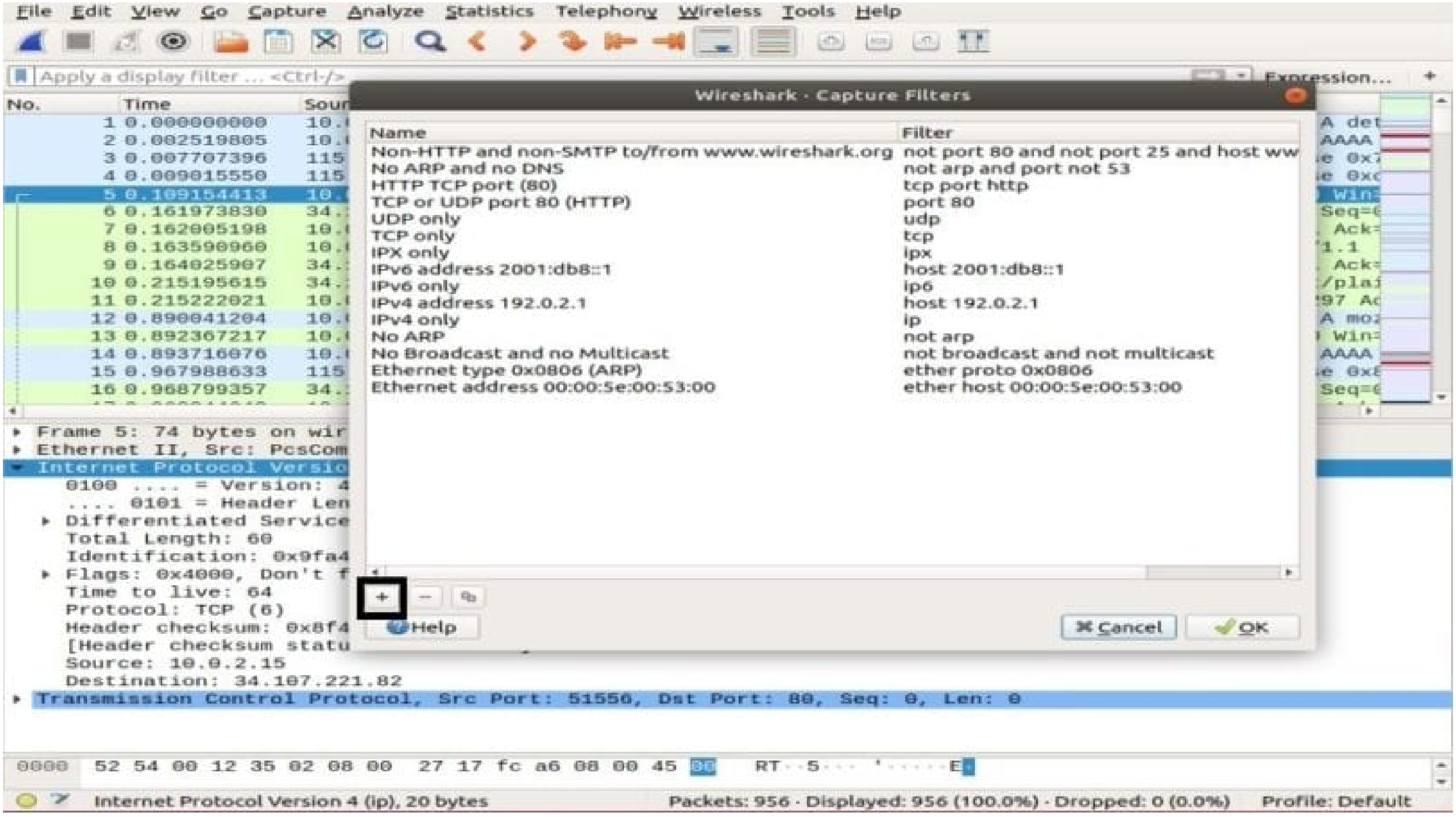
```
No.   Time                                                   Protocol Length Info
    1 0.0000000                          .188.3              DNS        95 Standard query 0x70e7 A det
    2 0.0025198                          .188.3              DNS        95 Standard query 0xc930 AAAA
    3 0.0077073                          15                  DNS       206 Standard query response 0x7
    4 0.009015550   115.186.188.3        10.0.2.15           DNS       218 Standard query response 0xc
    5 0.109154413   10.0.2.15            34.107.221.82       TCP        74 51556 → 80 [SYN] Seq=0 Win=
    6 0.161973830   34.107.221.82        10.0.2.15           TCP        60 80 → 51556 [SYN, ACK] Seq=0
    7 0.162005198   10.0.2.15            34.107.221.82       TCP        54 51556 → 80 [ACK] Seq=1 Ack=
    8 0.163590960   10.0.2.15            34.107.221.82       HTTP      350 GET /success.txt HTTP/1.1
    9 0.164025907   34.107.221.82        10.0.2.15           TCP        60 80 → 51556 [ACK] Seq=1 Ack=
   10 0.215195615   34.107.221.82        10.0.2.15           HTTP      274 HTTP/1.1 200 OK  (text/plai
   11 0.215222021   10.0.2.15            34.107.221.82       TCP        54 51556 → 80 [ACK] Seq=297 Ac
   12 0.890041204   10.0.2.15            115.186.188.3       DNS        82 Standard query 0x802e A moz
   13 0.892367217   10.0.2.15            34.107.221.82       TCP        74 51558 → 80 [SYN] Seq=0 Win=
   14 0.893716076   10.0.2.15            115.186.188.3       DNS        82 Standard query 0x6358 AAAA
   15 0.967988633   115.186.188.3        10.0.2.15           DNS       130 Standard query response 0xE
   16 0.968799357   34.107.221.82        10.0.2.15           TCP        60 80 → 51558 [SYN, ACK] Seq=0
```

```
▸ Frame 5: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
▸ Ethernet II, Src: PcsCompu_17:fc:a6 (08:00:27:17:fc:a6), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
▾ Internet Protocol Version 4, Src: 10.0.2.15, Dst: 34.107.221.82
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 60
    Identification: 0x9fa4 (40868)
  ▸ Flags: 0x4000, Don't fragment
    Time to live: 64
    Protocol: TCP (6)
    Header checksum: 0x8f4b [validation disabled]
    [Header checksum status: Unverified]
    Source: 10.0.2.15
    Destination: 34.107.221.82
▸ Transmission Control Protocol, Src Port: 51556, Dst Port: 80, Seq: 0, Len: 0
```

```
0000  52 54 00 12 35 02 08 00  27 17 fc a6 08 00 45 00   RT··5···'·····E·
```

```
Internet Protocol Version 4 (ip), 20 bytes          Packets: 956 · Displayed: 956 (100.0%) · Dropped: 0 (0.0%)     Profile: Default
```

You will be prompted with a window, as shown in the snapshot. You can choose any filter from the list of filters or add/create a new filter by clicking on the **+** button.



Wireshark · Capture Filters

Name	Filter
Non-HTTP and non-SMTP to/from www.wireshark.org	not port 80 and not port 25 and host ww
No ARP and no DNS	not arp and port not 53
HTTP TCP port (80)	tcp port http
TCP or UDP port 80 (HTTP)	port 80
UDP only	udp
TCP only	tcp
IPX only	ipx
IPv6 address 2001:db8::1	host 2001:db8::1
IPv6 only	ip6
IPv4 address 192.0.2.1	host 192.0.2.1
IPv4 only	ip
No ARP	not arp
No Broadcast and no Multicast	not broadcast and not multicast
Ethernet type 0x0806 (ARP)	ether proto 0x0806
Ethernet address 00:00:5e:00:53:00	ether host 00:00:5e:00:53:00

```
No.   Time           Sour
    1 0.000000000    10.
    2 0.002519805    10.
    3 0.007707396    115
    4 0.009015550    115
    5 0.109154413    10.
    6 0.161973830    34.
    7 0.162005198    10.
    8 0.163590960    10.
    9 0.164025907    34.
   10 0.215195615    34.
   11 0.215222021    10.
   12 0.890041204    10.
   13 0.892367217    10.
   14 0.893716076    10.
   15 0.967988633    115
   16 0.968799357    34.
```

```
▸ Frame 5: 74 bytes on wir
▸ Ethernet II, Src: PcsCom
▾ Internet Protocol Versio
    0100 .... = Version: 4
    .... 0101 = Header Len
  ▸ Differentiated Service
    Total Length: 60
    Identification: 0x9fa4
  ▸ Flags: 0x4000, Don't f
    Time to live: 64
    Protocol: TCP (6)
    Header checksum: 0x8f4
    [Header checksum statu
    Source: 10.0.2.15
    Destination: 34.107.221.82
▸ Transmission Control Protocol, Src Port: 51556, Dst Port: 80, Seq: 0, Len: 0
```

```
0000  52 54 00 12 35 02 08 00  27 17 fc a6 08 00 45 00   RT··5···  '·····E·
```

Internet Protocol Version 4 (ip), 20 bytes   Packets: 956 · Displayed: 956 (100.0%) · Dropped: 0 (0.0%)   Profile: Default

Examples of the list of helpful Capture Filters:

- **host ip_address** – captures traffic, only between the specific communicating IP address
- **net 192.168.0.0/24** – captures traffic between IP address ranges/CIDRs
- **port 53** – captures DNS traffic
- **tcp portrange 2051-3502** – captures TCP traffic from port range 2051-3502
- **port not 22 and not 21** – capture all the traffic except SSH and FTP

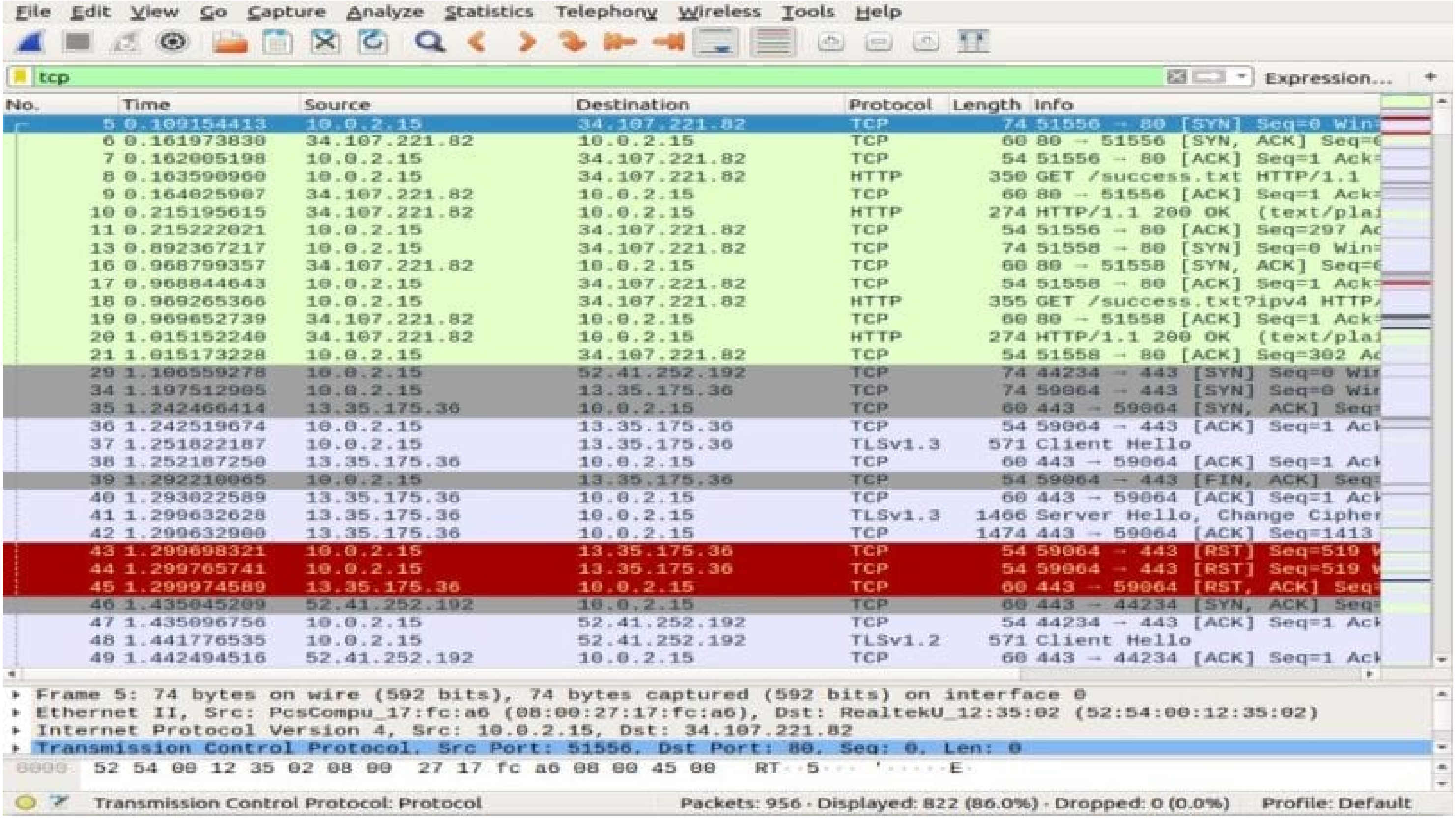
Display Filter

Display filters allow you to hide some packets from the already captured network traffic. These filters can be added above the captured list and can be modified on the fly. You can now control and narrow down the packets you want to concentrate on while hiding the unnecessary packets.

You can add filters in the display filter toolbar right above the first pane containing packet information. This filter can be used to display packets based on protocol, source IP address, destination IP address, ports, value and information of fields, comparison between fields, and a lot more.

That’s right! You can build a combination of filters using logical operators like ==.!=,||,&&, etc.

