WIRESHARK

CPSC 441 - Tutorial 5

Winter 2018

UNIVERSITY OF CALGARY

WHAT IS **WIRESHARK**?

- **Wireshark** is a free and open source packet analyzer
- It is used for network troubleshooting, analysis, software and communication protocol development, and education.
- Originally named **Ethereal**, the project was renamed Wireshark in May 2006 due to trademark issues

FUNCTIONALITY

- Wireshark is very similar to tcpdump, but has a graphical front-end, and some integrated sorting and filtering options
- Data can be captured from a live network connection or read from a file of already-captured packets.
- Live data can be read from different types of networks, including Ethernet, IEEE 802.11, PPP, and loopback.
- Captured network data can be browsed via a GUI, or via the terminal version of the utility, TShark
- Data display can be refined using a display filter
- Wireless connections can also be filtered as long as they traverse the monitored Ethernet
- Various settings, timers, and filters can be set to provide the facility of filtering the output of the captured traffic

INSTALLATION

- Download Wireshark:
  http://www.wireshark.org/download.html

  Choose the appropriate version according to your operating system
  For Windows, during the installation, agree to install WinPcap

- There is a good tutorial on how to capture data using WireShark:
  http://wiki.wireshark.org/CaptureSetup

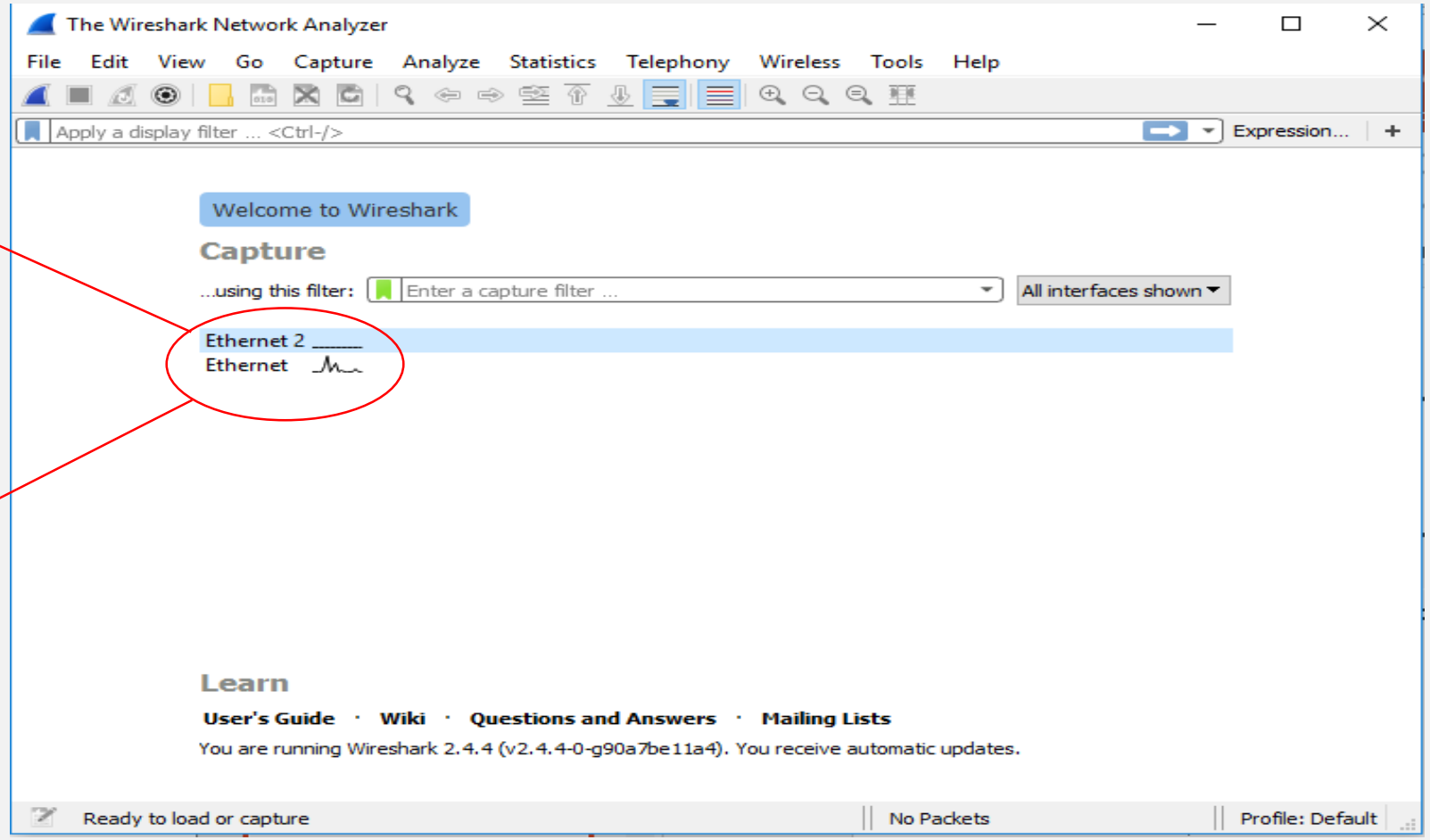
BEFORE CAPTURING

- Are you allowed to do this?
  - Ensure that you have permission to capture packets from the network you are connected with
  - Corporate policies or applicable laws may prohibit capturing data from the network

- General Setup
  - Operating system must support packet capturing, e.g. capture support is enabled
  - You must have sufficient privileges to capture packets, e.g. root / administrator privileges
  - Your computer's time and time zone settings should be correct