Some examples of display filters of a single TCP protocol and a combination filter are shown below:


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
tcp
Expression…

No.	Time	Source	Destination	Protocol	Length	Info
5	0.109154413	10.0.2.15	34.107.221.82	TCP	74	51556 → 80 [SYN] Seq=0 Win=
6	0.161973830	34.107.221.82	10.0.2.15	TCP	60	80 → 51556 [SYN, ACK] Seq=(
7	0.162005198	10.0.2.15	34.107.221.82	TCP	54	51556 → 80 [ACK] Seq=1 Ack=
8	0.163590960	10.0.2.15	34.107.221.82	HTTP	350	GET /success.txt HTTP/1.1
9	0.164025907	34.107.221.82	10.0.2.15	TCP	60	80 → 51556 [ACK] Seq=1 Ack=
10	0.215195615	34.107.221.82	10.0.2.15	HTTP	274	HTTP/1.1 200 OK (text/plai
11	0.215222021	10.0.2.15	34.107.221.82	TCP	54	51556 → 80 [ACK] Seq=297 Ac
13	0.892367217	10.0.2.15	34.107.221.82	TCP	74	51558 → 80 [SYN] Seq=0 Win=
16	0.968799357	34.107.221.82	10.0.2.15	TCP	60	80 → 51558 [SYN, ACK] Seq=(
17	0.968844643	10.0.2.15	34.107.221.82	TCP	54	51558 → 80 [ACK] Seq=1 Ack=
18	0.969265366	10.0.2.15	34.107.221.82	HTTP	355	GET /success.txt?ipv4 HTTP/
19	0.969652739	34.107.221.82	10.0.2.15	TCP	60	80 → 51558 [ACK] Seq=1 Ack=
20	1.015152240	34.107.221.82	10.0.2.15	HTTP	274	HTTP/1.1 200 OK (text/plai
21	1.015173228	10.0.2.15	34.107.221.82	TCP	54	51558 → 80 [ACK] Seq=302 Ac
29	1.106559278	10.0.2.15	52.41.252.192	TCP	74	44234 → 443 [SYN] Seq=0 Win
34	1.197512905	10.0.2.15	13.35.175.36	TCP	74	59064 → 443 [SYN] Seq=0 Win
35	1.242466414	13.35.175.36	10.0.2.15	TCP	60	443 → 59064 [SYN, ACK] Seq=
36	1.242519674	10.0.2.15	13.35.175.36	TCP	54	59064 → 443 [ACK] Seq=1 Ack
37	1.251822187	10.0.2.15	13.35.175.36	TLSv1.3	571	Client Hello
38	1.252187250	13.35.175.36	10.0.2.15	TCP	60	443 → 59064 [ACK] Seq=1 Ack
39	1.292210065	10.0.2.15	13.35.175.36	TCP	54	59064 → 443 [FIN, ACK] Seq=
40	1.293022589	13.35.175.36	10.0.2.15	TCP	60	443 → 59064 [ACK] Seq=1 Ack
41	1.299632628	13.35.175.36	10.0.2.15	TLSv1.3	1466	Server Hello, Change Cipher
42	1.299632900	13.35.175.36	10.0.2.15	TCP	1474	443 → 59064 [ACK] Seq=1413
43	1.299698321	10.0.2.15	13.35.175.36	TCP	54	59064 → 443 [RST] Seq=519 W
44	1.299765741	10.0.2.15	13.35.175.36	TCP	54	59064 → 443 [RST] Seq=519 W
45	1.299974589	13.35.175.36	10.0.2.15	TCP	60	443 → 59064 [RST, ACK] Seq=
46	1.435045209	52.41.252.192	10.0.2.15	TCP	60	443 → 44234 [SYN, ACK] Seq=
47	1.435096756	10.0.2.15	52.41.252.192	TCP	54	44234 → 443 [ACK] Seq=1 Ack
48	1.441776535	10.0.2.15	52.41.252.192	TLSv1.2	571	Client Hello
49	1.442494516	52.41.252.192	10.0.2.15	TCP	60	443 → 44234 [ACK] Seq=1 Ack

```
▸ Frame 5: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
▸ Ethernet II, Src: PcsCompu_17:fc:a6 (08:00:27:17:fc:a6), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
▸ Internet Protocol Version 4, Src: 10.0.2.15, Dst: 34.107.221.82
▸ Transmission Control Protocol, Src Port: 51556, Dst Port: 80, Seq: 0, Len: 0
0000  52 54 00 12 35 02 08 00  27 17 fc a6 08 00 45 00   RT··5···'·····E·
```

Transmission Control Protocol: Protocol   Packets: 956 · Displayed: 822 (86.0%) · Dropped: 0 (0.0%)   Profile: Default

Network Layers in Wireshark

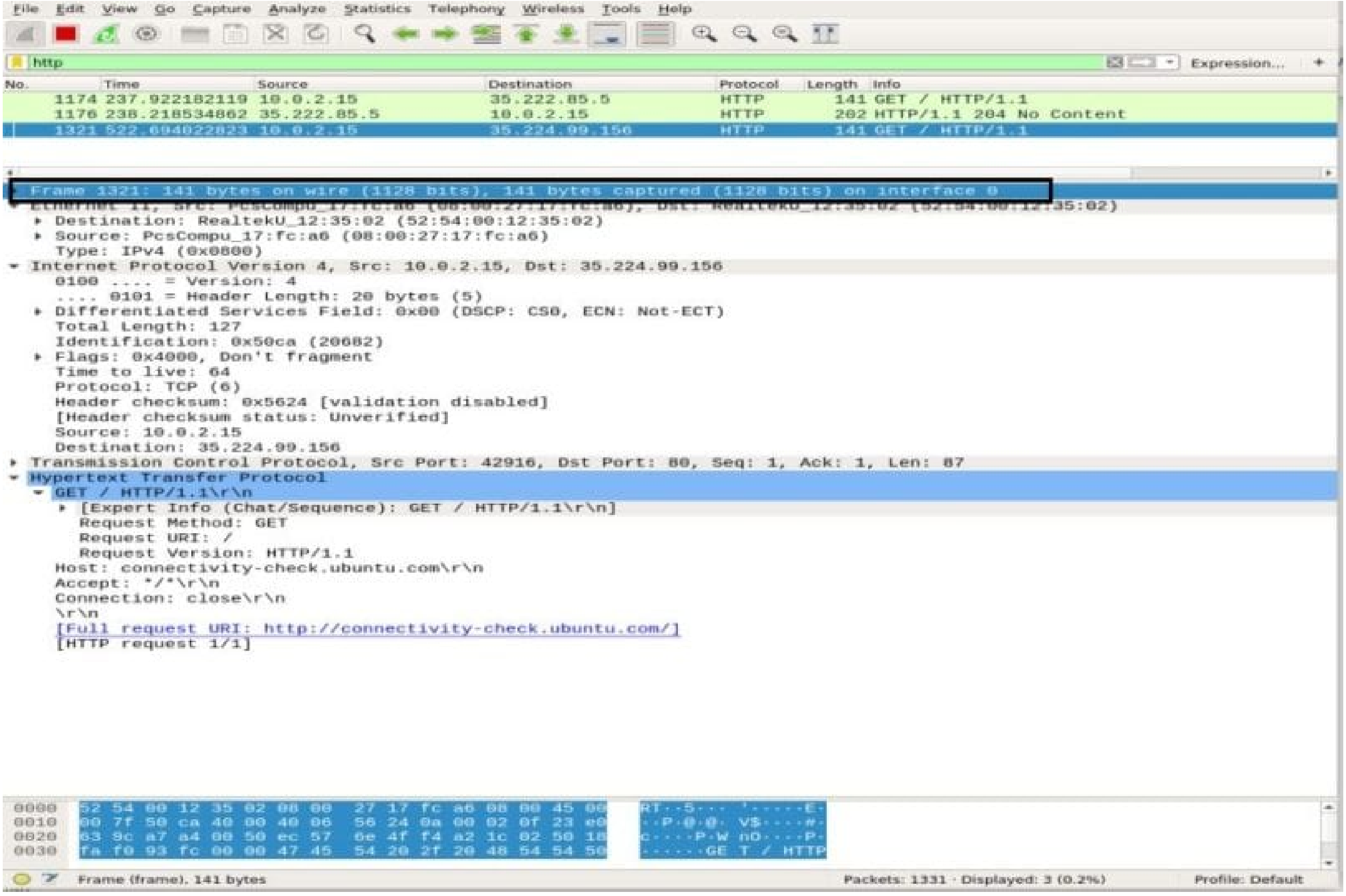
Other than packet inspection, Wireshark presents OSI layers that aids in the troubleshooting process. Wireshark shows the layers in reverse order, such as:

1. Physical Layer
2. Data Link Layer
3. Network Layer
4. Transport Layer
5. Application Layer

Note that Wireshark does not always show the Physical layer. We will now dig in each layer to understand the important aspect of packet analysis, and what each layer presents in Wireshark.

Physical Layer

The Physical layer, as shown in the following snapshot, presents the physical summary of the frame, such as hardware information. As a network administrator, you do not generally extract information from this layer.



```
http
```

```
No.     Time            Source          Destination      Protocol  Length Info
   1174 237.922182119 10.0.2.15       35.222.85.5      HTTP       141 GET / HTTP/1.1
   1176 238.218534862 35.222.85.5     10.0.2.15        HTTP       202 HTTP/1.1 204 No Content
   1321 522.694022823 10.0.2.15       35.224.99.156    HTTP       141 GET / HTTP/1.1
```

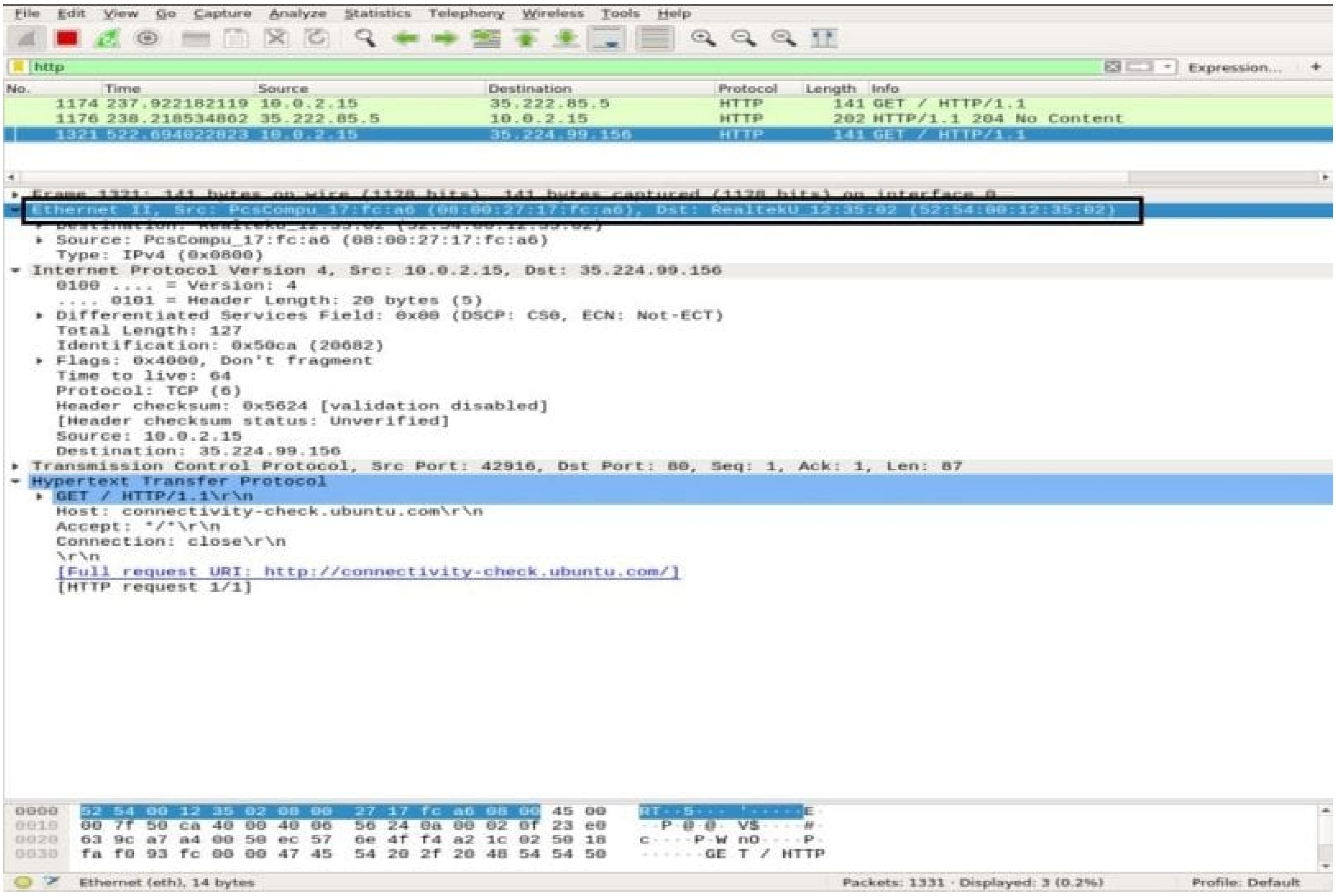
```
▸ Frame 1321: 141 bytes on wire (1128 bits), 141 bytes captured (1128 bits) on interface 0
▾ Ethernet II, Src: PcsCompu_17:fc:a6 (08:00:27:17:fc:a6), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
  ▸ Destination: RealtekU_12:35:02 (52:54:00:12:35:02)
  ▸ Source: PcsCompu_17:fc:a6 (08:00:27:17:fc:a6)
    Type: IPv4 (0x0800)
▾ Internet Protocol Version 4, Src: 10.0.2.15, Dst: 35.224.99.156
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 127
    Identification: 0x50ca (20682)
  ▸ Flags: 0x4000, Don't fragment
    Time to live: 64
    Protocol: TCP (6)
    Header checksum: 0x5624 [validation disabled]
    [Header checksum status: Unverified]
    Source: 10.0.2.15
    Destination: 35.224.99.156
▸ Transmission Control Protocol, Src Port: 42916, Dst Port: 80, Seq: 1, Ack: 1, Len: 87
▾ Hypertext Transfer Protocol
  ▾ GET / HTTP/1.1\r\n
    ▸ [Expert Info (Chat/Sequence): GET / HTTP/1.1\r\n]
      Request Method: GET
      Request URI: /
      Request Version: HTTP/1.1
    Host: connectivity-check.ubuntu.com\r\n
    Accept: */*\r\n
    Connection: close\r\n
    \r\n
    [Full request URI: http://connectivity-check.ubuntu.com/]
    [HTTP request 1/1]
```

```
0000  52 54 00 12 35 02 08 00  27 17 fc a6 08 00 45 00   RT··5··· '·····E·
0010  00 7f 50 ca 40 00 40 06  56 24 0a 00 02 0f 23 e0   ··P·@·@· V$····#·
0020  63 9c a7 a4 00 50 ec 57  6e 4f f4 a2 1c 02 50 18   c····P·W nO····P·
0030  fa f0 93 fc 00 00 47 45  54 20 2f 20 48 54 54 50   ······GE T / HTTP
```

Frame (frame), 141 bytes   Packets: 1331 · Displayed: 3 (0.2%)   Profile: Default

Data Link Layer

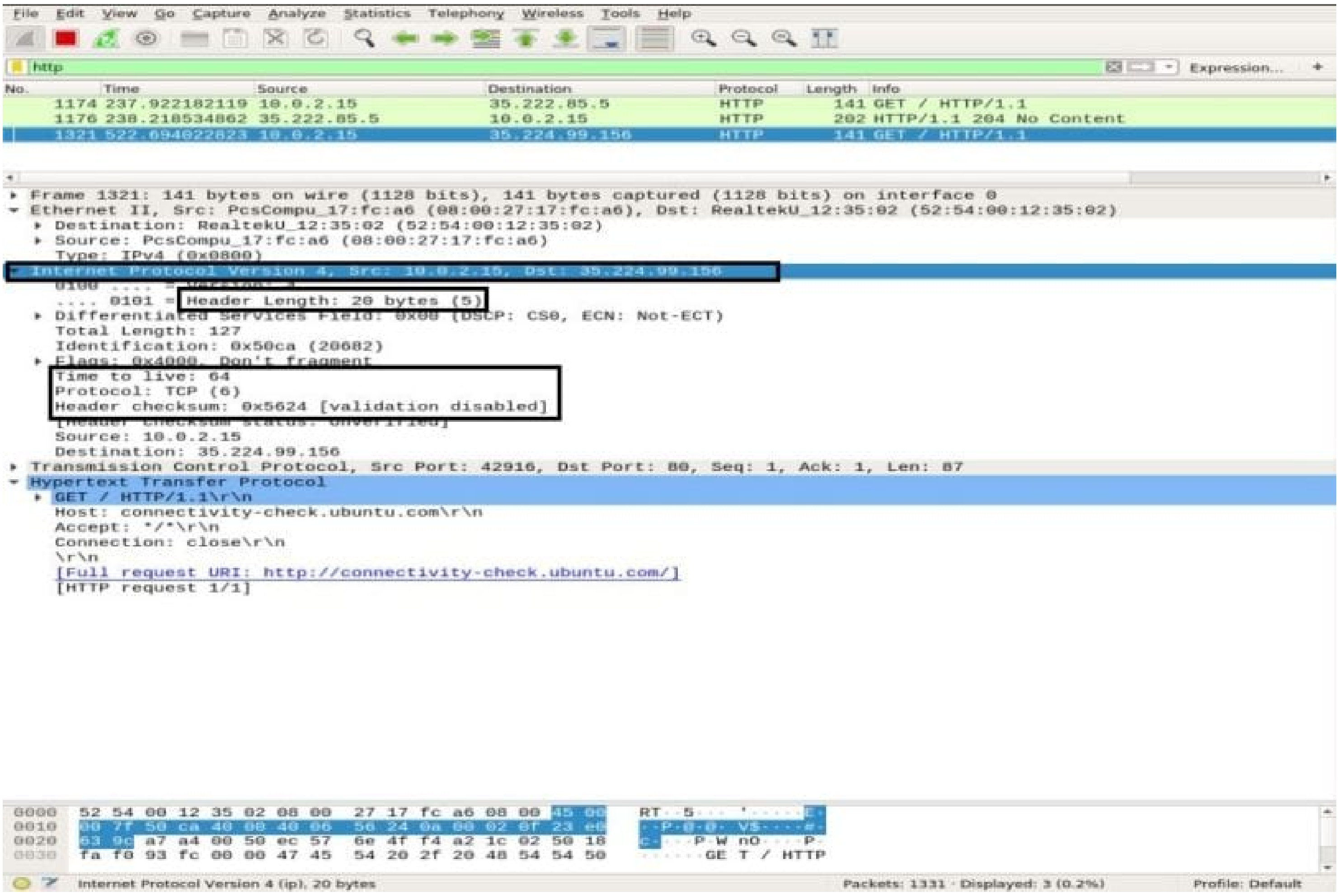
The next data link layer contains the source and destination network card address. It is relatively simple as it only delivers the frame from the laptop to the router or the next adjacent frame in the physical medium.


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
http
Expression…

```
No.    Time            Source         Destination     Protocol  Length Info
  1174 237.922182119 10.0.2.15       35.222.85.5     HTTP        141 GET / HTTP/1.1
  1176 238.218534862 35.222.85.5     10.0.2.15       HTTP        202 HTTP/1.1 204 No Content
  1321 522.694022823 10.0.2.15       35.224.99.156   HTTP        141 GET / HTTP/1.1

▸ Frame 1321: 141 bytes on wire (1128 bits), 141 bytes captured (1128 bits) on interface 0
▾ Ethernet II, Src: PcsCompu_17:fc:a6 (08:00:27:17:fc:a6), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
  ▸ Destination: RealtekU_12:35:02 (52:54:00:12:35:02)
  ▸ Source: PcsCompu_17:fc:a6 (08:00:27:17:fc:a6)
    Type: IPv4 (0x0800)
▾ Internet Protocol Version 4, Src: 10.0.2.15, Dst: 35.224.99.156
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 127
    Identification: 0x50ca (20682)
  ▸ Flags: 0x4000, Don't fragment
    Time to live: 64
    Protocol: TCP (6)
    Header checksum: 0x5624 [validation disabled]
    [Header checksum status: Unverified]
    Source: 10.0.2.15
    Destination: 35.224.99.156
▸ Transmission Control Protocol, Src Port: 42916, Dst Port: 80, Seq: 1, Ack: 1, Len: 87
▾ Hypertext Transfer Protocol
  ▸ GET / HTTP/1.1\r\n
    Host: connectivity-check.ubuntu.com\r\n
    Accept: */*\r\n
    Connection: close\r\n
    \r\n
    [Full request URI: http://connectivity-check.ubuntu.com/]
    [HTTP request 1/1]

0000  52 54 00 12 35 02 08 00  27 17 fc a6 08 00 45 00   RT··5··· '·····E·
0010  00 7f 50 ca 40 00 40 06  56 24 0a 00 02 0f 23 e0   ··P·@·@· V$····#·
0020  63 9c a7 a4 00 50 ec 57  6e 4f f4 a2 1c 02 50 18   c····P·W nO····P·
0030  fa f0 93 fc 00 00 47 45  54 20 2f 20 48 54 54 50   ······GE T / HTTP

Ethernet (eth), 14 bytes          Packets: 1331 · Displayed: 3 (0.2%)          Profile: Default
```

Network Layer

The network layer presents the source and destination IP addresses, IP version, header length, total packet length, and loads of other information.



```
No.     Time              Source         Destination      Protocol  Length Info
   1174 237.922182119 10.0.2.15          35.222.85.5      HTTP         141 GET / HTTP/1.1
   1176 238.218534862 35.222.85.5        10.0.2.15        HTTP         202 HTTP/1.1 204 No Content
   1321 522.694022823 10.0.2.15          35.224.99.156    HTTP         141 GET / HTTP/1.1
```

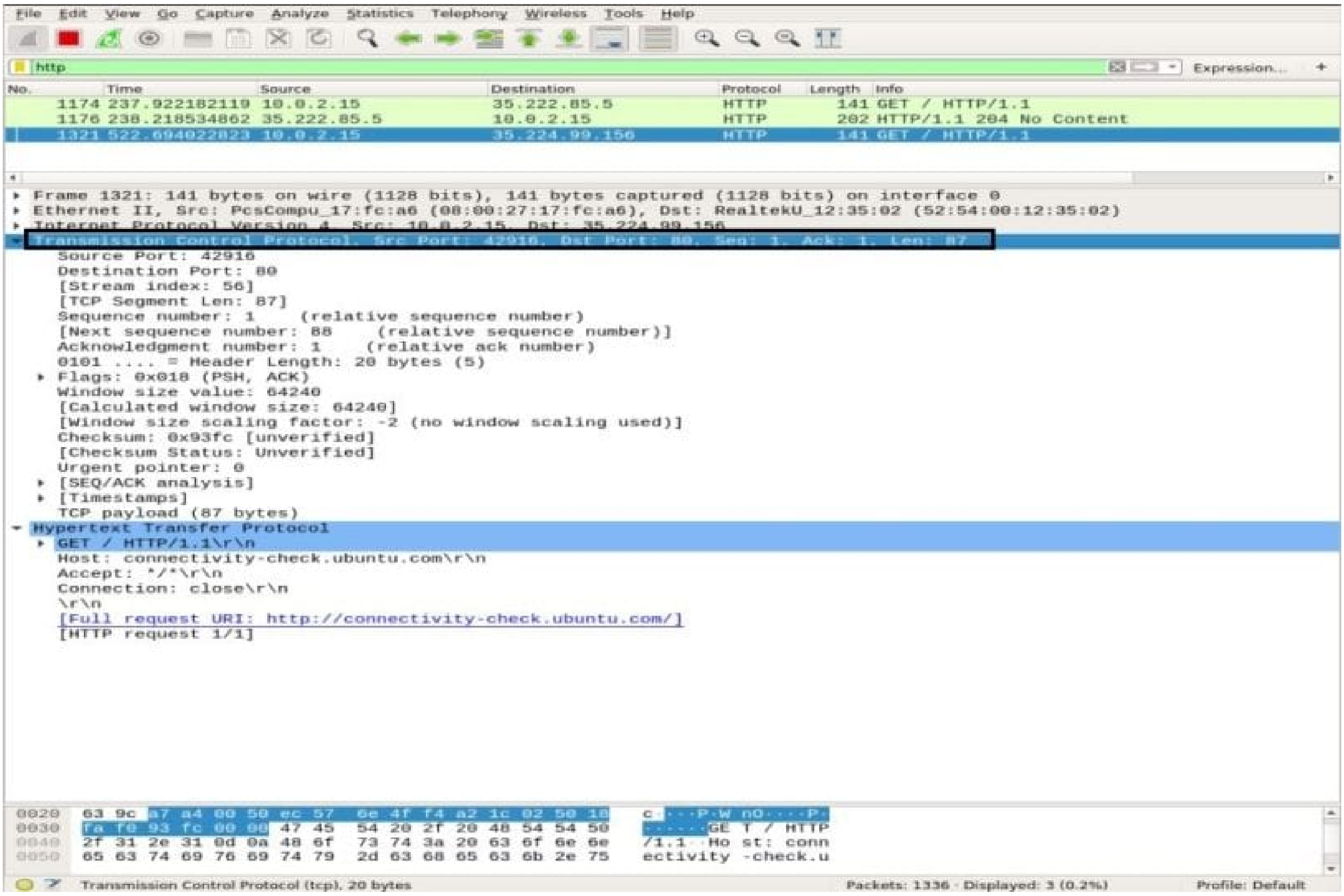
```
▸ Frame 1321: 141 bytes on wire (1128 bits), 141 bytes captured (1128 bits) on interface 0
▾ Ethernet II, Src: PcsCompu_17:fc:a6 (08:00:27:17:fc:a6), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
  ▸ Destination: RealtekU_12:35:02 (52:54:00:12:35:02)
  ▸ Source: PcsCompu_17:fc:a6 (08:00:27:17:fc:a6)
    Type: IPv4 (0x0800)
▾ Internet Protocol Version 4, Src: 10.0.2.15, Dst: 35.224.99.156
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 127
    Identification: 0x50ca (20682)
  ▸ Flags: 0x4000, Don't fragment
    Time to live: 64
    Protocol: TCP (6)
    Header checksum: 0x5624 [validation disabled]
    [Header checksum status: Unverified]
    Source: 10.0.2.15
    Destination: 35.224.99.156
▸ Transmission Control Protocol, Src Port: 42916, Dst Port: 80, Seq: 1, Ack: 1, Len: 87
▾ Hypertext Transfer Protocol
  ▸ GET / HTTP/1.1\r\n
    Host: connectivity-check.ubuntu.com\r\n
    Accept: */*\r\n
    Connection: close\r\n
    \r\n
    [Full request URI: http://connectivity-check.ubuntu.com/]
    [HTTP request 1/1]
```

```
0000  52 54 00 12 35 02 08 00  27 17 fc a6 08 00 45 00   RT··5···'·····E·
0010  00 7f 50 ca 40 00 40 06  56 24 0a 00 02 0f 23 e0   ··P·@·@· V$····#·
0020  63 9c a7 a4 00 50 ec 57  6e 4f f4 a2 1c 02 50 18   c····P·W nO····P·
0030  fa f0 93 fc 00 00 47 45  54 20 2f 20 48 54 54 50   ······GE T / HTTP
```

Internet Protocol Version 4 (ip), 20 bytes
Packets: 1331 · Displayed: 3 (0.2%)
Profile: Default

Transport Layer

In this layer, Wireshark displays information about the transport layer, which consists of the SRC port, DST port, header length, and sequence number that changes for each packet.



File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
http
Expression...

```
No.      Time            Source          Destination      Protocol  Length Info
    1174 237.922182119 10.0.2.15       35.222.85.5      HTTP         141 GET / HTTP/1.1
    1176 238.218534862 35.222.85.5     10.0.2.15        HTTP         202 HTTP/1.1 204 No Content
    1321 522.694022823 10.0.2.15       35.224.99.156    HTTP         141 GET / HTTP/1.1
```

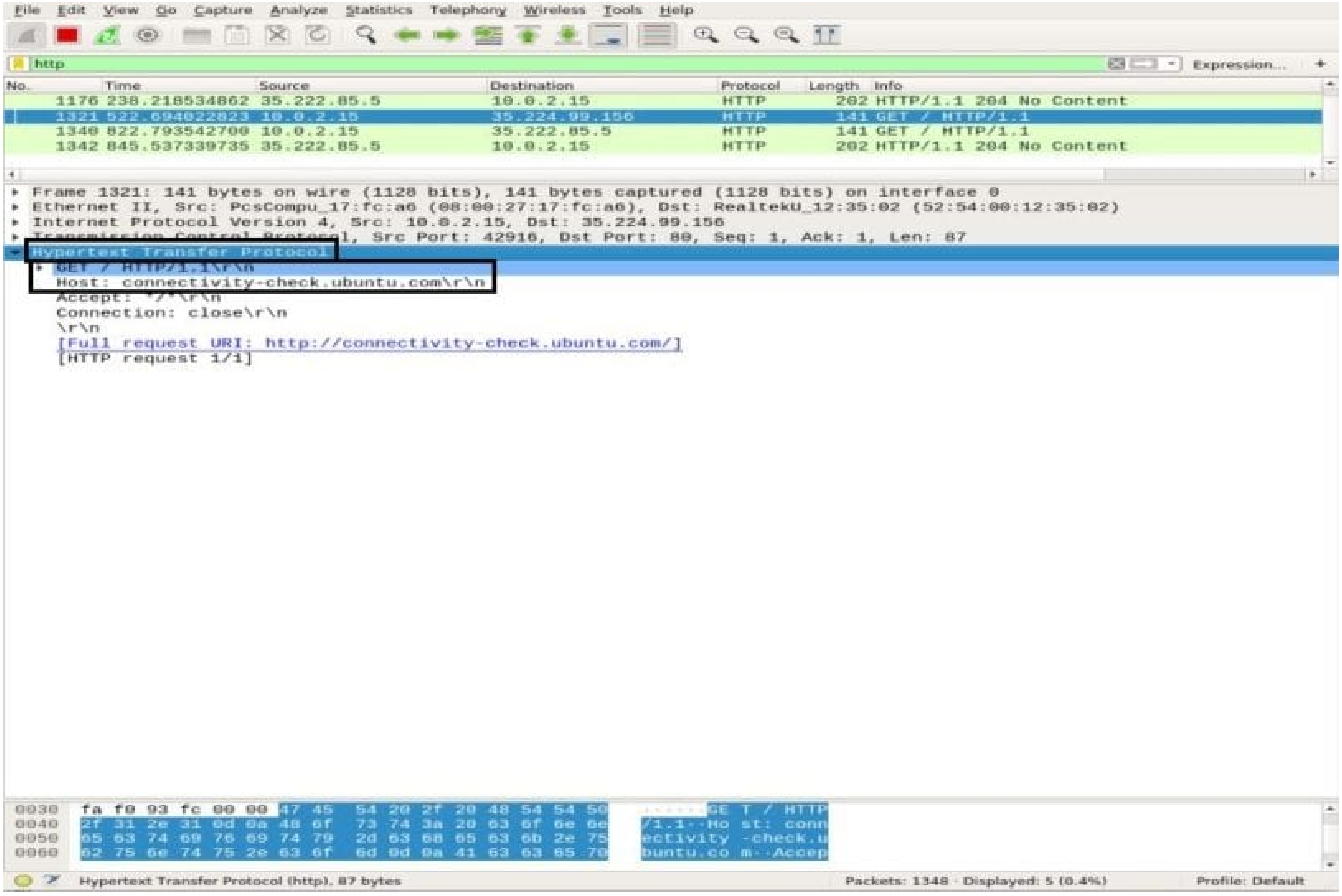
```
▸ Frame 1321: 141 bytes on wire (1128 bits), 141 bytes captured (1128 bits) on interface 0
▸ Ethernet II, Src: PcsCompu_17:fc:a6 (08:00:27:17:fc:a6), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
▸ Internet Protocol Version 4, Src: 10.0.2.15, Dst: 35.224.99.156
▾ Transmission Control Protocol, Src Port: 42916, Dst Port: 80, Seq: 1, Ack: 1, Len: 87
    Source Port: 42916
    Destination Port: 80
    [Stream index: 56]
    [TCP Segment Len: 87]
    Sequence number: 1    (relative sequence number)
    [Next sequence number: 88    (relative sequence number)]
    Acknowledgment number: 1    (relative ack number)
    0101 .... = Header Length: 20 bytes (5)
  ▸ Flags: 0x018 (PSH, ACK)
    Window size value: 64240
    [Calculated window size: 64240]
    [Window size scaling factor: -2 (no window scaling used)]
    Checksum: 0x93fc [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0
  ▸ [SEQ/ACK analysis]
  ▸ [Timestamps]
    TCP payload (87 bytes)
▾ Hypertext Transfer Protocol
  ▸ GET / HTTP/1.1\r\n
    Host: connectivity-check.ubuntu.com\r\n
    Accept: */*\r\n
    Connection: close\r\n
    \r\n
    [Full request URI: http://connectivity-check.ubuntu.com/]
    [HTTP request 1/1]
```

```
0020  63 9c a7 a4 00 50 ec 57  6e 4f f4 a2 1c 02 50 18   c····P·W nO····P·
0030  fa f0 93 fc 00 00 47 45  54 20 2f 20 48 54 54 50   ······GE T / HTTP
0040  2f 31 2e 31 0d 0a 48 6f  73 74 3a 20 63 6f 6e 6e   /1.1··Ho st: conn
0050  65 63 74 69 76 69 74 79  2d 63 68 65 63 6b 2e 75   ectivity -check.u
```

Transmission Control Protocol (tcp), 20 bytes   Packets: 1336 · Displayed: 3 (0.2%)   Profile: Default

Application Layer

In the final layer, you can see what type of data is being sent over the medium and which application is being used, such as FTP, HTTP, SSH, etc.