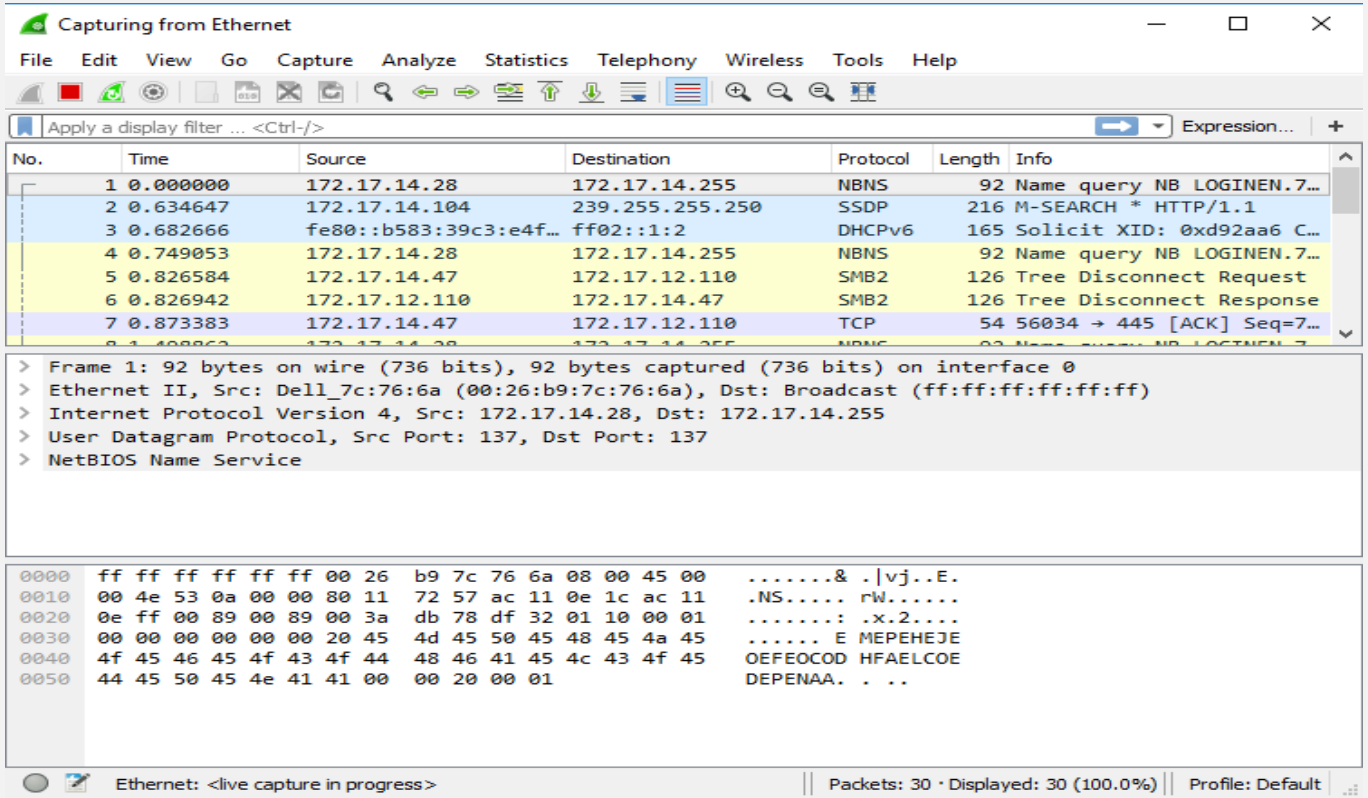
Available Interfaces

Choose the one with traffic



START CAPTURING PACKETS

- After clicking on desired interface, Wireshark starts capturing packets



ANALYZE CAPTURED PACKETS

Time of capturing packet

Source IP

Destination IP

Short description of packet

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	172.17.14.28	172.17.14.255	NBNS	92	Name query NB LOGINEN.7…
2	0.634647	172.17.14.104	239.255.255.250	SSDP	216	M-SEARCH * HTTP/1.1
3	0.682666	fe80::b583:39c3:e4f…	ff02::1:2	DHCPv6	165	Solicit XID: 0xd92aa6 C…
4	0.749053	172.17.14.28	172.17.14.255	NBNS	92	Name query NB LOGINEN.7…
5	0.826584	172.17.14.47	172.17.12.110	SMB2	126	Tree Disconnect Request
6	0.826942	172.17.12.110	172.17.14.47	SMB2	126	Tree Disconnect Response
7	0.873383	172.17.14.47	172.17.12.110	TCP	54	56034 → 445 [ACK] Seq=7…

ANALYZE CAPTURED PACKETS

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	172.17.14.28	172.17.14.255	NBNS	92	Name query NB LOGINEN.7…
2	0.634647	172.17.14.104	239.255.255.250	SSDP	216	M-SEARCH * HTTP/1.1
3	0.682666	fe80::b583:39c3:e4f…	ff02::1:2	DHCPv6	165	Solicit XID: 0xd92aa6 C…
4	0.749053	172.17.14.28	172.17.14.255	NBNS	92	Name query NB LOGINEN.7…
5	0.826584	172.17.14.47	172.17.12.110	SMB2	126	Tree Disconnect Request
6	0.826942	172.17.12.110	172.17.14.47	SMB2	126	Tree Disconnect Response
7	0.873383	172.17.14.47	172.17.12.110	TCP	54	56034 → 445 [ACK] Seq=7…

```
> Frame 7: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface 0
> Ethernet II, Src: Dell_dc:22:2d (34:17:eb:dc:22:2d), Dst: HewlettP_9d:62:00 (58:20:b1:9d:62:00)
> Internet Protocol Version 4, Src: 172.17.14.47, Dst: 172.17.12.110
> Transmission Control Protocol, Src Port: 56034, Dst Port: 445, Seq: 73, Ack: 73, Len: 0
```

Hierarchical View:

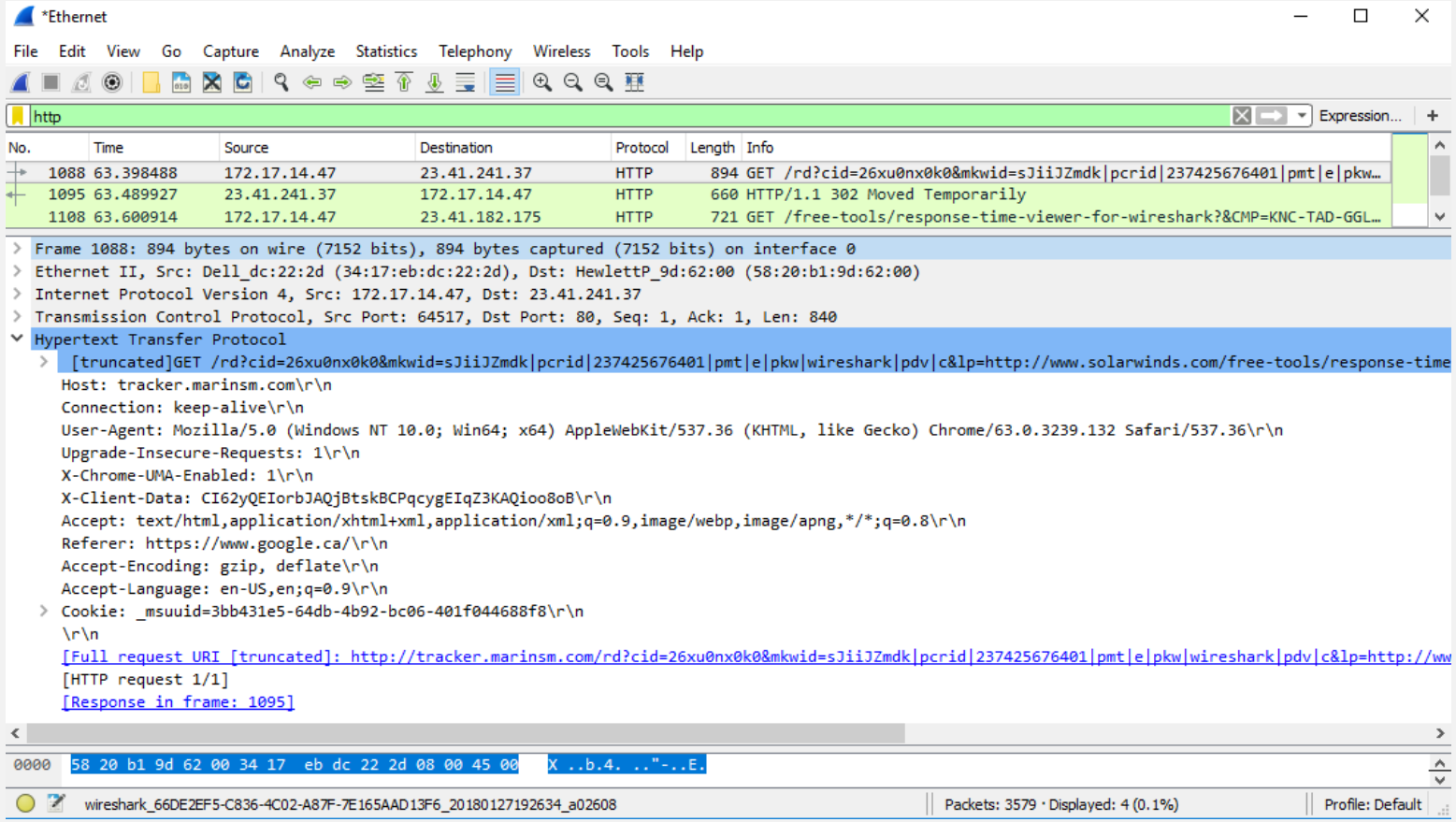
Frame

Ethernet

IP

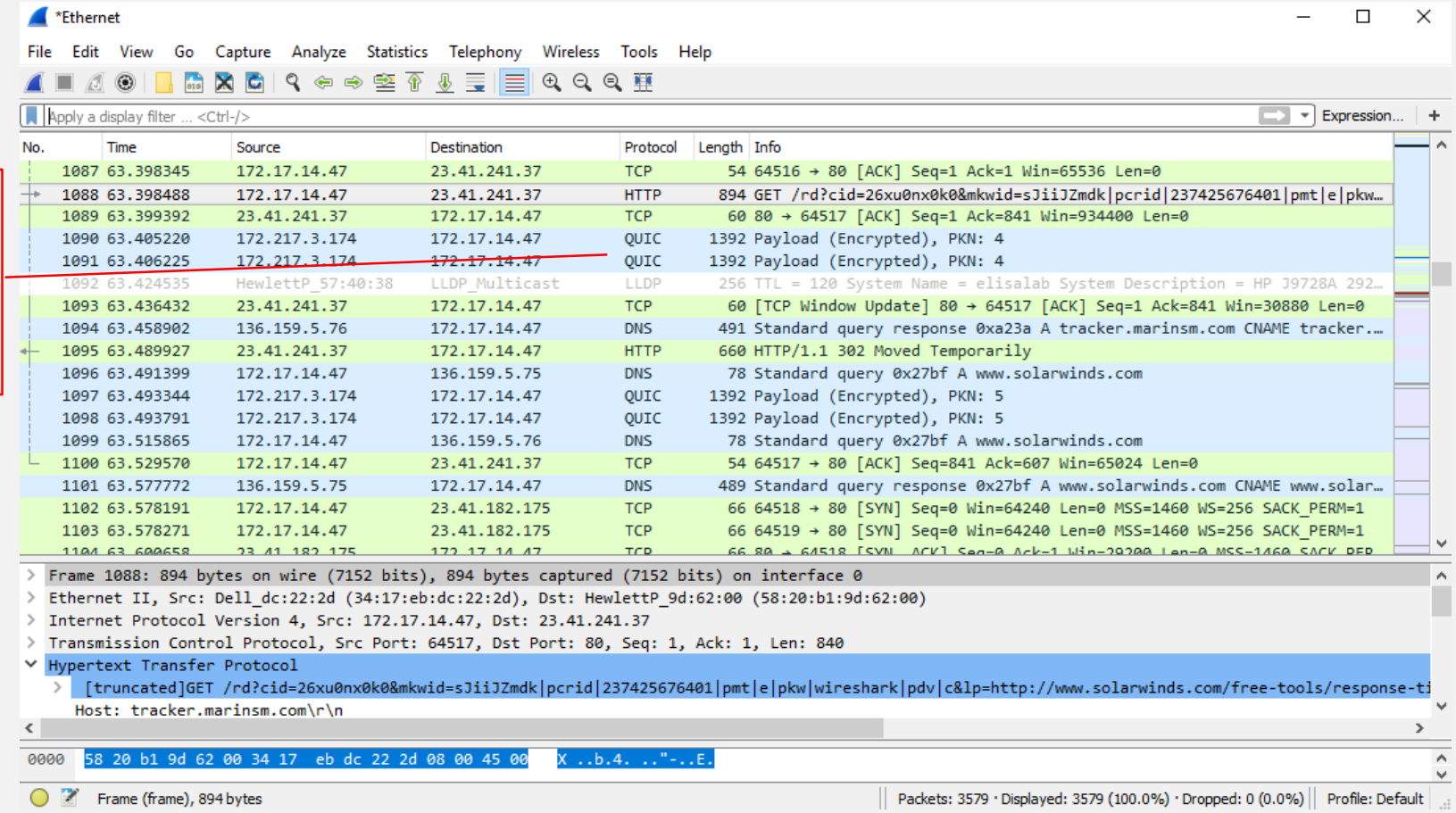
TCP

ANALYZE A HTTP REQUEST



PACKETS WITH STRANGE PROTOCOLS