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Expression...

http

```
No.   Time             Source        Destination    Protocol Length Info
 1176 238.218534862 35.222.85.5   10.0.2.15      HTTP     202 HTTP/1.1 204 No Content
 1321 522.694022823 10.0.2.15     35.224.99.156  HTTP     141 GET / HTTP/1.1
 1340 822.793542700 10.0.2.15     35.222.85.5    HTTP     141 GET / HTTP/1.1
 1342 845.537339735 35.222.85.5   10.0.2.15      HTTP     202 HTTP/1.1 204 No Content
```

```
▸ Frame 1321: 141 bytes on wire (1128 bits), 141 bytes captured (1128 bits) on interface 0
▸ Ethernet II, Src: PcsCompu_17:fc:a6 (08:00:27:17:fc:a6), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
▸ Internet Protocol Version 4, Src: 10.0.2.15, Dst: 35.224.99.156
▸ Transmission Control Protocol, Src Port: 42916, Dst Port: 80, Seq: 1, Ack: 1, Len: 87
▾ Hypertext Transfer Protocol
  ▸ GET / HTTP/1.1\r\n
    Host: connectivity-check.ubuntu.com\r\n
    Accept: */*\r\n
    Connection: close\r\n
    \r\n
    [Full request URI: http://connectivity-check.ubuntu.com/]
    [HTTP request 1/1]
```

```
0030  fa f0 93 fc 00 00 47 45  54 20 2f 20 48 54 54 50   ......GE T / HTTP
0040  2f 31 2e 31 0d 0a 48 6f  73 74 3a 20 63 6f 6e 6e   /1.1..Ho st: conn
0050  65 63 74 69 76 69 74 79  2d 63 68 65 63 6b 2e 75   ectivity -check.u
0060  62 75 6e 74 75 2e 63 6f  6d 0d 0a 41 63 63 65 70   buntu.co m..Accep
```

Hypertext Transfer Protocol (http), 87 bytes   Packets: 1348 · Displayed: 5 (0.4%)   Profile: Default

Traffic Analysis

ICMP Traffic Analysis

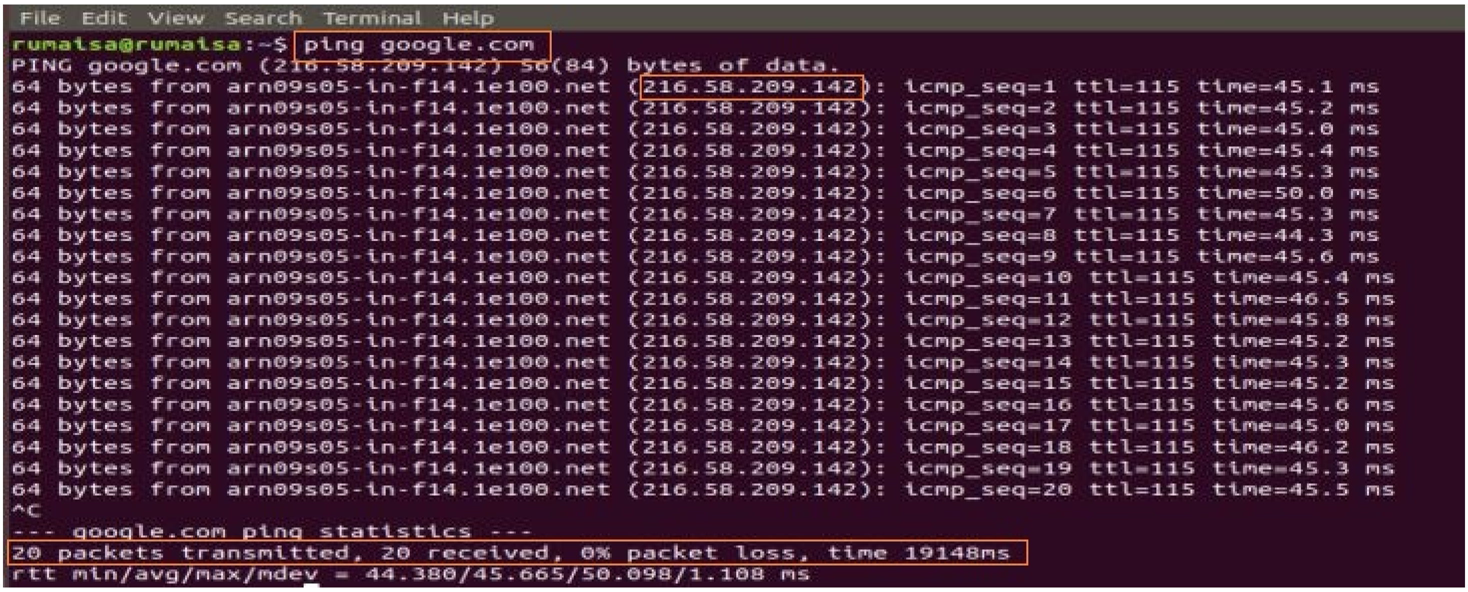
ICMP is used for error reporting and testing by determining if the data reaches the intended destination on time or not. Ping utility uses ICMP messages to test the speed of the connection between devices, and report how long the packet takes to reach its destination then come back.

The ping uses ICMP_echo_request message to the device on the network, and the device responds by ICMP_echo_reply message. To capture packets on the

Wireshark, start the Capture function of the Wireshark, open the terminal, and run the following command:

ubuntu$ubuntu:~$ ping google.com

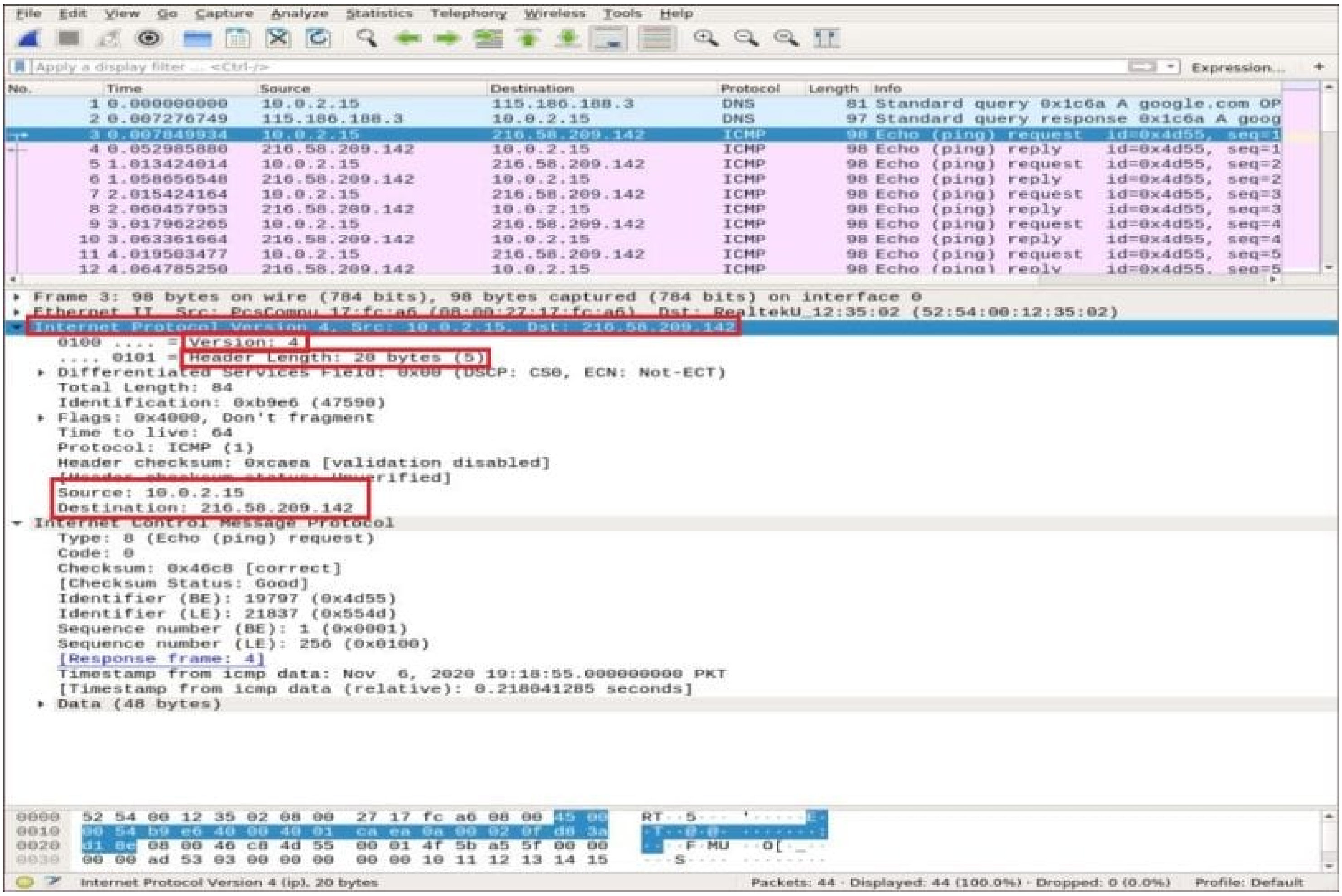
Use **Ctrl+C** to terminate the packet capture process in Wireshark. In the snapshot below, you can notice the **ICMP packet sent = ICMP packet received** with 0% packet loss.



```
rumaisa@rumaisa:~$ ping google.com
PING google.com (216.58.209.142) 56(84) bytes of data.
64 bytes from arn09s05-in-f14.1e100.net (216.58.209.142): icmp_seq=1 ttl=115 time=45.1 ms
64 bytes from arn09s05-in-f14.1e100.net (216.58.209.142): icmp_seq=2 ttl=115 time=45.2 ms
64 bytes from arn09s05-in-f14.1e100.net (216.58.209.142): icmp_seq=3 ttl=115 time=45.0 ms
64 bytes from arn09s05-in-f14.1e100.net (216.58.209.142): icmp_seq=4 ttl=115 time=45.4 ms
64 bytes from arn09s05-in-f14.1e100.net (216.58.209.142): icmp_seq=5 ttl=115 time=45.3 ms
64 bytes from arn09s05-in-f14.1e100.net (216.58.209.142): icmp_seq=6 ttl=115 time=50.0 ms
64 bytes from arn09s05-in-f14.1e100.net (216.58.209.142): icmp_seq=7 ttl=115 time=45.3 ms
64 bytes from arn09s05-in-f14.1e100.net (216.58.209.142): icmp_seq=8 ttl=115 time=44.3 ms
64 bytes from arn09s05-in-f14.1e100.net (216.58.209.142): icmp_seq=9 ttl=115 time=45.6 ms
64 bytes from arn09s05-in-f14.1e100.net (216.58.209.142): icmp_seq=10 ttl=115 time=45.4 ms
64 bytes from arn09s05-in-f14.1e100.net (216.58.209.142): icmp_seq=11 ttl=115 time=46.5 ms
64 bytes from arn09s05-in-f14.1e100.net (216.58.209.142): icmp_seq=12 ttl=115 time=45.8 ms
64 bytes from arn09s05-in-f14.1e100.net (216.58.209.142): icmp_seq=13 ttl=115 time=45.2 ms
64 bytes from arn09s05-in-f14.1e100.net (216.58.209.142): icmp_seq=14 ttl=115 time=45.3 ms
64 bytes from arn09s05-in-f14.1e100.net (216.58.209.142): icmp_seq=15 ttl=115 time=45.2 ms
64 bytes from arn09s05-in-f14.1e100.net (216.58.209.142): icmp_seq=16 ttl=115 time=45.6 ms
64 bytes from arn09s05-in-f14.1e100.net (216.58.209.142): icmp_seq=17 ttl=115 time=45.0 ms
64 bytes from arn09s05-in-f14.1e100.net (216.58.209.142): icmp_seq=18 ttl=115 time=46.2 ms
64 bytes from arn09s05-in-f14.1e100.net (216.58.209.142): icmp_seq=19 ttl=115 time=45.3 ms
64 bytes from arn09s05-in-f14.1e100.net (216.58.209.142): icmp_seq=20 ttl=115 time=45.5 ms
^C
--- google.com ping statistics ---
20 packets transmitted, 20 received, 0% packet loss, time 19148ms
rtt min/avg/max/mdev = 44.380/45.665/50.098/1.108 ms
```