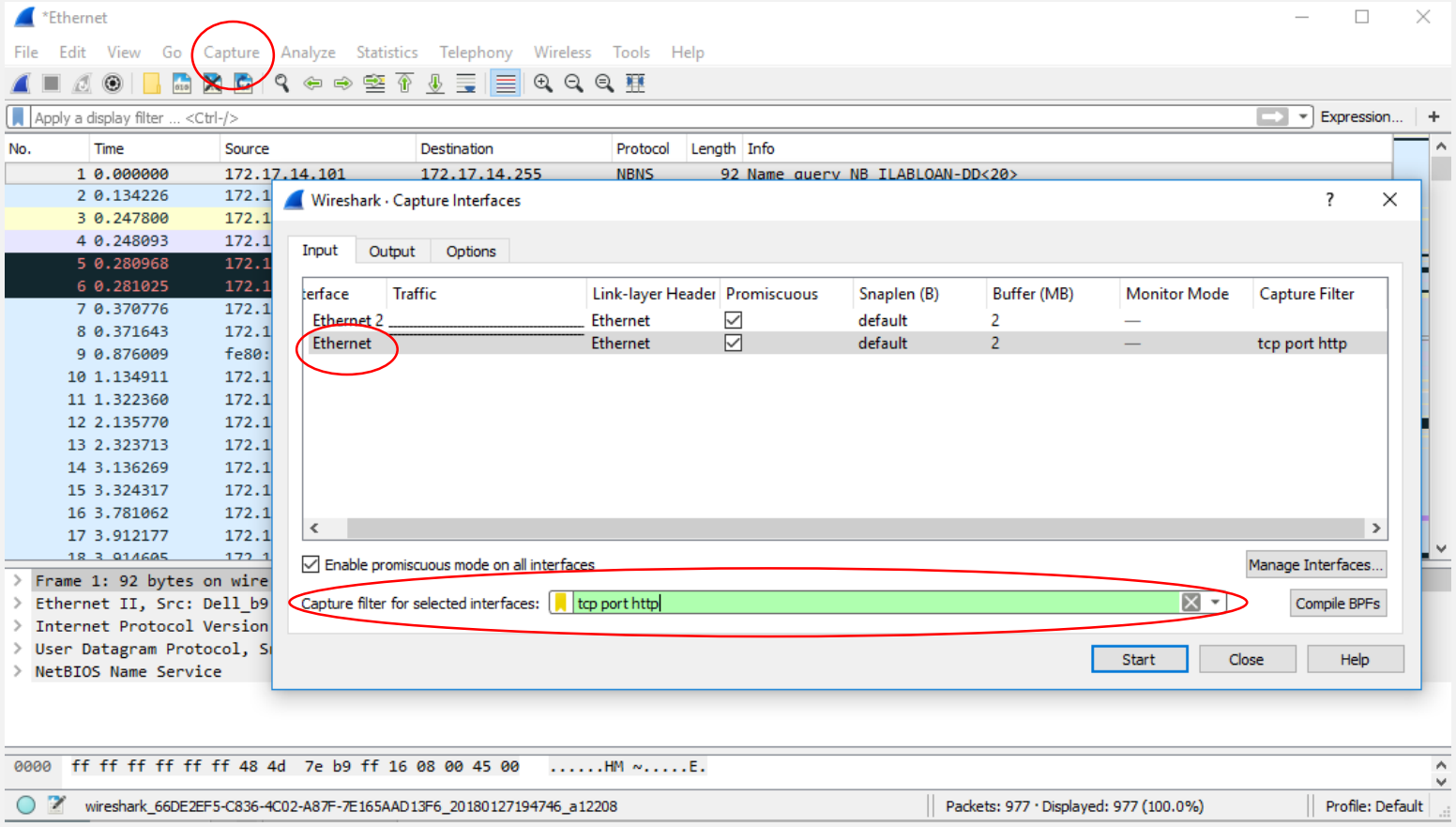
QUIC: Quick UDP Internet Connections



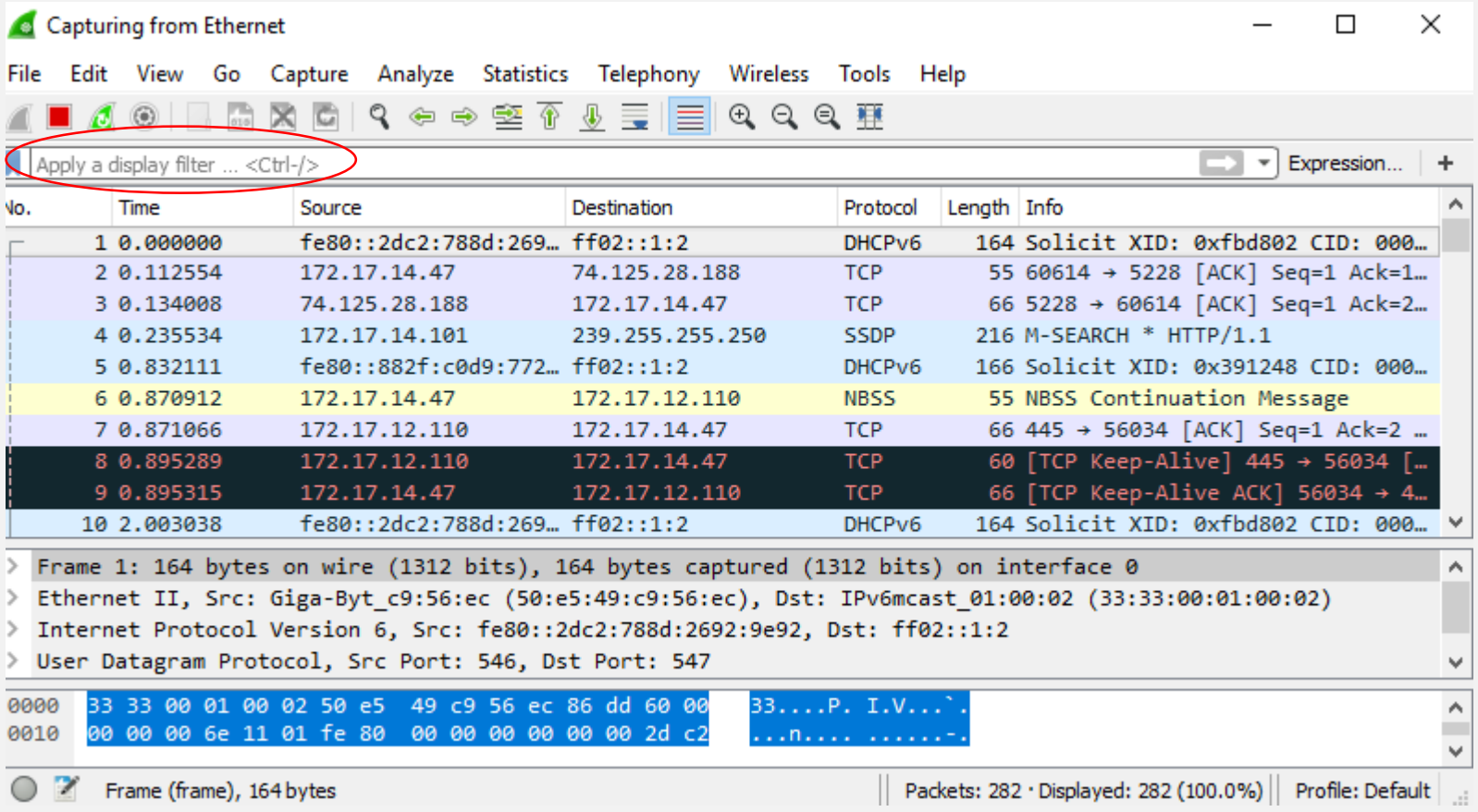
WIRESHARK FILTERS

- Wireshark has two types of filters:
  - Capture Filters
    - A powerful capture filter engine helps **remove unwanted packets** from a packet trace and only retrieve the packets of interest
  - Display Filters
    - Let you compare the fields within a protocol against a specific value, compare fields against other fields, and check the existence of specified fields or protocols

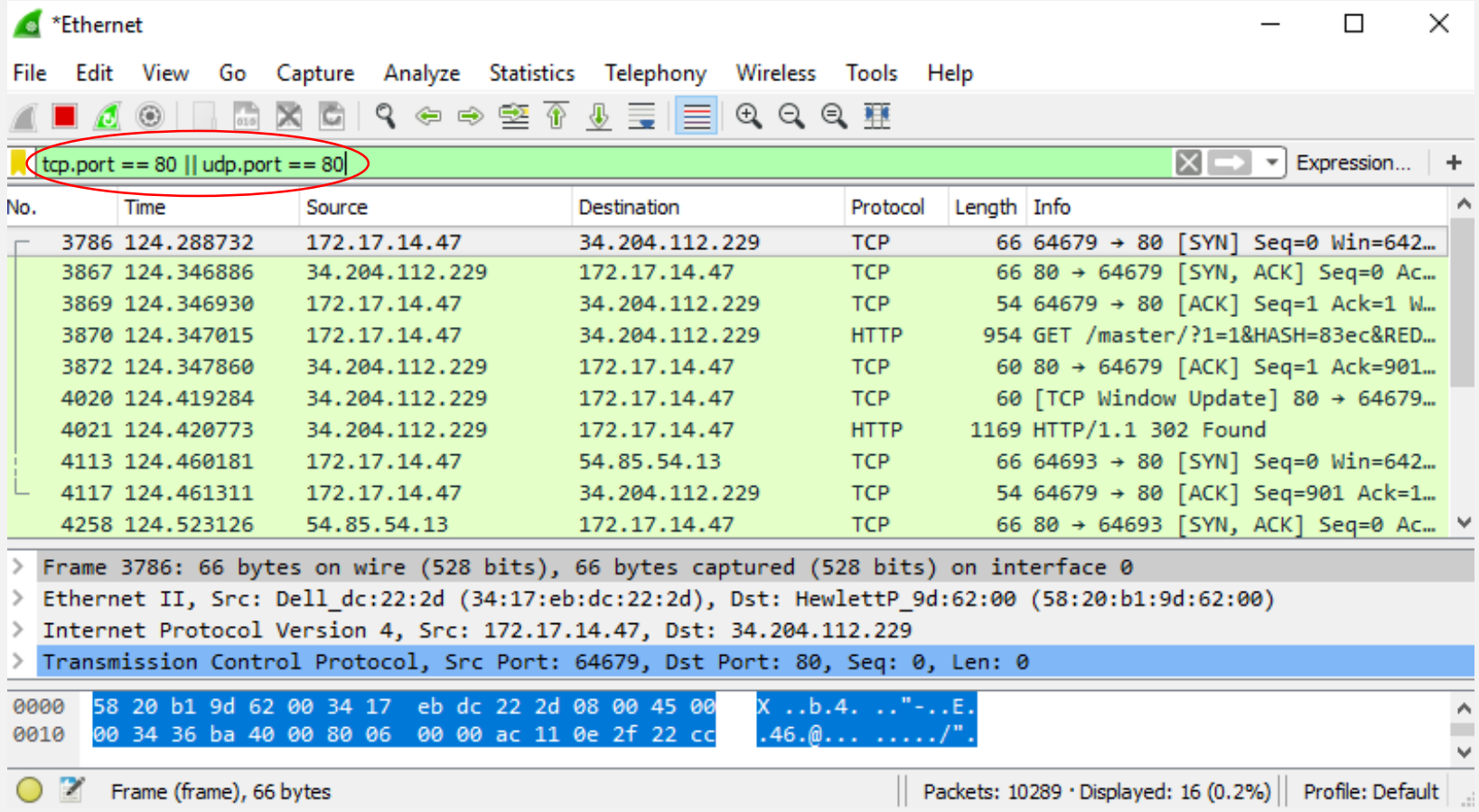
CAPTURE FILTER



DISPLAY FILTER



DISPLAY FILTER EXAMPLE



FILTERS: COMPARISON OPERATORS AND LOGICAL EXPRESSIONS

The comparison operators can be expressed either through English-like abbreviations or through C-like symbols:

- eq, == Equal
- ne, != Not Equal
- gt, > Greater Than
- lt, < Less Than
- ge, >= Greater than or Equal to
- le, <= Less than or Equal to

Logical:

- and, && Logical AND
- or, || Logical OR
- not, ! Logical NOT

In display Filter:

- tcp.port == 80 (tcp.port eq 80)
- eth.addr == 00:00:5e:00:53:00
- tcp.port == 80 || udp.port == 80
- tcp.port == 80 && ip.src == 172.17.14.47
- http.request.version=="HTTP/1.1"
- tcp.dstport == 25