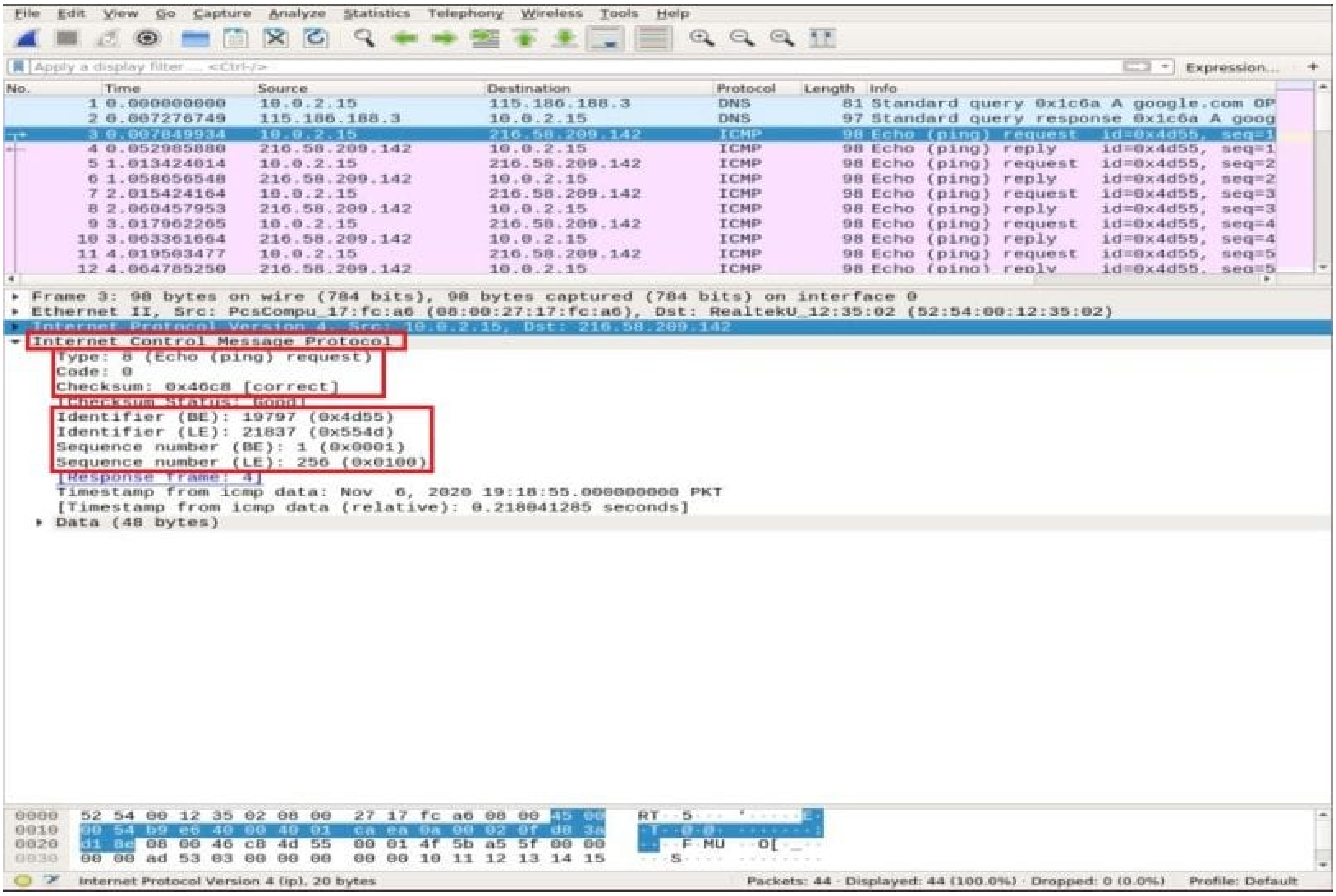
In the Wireshark capture pane, select the first ICMP_echo_request packet and observe the details by opening the middle Wireshark pane.

In the Network Layer, you can notice the source **Src** as my ip_address, whereas the destination **Dst** ip_address is of Google server, whereas the IP layer mentions the protocol to be ICMP.


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...

```
No.   Time          Source           Destination      Protocol Length Info
    1 0.000000000   10.0.2.15        115.186.188.3    DNS      81 Standard query 0x1c6a A google.com OP
    2 0.007276749   115.186.188.3    10.0.2.15        DNS      97 Standard query response 0x1c6a A goog
    3 0.007849934   10.0.2.15        216.58.209.142   ICMP     98 Echo (ping) request  id=0x4d55, seq=1
    4 0.052985880   216.58.209.142   10.0.2.15        ICMP     98 Echo (ping) reply    id=0x4d55, seq=1
    5 1.013424014   10.0.2.15        216.58.209.142   ICMP     98 Echo (ping) request  id=0x4d55, seq=2
    6 1.058656548   216.58.209.142   10.0.2.15        ICMP     98 Echo (ping) reply    id=0x4d55, seq=2
    7 2.015424164   10.0.2.15        216.58.209.142   ICMP     98 Echo (ping) request  id=0x4d55, seq=3
    8 2.060457953   216.58.209.142   10.0.2.15        ICMP     98 Echo (ping) reply    id=0x4d55, seq=3
    9 3.017962265   10.0.2.15        216.58.209.142   ICMP     98 Echo (ping) request  id=0x4d55, seq=4
   10 3.063361664   216.58.209.142   10.0.2.15        ICMP     98 Echo (ping) reply    id=0x4d55, seq=4
   11 4.019583477   10.0.2.15        216.58.209.142   ICMP     98 Echo (ping) request  id=0x4d55, seq=5
   12 4.064785250   216.58.209.142   10.0.2.15        ICMP     98 Echo (ping) reply    id=0x4d55, seq=5

▸ Frame 3: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface 0
▸ Ethernet II, Src: PcsCompu_17:fc:a6 (08:00:27:17:fc:a6), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
▾ Internet Protocol Version 4, Src: 10.0.2.15, Dst: 216.58.209.142
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 84
    Identification: 0xb9e6 (47590)
  ▸ Flags: 0x4000, Don't fragment
    Time to live: 64
    Protocol: ICMP (1)
    Header checksum: 0xcaea [validation disabled]
    [Header checksum status: Unverified]
    Source: 10.0.2.15
    Destination: 216.58.209.142
▾ Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
    Checksum: 0x46c8 [correct]
    [Checksum Status: Good]
    Identifier (BE): 19797 (0x4d55)
    Identifier (LE): 21837 (0x554d)
    Sequence number (BE): 1 (0x0001)
    Sequence number (LE): 256 (0x0100)
    [Response frame: 4]
    Timestamp from icmp data: Nov  6, 2020 19:18:55.000000000 PKT
    [Timestamp from icmp data (relative): 0.218041285 seconds]
  ▸ Data (48 bytes)

0000  52 54 00 12 35 02 08 00  27 17 fc a6 08 00 45 00   RT··5··· '·····E·
0010  00 54 b9 e6 40 00 40 01  ca ea 0a 00 02 0f d8 3a   ·T··@·@· ·······:
0020  d1 8e 08 00 46 c8 4d 55  00 01 4f 5b a5 5f 00 00   ····F·MU ··O[·_··
0030  00 00 ad 53 03 00 00 00  00 00 10 11 12 13 14 15   ···S···· ········

Internet Protocol Version 4 (ip), 20 bytes        Packets: 44 · Displayed: 44 (100.0%) · Dropped: 0 (0.0%)    Profile: Default
```

Now, we zoom into the ICMP packet details by expanding Internet Control Message Protocol and decode the highlighted boxes in the snapshot below:



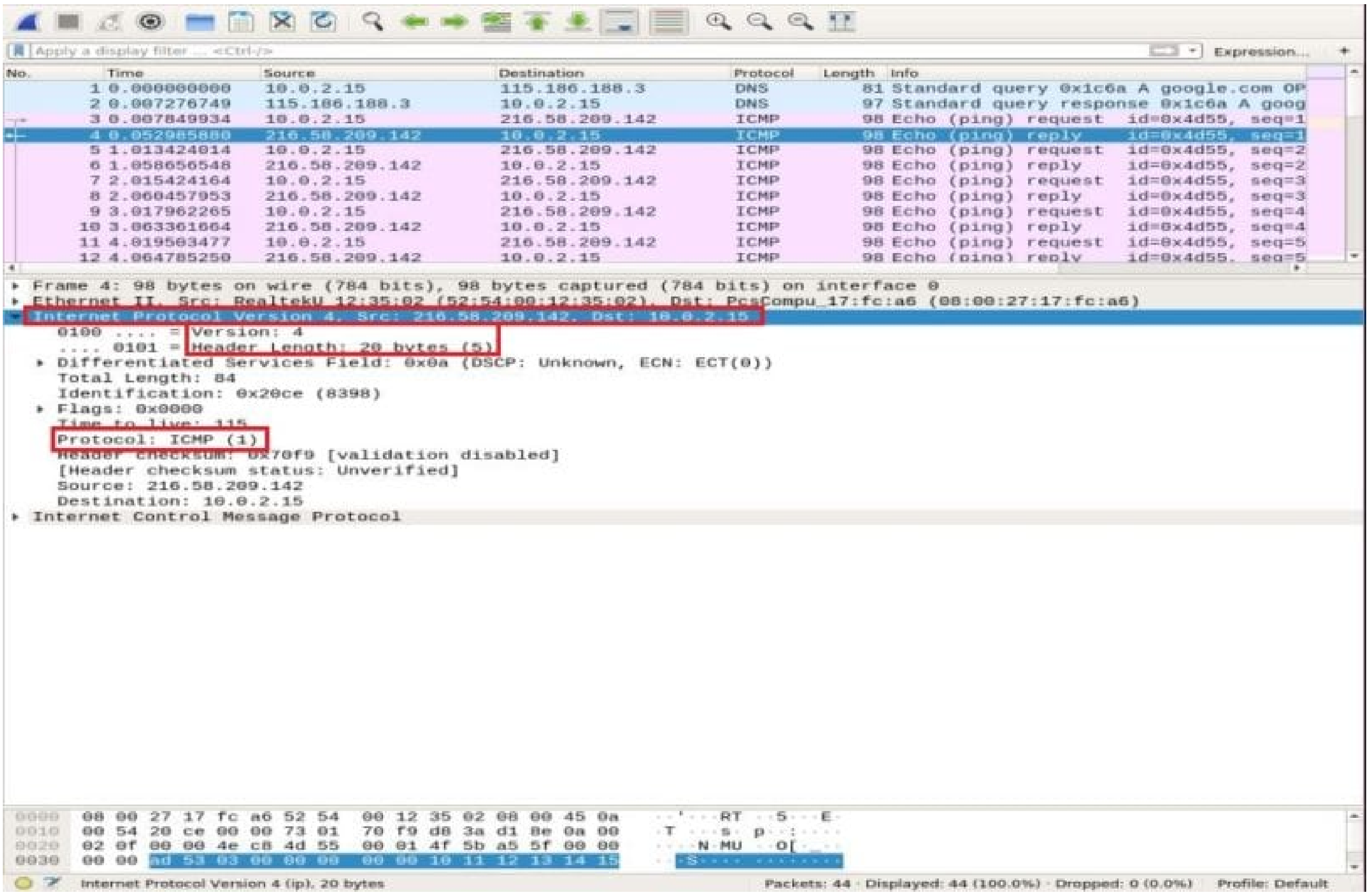
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>
Expression…

```
No.  Time         Source          Destination     Protocol Length Info
   1 0.000000000  10.0.2.15       115.186.188.3   DNS        81 Standard query 0x1c6a A google.com OP
   2 0.007276749  115.186.188.3   10.0.2.15       DNS        97 Standard query response 0x1c6a A goog
   3 0.007849934  10.0.2.15       216.58.209.142  ICMP       98 Echo (ping) request  id=0x4d55, seq=1
   4 0.052985880  216.58.209.142  10.0.2.15       ICMP       98 Echo (ping) reply    id=0x4d55, seq=1
   5 1.013424014  10.0.2.15       216.58.209.142  ICMP       98 Echo (ping) request  id=0x4d55, seq=2
   6 1.058656548  216.58.209.142  10.0.2.15       ICMP       98 Echo (ping) reply    id=0x4d55, seq=2
   7 2.015424164  10.0.2.15       216.58.209.142  ICMP       98 Echo (ping) request  id=0x4d55, seq=3
   8 2.060457953  216.58.209.142  10.0.2.15       ICMP       98 Echo (ping) reply    id=0x4d55, seq=3
   9 3.017962265  10.0.2.15       216.58.209.142  ICMP       98 Echo (ping) request  id=0x4d55, seq=4
  10 3.063361664  216.58.209.142  10.0.2.15       ICMP       98 Echo (ping) reply    id=0x4d55, seq=4
  11 4.019503477  10.0.2.15       216.58.209.142  ICMP       98 Echo (ping) request  id=0x4d55, seq=5
  12 4.064785250  216.58.209.142  10.0.2.15       ICMP       98 Echo (ping) reply    id=0x4d55, seq=5

▸ Frame 3: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface 0
▸ Ethernet II, Src: PcsCompu_17:fc:a6 (08:00:27:17:fc:a6), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
▸ Internet Protocol Version 4, Src: 10.0.2.15, Dst: 216.58.209.142
▾ Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
    Checksum: 0x46c8 [correct]
    [Checksum Status: Good]
    Identifier (BE): 19797 (0x4d55)
    Identifier (LE): 21837 (0x554d)
    Sequence number (BE): 1 (0x0001)
    Sequence number (LE): 256 (0x0100)
    [Response frame: 4]
    Timestamp from icmp data: Nov  6, 2020 19:18:55.000000000 PKT
    [Timestamp from icmp data (relative): 0.218041285 seconds]
  ▸ Data (48 bytes)

0000  52 54 00 12 35 02 08 00  27 17 fc a6 08 00 45 00   RT··5···'·····E·
0010  00 54 b9 e6 40 00 40 01  ca ea 0a 00 02 0f d8 3a   ·T··@·@·········:
0020  d1 8e 08 00 46 c8 4d 55  00 01 4f 5b a5 5f 00 00   ····F·MU··O[·_··
0030  00 00 ad 53 03 00 00 00  00 00 10 11 12 13 14 15   ···S············

Internet Protocol Version 4 (ip), 20 bytes      Packets: 44 · Displayed: 44 (100.0%) · Dropped: 0 (0.0%)      Profile: Default
```

- Type: 08-bit field set to 8 means Echo request message
- Code: always zero for ICMP packets
- checksum: 0x46c8
- Identifier Number (BE): 19797
- Identifier Number (LE): 21837
- Sequence Number (BE): 1
- Sequence Number (LE): 256

The identifier and the sequence numbers are matched to aid in identifying the replies to echo requests. Similarly, before packet transmission, the checksum is computed and added to the field to be compared against the checksum in the received data packet.

Now, in the ICMP reply packet, notice the IPv4 layer. The source and destination addresses have swapped.



```
No.   Time          Source            Destination       Protocol  Length Info
    1 0.000000000   10.0.2.15         115.186.188.3     DNS           81 Standard query 0x1c6a A google.com OP
    2 0.007276749   115.186.188.3     10.0.2.15         DNS           97 Standard query response 0x1c6a A goog
    3 0.007849934   10.0.2.15         216.58.209.142    ICMP          98 Echo (ping) request  id=0x4d55, seq=1
    4 0.052985880   216.58.209.142    10.0.2.15         ICMP          98 Echo (ping) reply    id=0x4d55, seq=1
    5 1.013424014   10.0.2.15         216.58.209.142    ICMP          98 Echo (ping) request  id=0x4d55, seq=2
    6 1.058656548   216.58.209.142    10.0.2.15         ICMP          98 Echo (ping) reply    id=0x4d55, seq=2
    7 2.015424164   10.0.2.15         216.58.209.142    ICMP          98 Echo (ping) request  id=0x4d55, seq=3
    8 2.060457953   216.58.209.142    10.0.2.15         ICMP          98 Echo (ping) reply    id=0x4d55, seq=3
    9 3.017962265   10.0.2.15         216.58.209.142    ICMP          98 Echo (ping) request  id=0x4d55, seq=4
   10 3.063361664   216.58.209.142    10.0.2.15         ICMP          98 Echo (ping) reply    id=0x4d55, seq=4
   11 4.019503477   10.0.2.15         216.58.209.142    ICMP          98 Echo (ping) request  id=0x4d55, seq=5
   12 4.064785250   216.58.209.142    10.0.2.15         ICMP          98 Echo (ping) reply    id=0x4d55, seq=5
```

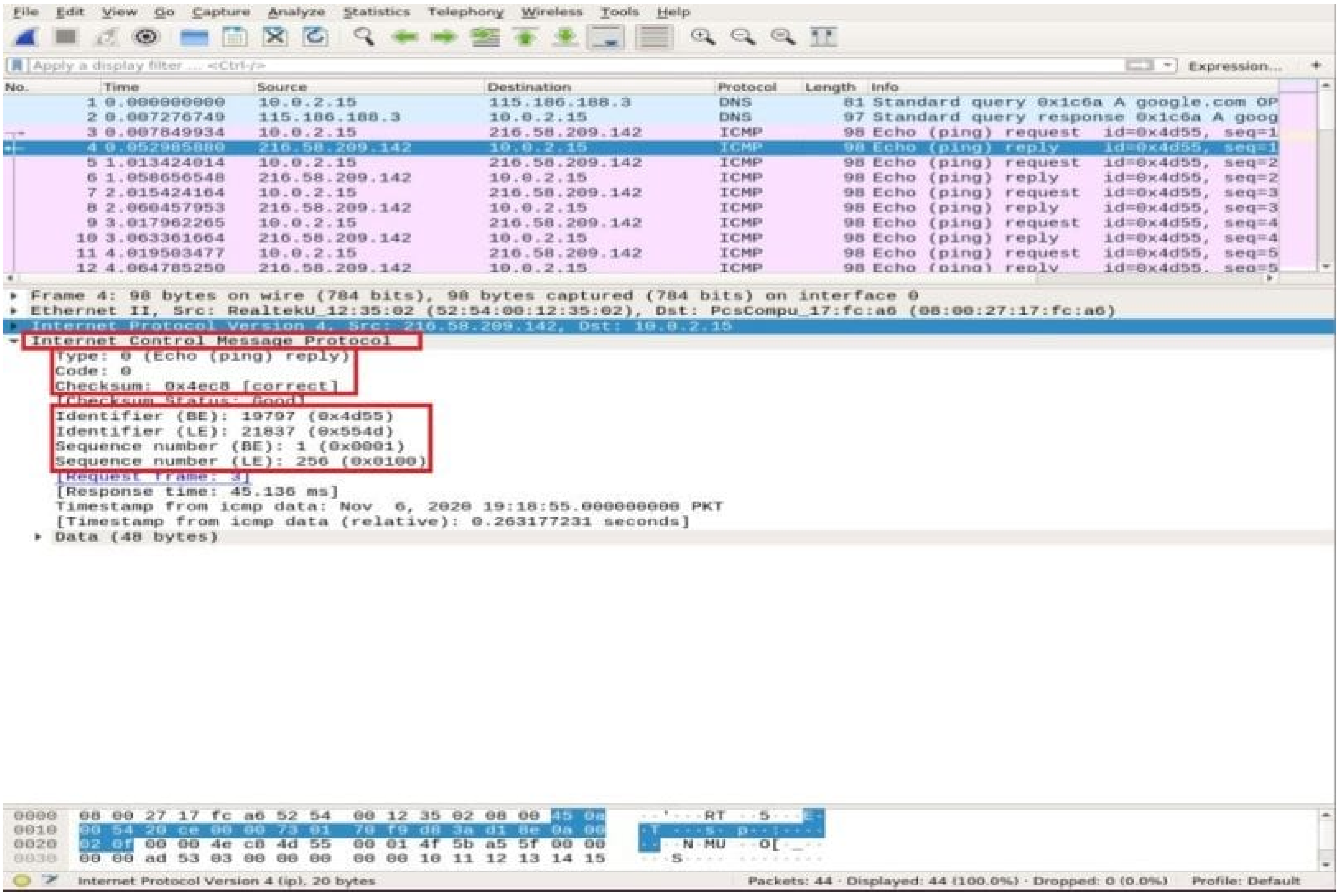
```
▸ Frame 4: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface 0
▸ Ethernet II, Src: RealtekU_12:35:02 (52:54:00:12:35:02), Dst: PcsCompu_17:fc:a6 (08:00:27:17:fc:a6)
▾ Internet Protocol Version 4, Src: 216.58.209.142, Dst: 10.0.2.15
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x0a (DSCP: Unknown, ECN: ECT(0))
    Total Length: 84
    Identification: 0x20ce (8398)
  ▸ Flags: 0x0000
    Time to live: 115
    Protocol: ICMP (1)
    Header checksum: 0x70f9 [validation disabled]
    [Header checksum status: Unverified]
    Source: 216.58.209.142
    Destination: 10.0.2.15
▸ Internet Control Message Protocol
```

```
0000  08 00 27 17 fc a6 52 54  00 12 35 02 08 00 45 0a   ··'···RT ··5···E·
0010  00 54 20 ce 00 00 73 01  70 f9 d8 3a d1 8e 0a 00   ·T ···s· p··:····
0020  02 0f 00 00 4e c8 4d 55  00 01 4f 5b a5 5f 00 00   ····N·MU ··O[·_··
0030  00 00 ad 53 03 00 00 00  00 00 10 11 12 13 14 15   ···S···· ········
```

Internet Protocol Version 4 (ip), 20 bytes
Packets: 44 · Displayed: 44 (100.0%) · Dropped: 0 (0.0%)
Profile: Default

In the ICMP layer, verify and compare the following important fields:

- Type: 08-bit field set to 0 means Echo reply message
- Code: always 0 for ICMP packets
- checksum: 0x46c8
- Identifier Number (BE): 19797
- Identifier Number (LE): 21837
- Sequence Number (BE): 1
- Sequence Number (LE): 256


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>
Expression…

```
No.   Time          Source           Destination      Protocol Length Info
    1 0.000000000   10.0.2.15        115.186.188.3    DNS      81 Standard query 0x1c6a A google.com OP
    2 0.007276749   115.186.188.3    10.0.2.15        DNS      97 Standard query response 0x1c6a A goog
    3 0.007849934   10.0.2.15        216.58.209.142   ICMP     98 Echo (ping) request  id=0x4d55, seq=1
    4 0.052985880   216.58.209.142   10.0.2.15        ICMP     98 Echo (ping) reply    id=0x4d55, seq=1
    5 1.013424014   10.0.2.15        216.58.209.142   ICMP     98 Echo (ping) request  id=0x4d55, seq=2
    6 1.058656548   216.58.209.142   10.0.2.15        ICMP     98 Echo (ping) reply    id=0x4d55, seq=2
    7 2.015424164   10.0.2.15        216.58.209.142   ICMP     98 Echo (ping) request  id=0x4d55, seq=3
    8 2.060457953   216.58.209.142   10.0.2.15        ICMP     98 Echo (ping) reply    id=0x4d55, seq=3
    9 3.017962265   10.0.2.15        216.58.209.142   ICMP     98 Echo (ping) request  id=0x4d55, seq=4
   10 3.063361664   216.58.209.142   10.0.2.15        ICMP     98 Echo (ping) reply    id=0x4d55, seq=4
   11 4.019503477   10.0.2.15        216.58.209.142   ICMP     98 Echo (ping) request  id=0x4d55, seq=5
   12 4.064785250   216.58.209.142   10.0.2.15        ICMP     98 Echo (ping) reply    id=0x4d55, seq=5
```

```
▸ Frame 4: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface 0
▸ Ethernet II, Src: RealtekU_12:35:02 (52:54:00:12:35:02), Dst: PcsCompu_17:fc:a6 (08:00:27:17:fc:a6)
▸ Internet Protocol Version 4, Src: 216.58.209.142, Dst: 10.0.2.15
▾ Internet Control Message Protocol
    Type: 0 (Echo (ping) reply)
    Code: 0
    Checksum: 0x4ec8 [correct]
    [Checksum Status: Good]
    Identifier (BE): 19797 (0x4d55)
    Identifier (LE): 21837 (0x554d)
    Sequence number (BE): 1 (0x0001)
    Sequence number (LE): 256 (0x0100)
    [Request frame: 3]
    [Response time: 45.136 ms]
    Timestamp from icmp data: Nov  6, 2020 19:18:55.000000000 PKT
    [Timestamp from icmp data (relative): 0.263177231 seconds]
  ▸ Data (48 bytes)
```

```
0000  08 00 27 17 fc a6 52 54  00 12 35 02 08 00 45 0a   ··'···RT ··5···E·
0010  00 54 20 ce 00 00 73 01  70 f9 d8 3a d1 8e 0a 00   ·T ···s· p··:····
0020  02 0f 00 00 4e c8 4d 55  00 01 4f 5b a5 5f 00 00   ····N·MU ··O[·_··
0030  00 00 ad 53 03 00 00 00  00 00 10 11 12 13 14 15   ···S···· ········
```

Internet Protocol Version 4 (ip), 20 bytes   Packets: 44 · Displayed: 44 (100.0%) · Dropped: 0 (0.0%)   Profile: Default

You can notice that the ICMP reply echoes the same request checksum, identifier, and sequence number.

HTTP Traffic Analysis

HTTP is a Hypertext Transfer application layer protocol. It is used by the world wide web and defines rules when the HTTP client/server transmits/receives HTTP commands. The most commonly used HTTP methods ae POST and GET:

POST: this method is used to securely send confidential information to the server that doesn’t appear in the URL.

GET: this method is usually used to retrieve data from the address bar from a web server.

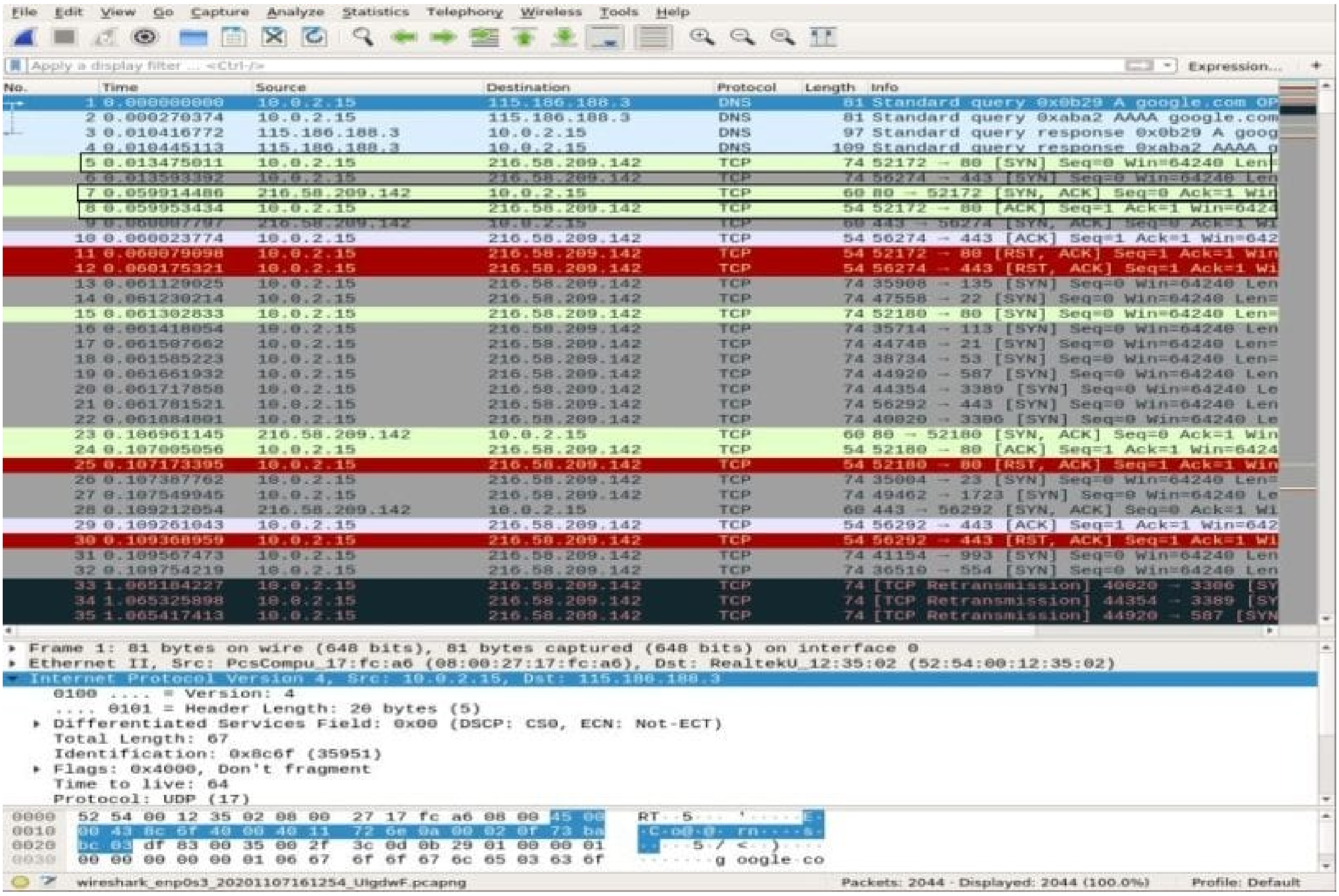
Before we dig deeper into HTTP packet analysis, we will first briefly demonstrate the TCP three-way-handshake in Wireshark.

TCP Three-Way-Handshake

In a three-way handshake, the client initiates a connection by sending an SYN packet and receiving an SYN-ACK response from the server, which is acknowledged by the client. We will use the Nmap TCP connect scan command to illustrate TCP handshake between client and server.

```
ubuntu$ubuntu:~$ nmap -sT google.com
```

In the Wireshark packet capture pane, scroll to the top of the window to notice various three-ways-handshakes established based on particular ports.