In capture filter:

- tcp port 80
- ip src host 136.159.5.20
- host 136.159.5.1 (source/destination)
- (src host 23.36.178.81 and not dst host 172.17.14.47) and tcp dst portrange 200-10000

UNIVERSITY OF CALGARY

FILTERS: SLICE OPERATOR

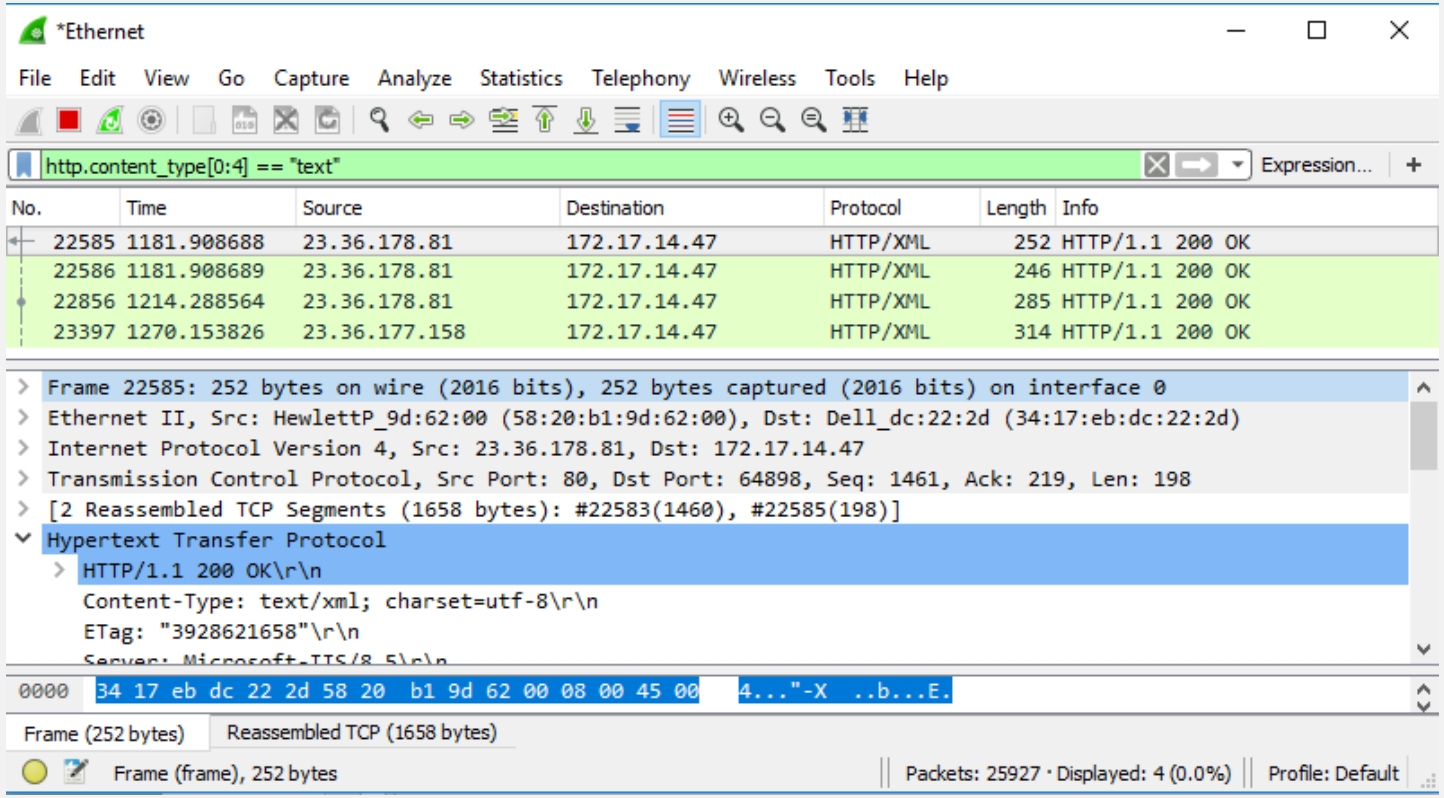
- You can take a slice of a field if the field is a text string or a byte array

  For example, you can filter the HTTP header fields with the header "location" which indicates that redirection happens:

  `http.location[0:12]==="http://pages"`

  Another example:

  `http.content_type[0:4] == "text"`



REFERENCES

- https://en.wikipedia.org/wiki/Wireshark
- https://wiki.wireshark.org/
- https://www.wireshark.org/