File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>
Expression…

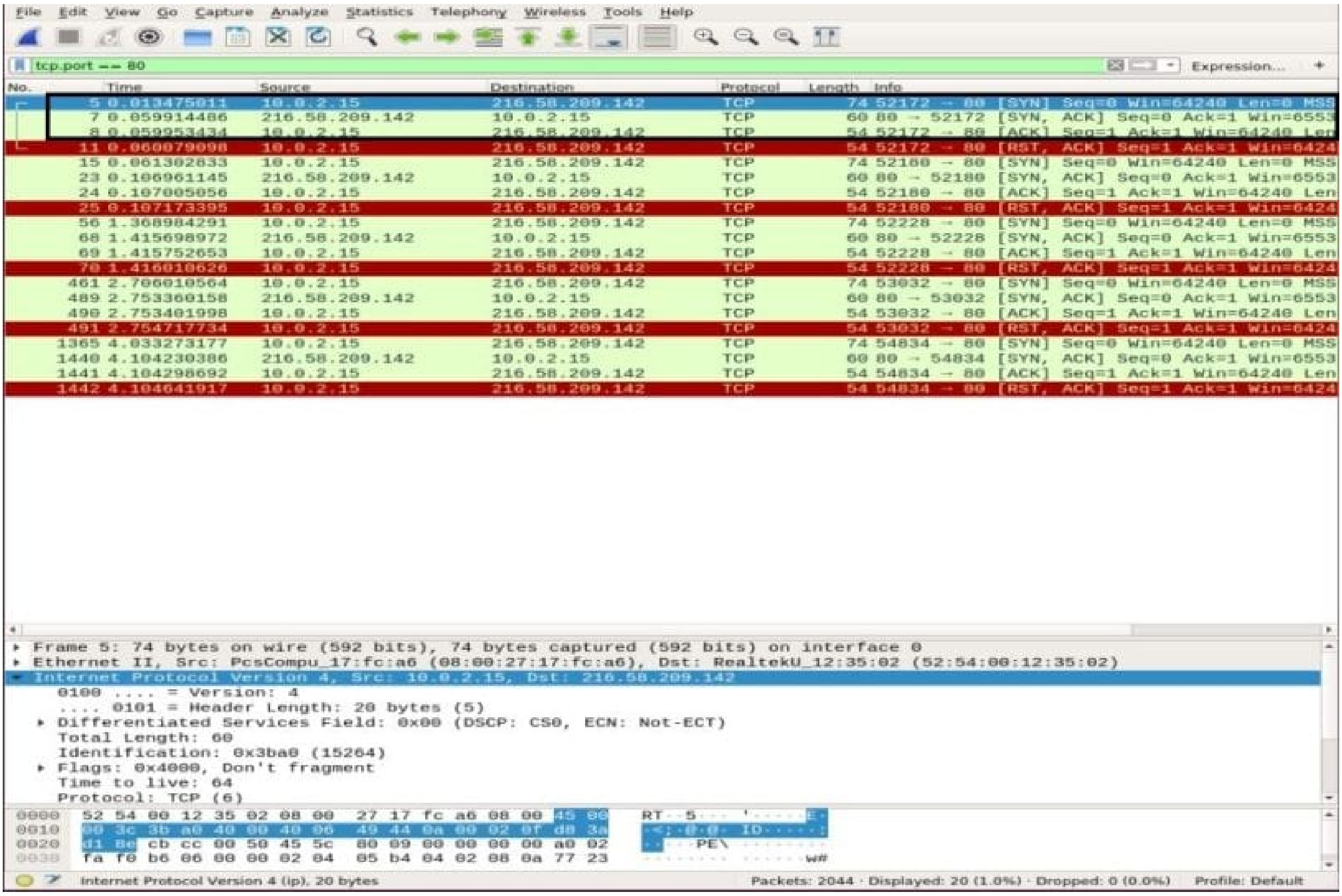
```
No.   Time          Source           Destination      Protocol Length Info
    1 0.000000000   10.0.2.15        115.186.188.3    DNS      81 Standard query 0x0b29 A google.com OP
    2 0.000270374   10.0.2.15        115.186.188.3    DNS      81 Standard query 0xaba2 AAAA google.com
    3 0.010416772   115.186.188.3    10.0.2.15        DNS      97 Standard query response 0x0b29 A goog
    4 0.010445113   115.186.188.3    10.0.2.15        DNS      109 Standard query response 0xaba2 AAAA g
    5 0.013475011   10.0.2.15        216.58.209.142   TCP      74 52172 → 80 [SYN] Seq=0 Win=64240 Len=
    6 0.013593392   10.0.2.15        216.58.209.142   TCP      74 56274 → 443 [SYN] Seq=0 Win=64240 Len
    7 0.059914486   216.58.209.142   10.0.2.15        TCP      60 80 → 52172 [SYN, ACK] Seq=0 Ack=1 Win
    8 0.059953434   10.0.2.15        216.58.209.142   TCP      54 52172 → 80 [ACK] Seq=1 Ack=1 Win=6424
    9 0.060007797   216.58.209.142   10.0.2.15        TCP      60 443 → 56274 [SYN, ACK] Seq=0 Ack=1 Wi
   10 0.060023774   10.0.2.15        216.58.209.142   TCP      54 56274 → 443 [ACK] Seq=1 Ack=1 Win=642
   11 0.060079098   10.0.2.15        216.58.209.142   TCP      54 52172 → 80 [RST, ACK] Seq=1 Ack=1 Win
   12 0.060175321   10.0.2.15        216.58.209.142   TCP      54 56274 → 443 [RST, ACK] Seq=1 Ack=1 Wi
   13 0.061129025   10.0.2.15        216.58.209.142   TCP      74 35908 → 135 [SYN] Seq=0 Win=64240 Len
   14 0.061230214   10.0.2.15        216.58.209.142   TCP      74 47558 → 22 [SYN] Seq=0 Win=64240 Len=
   15 0.061302833   10.0.2.15        216.58.209.142   TCP      74 52180 → 80 [SYN] Seq=0 Win=64240 Len=
   16 0.061418054   10.0.2.15        216.58.209.142   TCP      74 35714 → 113 [SYN] Seq=0 Win=64240 Len
   17 0.061507662   10.0.2.15        216.58.209.142   TCP      74 44748 → 21 [SYN] Seq=0 Win=64240 Len=
   18 0.061585223   10.0.2.15        216.58.209.142   TCP      74 38734 → 53 [SYN] Seq=0 Win=64240 Len=
   19 0.061661932   10.0.2.15        216.58.209.142   TCP      74 44920 → 587 [SYN] Seq=0 Win=64240 Len
   20 0.061717858   10.0.2.15        216.58.209.142   TCP      74 44354 → 3389 [SYN] Seq=0 Win=64240 Le
   21 0.061781521   10.0.2.15        216.58.209.142   TCP      74 56292 → 443 [SYN] Seq=0 Win=64240 Len
   22 0.061884801   10.0.2.15        216.58.209.142   TCP      74 40020 → 3306 [SYN] Seq=0 Win=64240 Le
   23 0.106961145   216.58.209.142   10.0.2.15        TCP      60 80 → 52180 [SYN, ACK] Seq=0 Ack=1 Win
   24 0.107005056   10.0.2.15        216.58.209.142   TCP      54 52180 → 80 [ACK] Seq=1 Ack=1 Win=6424
   25 0.107173395   10.0.2.15        216.58.209.142   TCP      54 52180 → 80 [RST, ACK] Seq=1 Ack=1 Win
   26 0.107387762   10.0.2.15        216.58.209.142   TCP      74 35004 → 23 [SYN] Seq=0 Win=64240 Len=
   27 0.107549945   10.0.2.15        216.58.209.142   TCP      74 49462 → 1723 [SYN] Seq=0 Win=64240 Le
   28 0.109212054   216.58.209.142   10.0.2.15        TCP      60 443 → 56292 [SYN, ACK] Seq=0 Ack=1 Wi
   29 0.109261043   10.0.2.15        216.58.209.142   TCP      54 56292 → 443 [ACK] Seq=1 Ack=1 Win=642
   30 0.109368959   10.0.2.15        216.58.209.142   TCP      54 56292 → 443 [RST, ACK] Seq=1 Ack=1 Wi
   31 0.109567473   10.0.2.15        216.58.209.142   TCP      74 41154 → 993 [SYN] Seq=0 Win=64240 Len
   32 0.109754219   10.0.2.15        216.58.209.142   TCP      74 36510 → 554 [SYN] Seq=0 Win=64240 Len
   33 1.065184227   10.0.2.15        216.58.209.142   TCP      74 [TCP Retransmission] 40020 → 3306 [SY
   34 1.065325898   10.0.2.15        216.58.209.142   TCP      74 [TCP Retransmission] 44354 → 3389 [SY
   35 1.065417413   10.0.2.15        216.58.209.142   TCP      74 [TCP Retransmission] 44920 → 587 [SYN
```

```
▸ Frame 1: 81 bytes on wire (648 bits), 81 bytes captured (648 bits) on interface 0
▸ Ethernet II, Src: PcsCompu_17:fc:a6 (08:00:27:17:fc:a6), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
▾ Internet Protocol Version 4, Src: 10.0.2.15, Dst: 115.186.188.3
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 67
    Identification: 0x8c6f (35951)
  ▸ Flags: 0x4000, Don't fragment
    Time to live: 64
    Protocol: UDP (17)
```

```
0000  52 54 00 12 35 02 08 00  27 17 fc a6 08 00 45 00   RT··5···'·····E·
0010  00 43 8c 6f 40 00 40 11  72 6e 0a 00 02 0f 73 ba   ·C·o@·@·rn····s·
0020  bc 03 df 83 00 35 00 2f  3c 0d 0b 29 01 00 00 01   ·····5·/<··)····
0030  00 00 00 00 00 01 06 67  6f 6f 67 6c 65 03 63 6f   ·······g oogle·co
```

```
wireshark_enp0s3_20201107161254_UlgdwF.pcapng     Packets: 2044 · Displayed: 2044 (100.0%)     Profile: Default
```

Use the **tcp.port == 80** filter to see if the connection is established via port 80. You can notice the complete three-way-handshake, i.e., **SYN**, **SYN-ACK**, and **ACK,** highlighted at the top of the snapshot, illustrating a reliable connection.


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Expression...

tcp.port == 80

```
No.       Time           Source            Destination       Protocol  Length  Info
     5 0.013475011   10.0.2.15         216.58.209.142    TCP       74 52172 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS
     7 0.059914486   216.58.209.142    10.0.2.15         TCP       60 80 → 52172 [SYN, ACK] Seq=0 Ack=1 Win=6553
     8 0.059953434   10.0.2.15         216.58.209.142    TCP       54 52172 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len
    11 0.060079098   10.0.2.15         216.58.209.142    TCP       54 52172 → 80 [RST, ACK] Seq=1 Ack=1 Win=6424
    15 0.061302833   10.0.2.15         216.58.209.142    TCP       74 52180 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS
    23 0.106961145   216.58.209.142    10.0.2.15         TCP       60 80 → 52180 [SYN, ACK] Seq=0 Ack=1 Win=6553
    24 0.107005056   10.0.2.15         216.58.209.142    TCP       54 52180 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len
    25 0.107173395   10.0.2.15         216.58.209.142    TCP       54 52180 → 80 [RST, ACK] Seq=1 Ack=1 Win=6424
    56 1.368984291   10.0.2.15         216.58.209.142    TCP       74 52228 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS
    68 1.415698972   216.58.209.142    10.0.2.15         TCP       60 80 → 52228 [SYN, ACK] Seq=0 Ack=1 Win=6553
    69 1.415752653   10.0.2.15         216.58.209.142    TCP       54 52228 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len
    70 1.416010626   10.0.2.15         216.58.209.142    TCP       54 52228 → 80 [RST, ACK] Seq=1 Ack=1 Win=6424
   461 2.706010564   10.0.2.15         216.58.209.142    TCP       74 53032 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS
   489 2.753360158   216.58.209.142    10.0.2.15         TCP       60 80 → 53032 [SYN, ACK] Seq=0 Ack=1 Win=6553
   490 2.753401998   10.0.2.15         216.58.209.142    TCP       54 53032 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len
   491 2.754717734   10.0.2.15         216.58.209.142    TCP       54 53032 → 80 [RST, ACK] Seq=1 Ack=1 Win=6424
  1365 4.033273177   10.0.2.15         216.58.209.142    TCP       74 54834 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS
  1440 4.104230386   216.58.209.142    10.0.2.15         TCP       60 80 → 54834 [SYN, ACK] Seq=0 Ack=1 Win=6553
  1441 4.104298692   10.0.2.15         216.58.209.142    TCP       54 54834 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len
  1442 4.104641917   10.0.2.15         216.58.209.142    TCP       54 54834 → 80 [RST, ACK] Seq=1 Ack=1 Win=6424
```

```
▸ Frame 5: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
▸ Ethernet II, Src: PcsCompu_17:fc:a6 (08:00:27:17:fc:a6), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
▾ Internet Protocol Version 4, Src: 10.0.2.15, Dst: 216.58.209.142
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 60
    Identification: 0x3ba0 (15264)
  ▸ Flags: 0x4000, Don't fragment
    Time to live: 64
    Protocol: TCP (6)
```

```
0000  52 54 00 12 35 02 08 00  27 17 fc a6 08 00 45 00   RT··5··· '·····E·
0010  00 3c 3b a0 40 00 40 06  49 44 0a 00 02 0f d8 3a   ·<;·@·@· ID·····:
0020  d1 8e cb cc 00 50 45 5c  00 00 00 00 00 00 a0 02   ·····PE\ ········
0030  fa f0 b6 06 00 00 02 04  05 b4 04 02 08 0a 77 23   ········ ······w#
```

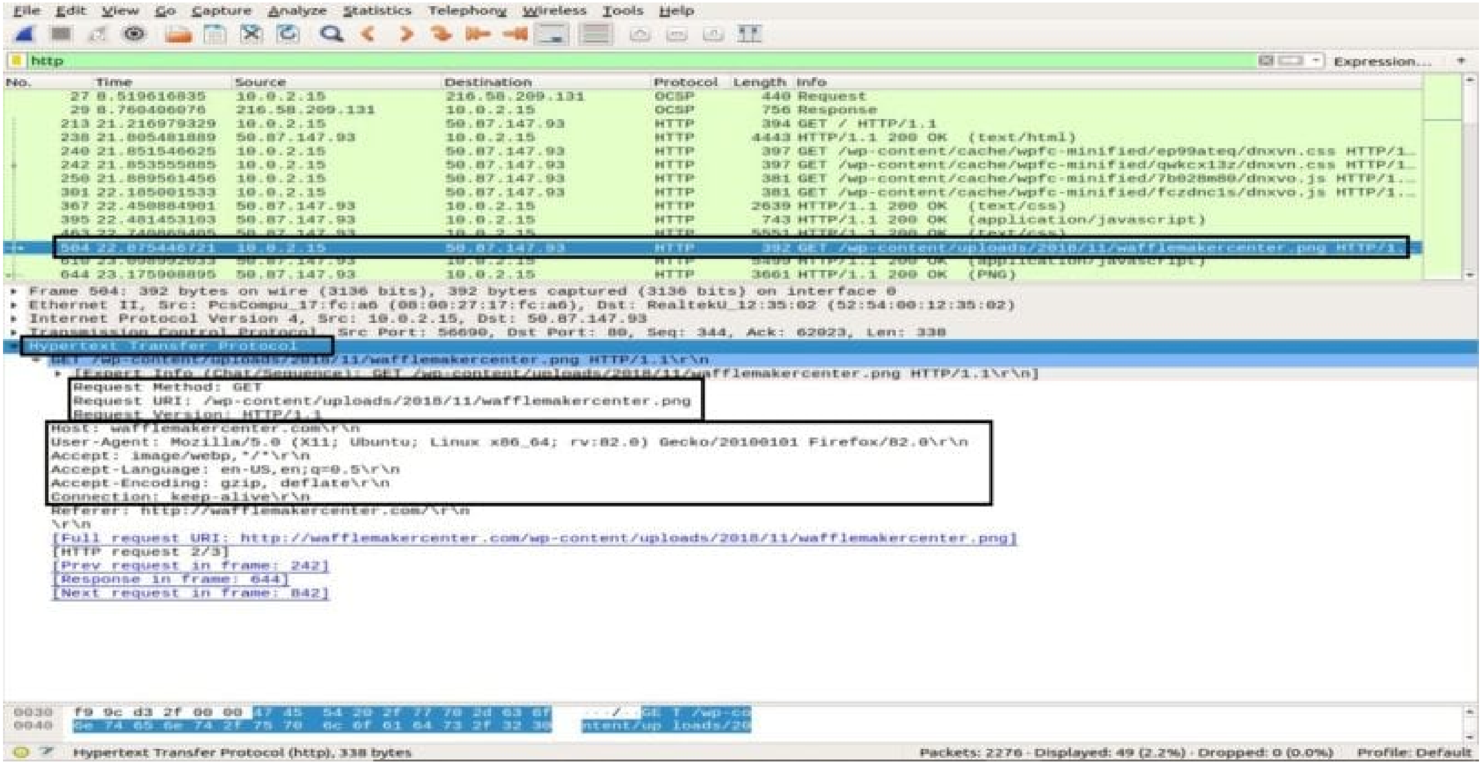
Internet Protocol Version 4 (ip), 20 bytes   Packets: 2044 · Displayed: 20 (1.0%) · Dropped: 0 (0.0%)   Profile: Default

HTTP Packet Analysis

For HTTP packet analysis, go to your browser and paste the Wireshark documentation URL: http://www.wafflemaker.com and download the user’s guide PDF. In the meantime, Wireshark must be capturing all the packets.

Apply an HTTP filter and look for the **HTTP GET** request sent to the server by the client. To view an HTTP packet, select it, and expand the application layer in the middle pane. There can be a lot of headers in a request, depending upon the website and browser as well. We will analyze the headers present in our request in the snapshot below.

- **Request Method:** the HTTP request method is GET
- **Host:** identifies the name of the server
- **User-Agent:** informs about the client-side browser type
- **Accept, Accept-Encoding, Accept-language:** informs the server about the file type, accepted encoding at the client-side, i.e., gzip, etc., and the accepted language
- **Cache-Control:** shows how the requested information is cached
- **Pragma:** shows the cookie's name and values the browser holds for the website
- **Connection:** header that controls whether the connection stays open after the transaction


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
http
Expression…

```
No.     Time          Source          Destination     Protocol Length Info
     27 8.519616835   10.0.2.15       216.58.209.131  OCSP      440 Request
     29 8.760406076   216.58.209.131  10.0.2.15       OCSP      756 Response
    213 21.216979329  10.0.2.15       50.87.147.93    HTTP      394 GET / HTTP/1.1
    238 21.805481889  50.87.147.93    10.0.2.15       HTTP     4443 HTTP/1.1 200 OK  (text/html)
    240 21.851546625  10.0.2.15       50.87.147.93    HTTP      397 GET /wp-content/cache/wpfc-minified/ep99ateq/dnxvn.css HTTP/1…
    242 21.853555885  10.0.2.15       50.87.147.93    HTTP      397 GET /wp-content/cache/wpfc-minified/qwkcx13z/dnxvn.css HTTP/1…
    250 21.889561456  10.0.2.15       50.87.147.93    HTTP      381 GET /wp-content/cache/wpfc-minified/7b028m80/dnxvo.js HTTP/1.…
    301 22.185001533  10.0.2.15       50.87.147.93    HTTP      381 GET /wp-content/cache/wpfc-minified/fczdnc1s/dnxvo.js HTTP/1.…
    367 22.450884901  50.87.147.93    10.0.2.15       HTTP     2639 HTTP/1.1 200 OK  (text/css)
    395 22.481453103  50.87.147.93    10.0.2.15       HTTP      743 HTTP/1.1 200 OK  (application/javascript)
    463 22.740868405  50.87.147.93    10.0.2.15       HTTP     5551 HTTP/1.1 200 OK  (text/css)
    504 22.875446721  10.0.2.15       50.87.147.93    HTTP      392 GET /wp-content/uploads/2018/11/wafflemakercenter.png HTTP/1.…
    610 23.098992033  50.87.147.93    10.0.2.15       HTTP     5499 HTTP/1.1 200 OK  (application/javascript)
    644 23.175908895  50.87.147.93    10.0.2.15       HTTP     3661 HTTP/1.1 200 OK  (PNG)
```

```
▸ Frame 504: 392 bytes on wire (3136 bits), 392 bytes captured (3136 bits) on interface 0
▸ Ethernet II, Src: PcsCompu_17:fc:a6 (08:00:27:17:fc:a6), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
▸ Internet Protocol Version 4, Src: 10.0.2.15, Dst: 50.87.147.93
▸ Transmission Control Protocol, Src Port: 56690, Dst Port: 80, Seq: 344, Ack: 62023, Len: 338
▾ Hypertext Transfer Protocol
  ▾ GET /wp-content/uploads/2018/11/wafflemakercenter.png HTTP/1.1\r\n
    ▸ [Expert Info (Chat/Sequence): GET /wp-content/uploads/2018/11/wafflemakercenter.png HTTP/1.1\r\n]
      Request Method: GET
      Request URI: /wp-content/uploads/2018/11/wafflemakercenter.png
      Request Version: HTTP/1.1
    Host: wafflemakercenter.com\r\n
    User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:82.0) Gecko/20100101 Firefox/82.0\r\n
    Accept: image/webp,*/*\r\n
    Accept-Language: en-US,en;q=0.5\r\n
    Accept-Encoding: gzip, deflate\r\n
    Connection: keep-alive\r\n
    Referer: http://wafflemakercenter.com/\r\n
    \r\n
    [Full request URI: http://wafflemakercenter.com/wp-content/uploads/2018/11/wafflemakercenter.png]
    [HTTP request 2/3]
    [Prev request in frame: 242]
    [Response in frame: 644]
    [Next request in frame: 842]
```

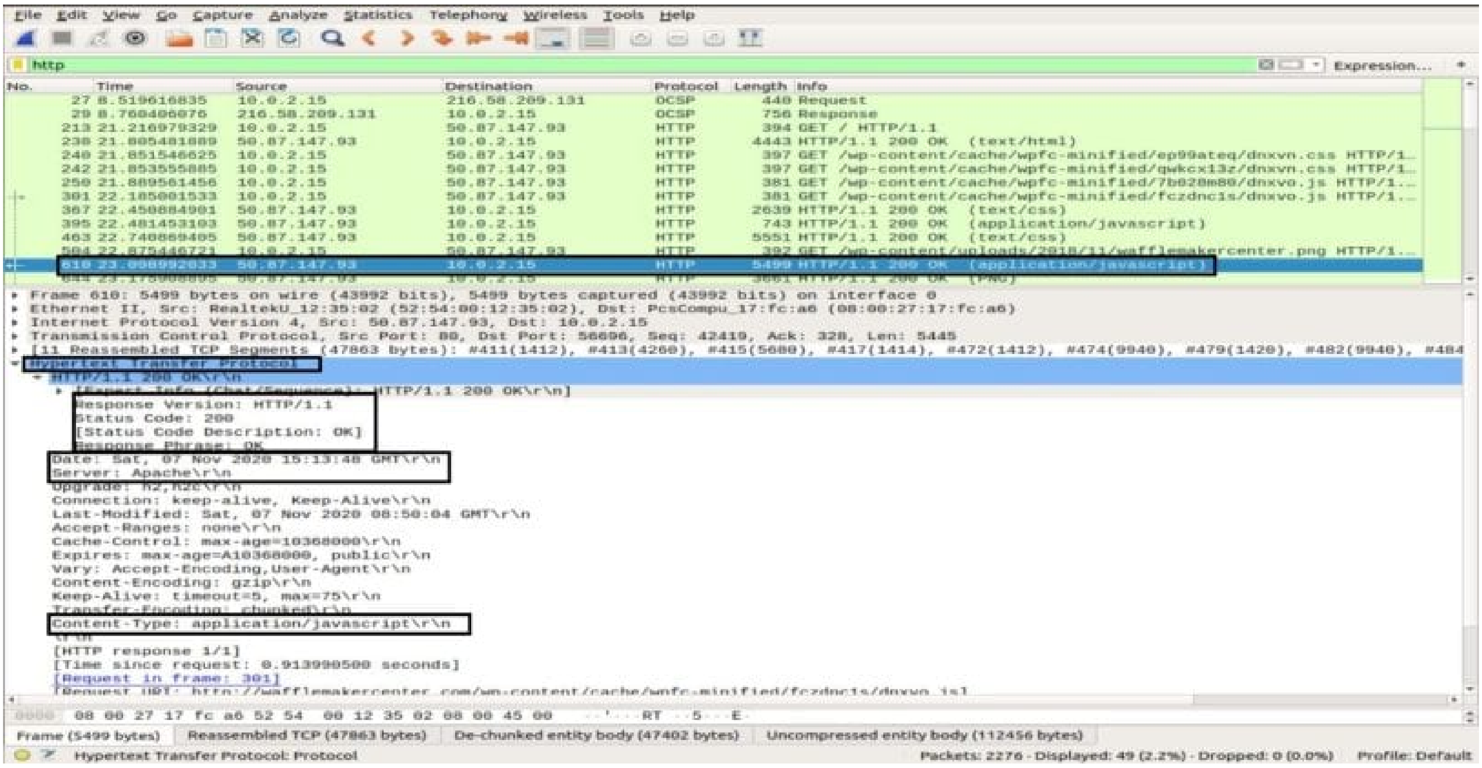
```
0030  f9 9c d3 2f 00 00 47 45  54 20 2f 77 70 2d 63 6f   ···/··GE T /wp-co
0040  6e 74 65 6e 74 2f 75 70  6c 6f 61 64 73 2f 32 30   ntent/up loads/20
```

Hypertext Transfer Protocol (http), 338 bytes

Packets: 2276 · Displayed: 49 (2.2%) · Dropped: 0 (0.0%)   Profile: Default

In the **HTTP OK** packet from server to client, observing the information in the Hypertext Transfer Protocol layer shows “**200 OK**“. This information indicates a normal successful transfer. In the HTTP OK packet, you can observe different headers in comparison to the **HTTP GET** packet. These headers contain information about the requested content.

- **Response Version:** informs about the HTTP version
- **Status Code, Response Phrase:** sent by the server
- **Date:** the time when the server received the HTTP GET packet
- **Server:** server details (Nginx, Apache, etc.)
- **Content-type:** type of content (json, txt/html, etc.)
- **Content-length:** total length of content; our file is 39696 bytes


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
http
Expression…
Frame (5499 bytes)
Reassembled TCP (47863 bytes)
De-chunked entity body (47402 bytes)
Uncompressed entity body (112456 bytes)
Hypertext Transfer Protocol: Protocol
Packets: 2276 · Displayed: 49 (2.2%) · Dropped: 0 (0.0%)
Profile: Default

```
No.     Time           Source          Destination     Protocol Length Info
     27 8.519616835    10.0.2.15       216.58.209.131  OCSP      440 Request
     29 8.760406076    216.58.209.131  10.0.2.15       OCSP      756 Response
    213 21.216979329   10.0.2.15       50.87.147.93    HTTP      394 GET / HTTP/1.1
    238 21.805481889   50.87.147.93    10.0.2.15       HTTP     4443 HTTP/1.1 200 OK  (text/html)
    240 21.851546625   10.0.2.15       50.87.147.93    HTTP      397 GET /wp-content/cache/wpfc-minified/ep99ateq/dnxvn.css HTTP/1…
    242 21.853555885   10.0.2.15       50.87.147.93    HTTP      397 GET /wp-content/cache/wpfc-minified/qwkcx13z/dnxvn.css HTTP/1…
    250 21.889561456   10.0.2.15       50.87.147.93    HTTP      381 GET /wp-content/cache/wpfc-minified/7b028m80/dnxvo.js HTTP/1…
    301 22.185001533   10.0.2.15       50.87.147.93    HTTP      381 GET /wp-content/cache/wpfc-minified/fczdnc1s/dnxvo.js HTTP/1…
    367 22.450884901   50.87.147.93    10.0.2.15       HTTP     2639 HTTP/1.1 200 OK  (text/css)
    395 22.481453103   50.87.147.93    10.0.2.15       HTTP      743 HTTP/1.1 200 OK  (application/javascript)
    463 22.740869405   50.87.147.93    10.0.2.15       HTTP     5551 HTTP/1.1 200 OK  (text/css)
    504 22.875446721   10.0.2.15       50.87.147.93    HTTP      392 GET /wp-content/uploads/2018/11/wafflemakercenter.png HTTP/1…
    610 23.098992033   50.87.147.93    10.0.2.15       HTTP     5499 HTTP/1.1 200 OK  (application/javascript)
    644 23.175908895   50.87.147.93    10.0.2.15       HTTP     3661 HTTP/1.1 200 OK  (PNG)
```

```
▸ Frame 610: 5499 bytes on wire (43992 bits), 5499 bytes captured (43992 bits) on interface 0
▸ Ethernet II, Src: RealtekU_12:35:02 (52:54:00:12:35:02), Dst: PcsCompu_17:fc:a6 (08:00:27:17:fc:a6)
▸ Internet Protocol Version 4, Src: 50.87.147.93, Dst: 10.0.2.15
▸ Transmission Control Protocol, Src Port: 80, Dst Port: 56696, Seq: 42419, Ack: 328, Len: 5445
▸ [11 Reassembled TCP Segments (47863 bytes): #411(1412), #413(4260), #415(5680), #417(1414), #472(1412), #474(9940), #479(1420), #482(9940), #484
▾ Hypertext Transfer Protocol
  ▾ HTTP/1.1 200 OK\r\n
    ▸ [Expert Info (Chat/Sequence): HTTP/1.1 200 OK\r\n]
      Response Version: HTTP/1.1
      Status Code: 200
      [Status Code Description: OK]
      Response Phrase: OK
    Date: Sat, 07 Nov 2020 15:13:48 GMT\r\n
    Server: Apache\r\n
    Upgrade: h2,h2c\r\n
    Connection: keep-alive, Keep-Alive\r\n
    Last-Modified: Sat, 07 Nov 2020 08:50:04 GMT\r\n
    Accept-Ranges: none\r\n
    Cache-Control: max-age=10368000\r\n
    Expires: max-age=A10368000, public\r\n
    Vary: Accept-Encoding,User-Agent\r\n
    Content-Encoding: gzip\r\n
    Keep-Alive: timeout=5, max=75\r\n
    Transfer-Encoding: chunked\r\n
    Content-Type: application/javascript\r\n
    \r\n
    [HTTP response 1/1]
    [Time since request: 0.913990500 seconds]
    [Request in frame: 301]
    [Request URI: http://wafflemakercenter.com/wp-content/cache/wpfc-minified/fczdnc1s/dnxvo.js]
```

```
0000  08 00 27 17 fc a6 52 54  00 12 35 02 08 00 45 00   ··'···RT ··5···E·
```

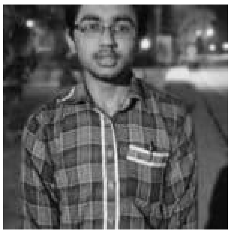
In this section, you have learned how HTTP works and what happens whenever we request content on the web.

Conclusion

Wireshark is the most popular and powerful network sniffer and analysis tool. It is widely used in day-to-day packet analysis tasks in various organizations and institutes. In this article, we have studied some beginner to medium level topics of the Wireshark in Ubuntu. We have learned the type of filters offered by Wireshark for packet analysis. We have covered the network layer model in Wireshark and performed in-depth ICMP and HTTP packet analysis.

However, learning and understanding various aspects of this tool is a long hard journey. Hence, there are a lot of other online lectures and tutorials available to help you around specific topics of Wireshark. You can follow the official user guide available on the Wireshark website. Moreover, once you have built the basic understanding of protocol analysis, it's also advised to use a tool like Varonis that points you at the potential threat and then use Wireshark to investigate for better understanding.

About the author



A security enthusiast who loves Terminal and Open Source. My area of expertise is Python, Linux (Debian), Bash, Penetration testing, and Firewalls. I’m born and raised in Wazirabad, Pakistan and currently doing Undergraduation from National University of Science and Technology (NUST). On Twitter i go by @UsamaAzad14

PDF's generated at: Wed Sep 01 2021 08:08:25 GMT+0000 (Coordinated Universal Time)

Source: https://linuxhint.com/wireshark-in-depth-tutorial/