

CN-Basic

L11/12

Networking tools

Dr. Ram P Rustagi
rprustagi@ksit.edu.in

<http://www.rprustagi.com>

<https://www.youtube.com/watch?v=ke6maEyUdKo>

<https://www.youtube.com/watch?v=R37eqQCr08w>

Tools - nc

- nc (netcat)
 - Works as both layer client & server
 - Supports both TCP and UDP
 - Supports both IPv4 and IPv6
 - Common use
 - Simple TCP/UDP based data transfer
 - Shell script based HTTP clients and servers
 - Network daemon testing
 - SOCKS or HTTP ProxyCommand for ssh

Tools - nc

- **nc usage**
 - `-l` acting as server
 - `-u` use UDP
 - `-i` for interval based transmission (lines)
 - `-k` for keeping server up after connection close
- **Examples**
 - `nc <servername> <server port> # client`
 - `nc -l port # server (add -p on windows)`
 - **To transfer files**
 - **Server:** `nc -l <port> >file.dat`
 - **Client:** `cat <file> | nc <servername> <port>`

Usage: nc

- Terminating connection after idle time

```
nc -w 10 <server> <port> # timeout 10s
```

- Don't use with server option
- Accessing a web server
 - nc -C mywww.com 80
 - ...

Tools - wget

- **wget:** `wget [options] <url>`
 - **Options**
 - `-d` # to debug headers
 - `-O <file>` # to save with different name
 - `-i <urlsfile>` # list of urls in a file
 - `-c` # to resume to broken download
 - `-b` # run in background
 - `--limit-rate=500k`
 - `--header=<headers>`
 - `--http-user=user --http-password=..`
 - `-mk` # mirroring with local link conversion
 - ...

Tools - curl

- General invocation, displays content

```
curl <url>
```

- To specify specific headers

```
curl -H <header> <urls>
```

- To dump HTTP headers

```
curl -D <file> <url>
```

Tools - ping

- `ping` (Packet Inter Network Groper)
 - Checking reachability
 - `ping hostname`
 - `-i` changing packet interval
 - `-c` packet count
 - `-f` flooding the network
 - `-q` quiet mode
 - `-s` change packet size
 - `-W` response timeout

Usage: ping

```
$ ping -c 10 www.google.com
```

```
PING www.google.com (74.125.203.104): 56 data bytes
```

```
64 bytes from 74.125.203.104: icmp_seq=0 ttl=47 time=93.346 ms
64 bytes from 74.125.203.104: icmp_seq=1 ttl=47 time=94.300 ms
64 bytes from 74.125.203.104: icmp_seq=2 ttl=47 time=99.392 ms
64 bytes from 74.125.203.104: icmp_seq=3 ttl=47 time=140.457 ms
64 bytes from 74.125.203.104: icmp_seq=4 ttl=47 time=168.882 ms
64 bytes from 74.125.203.104: icmp_seq=5 ttl=47 time=218.813 ms
64 bytes from 74.125.203.104: icmp_seq=6 ttl=47 time=270.833 ms
64 bytes from 74.125.203.104: icmp_seq=7 ttl=47 time=312.594 ms
64 bytes from 74.125.203.104: icmp_seq=8 ttl=47 time=263.280 ms
64 bytes from 74.125.203.104: icmp_seq=9 ttl=47 time=309.129 ms
```

```
--- www.google.com ping statistics ---
```

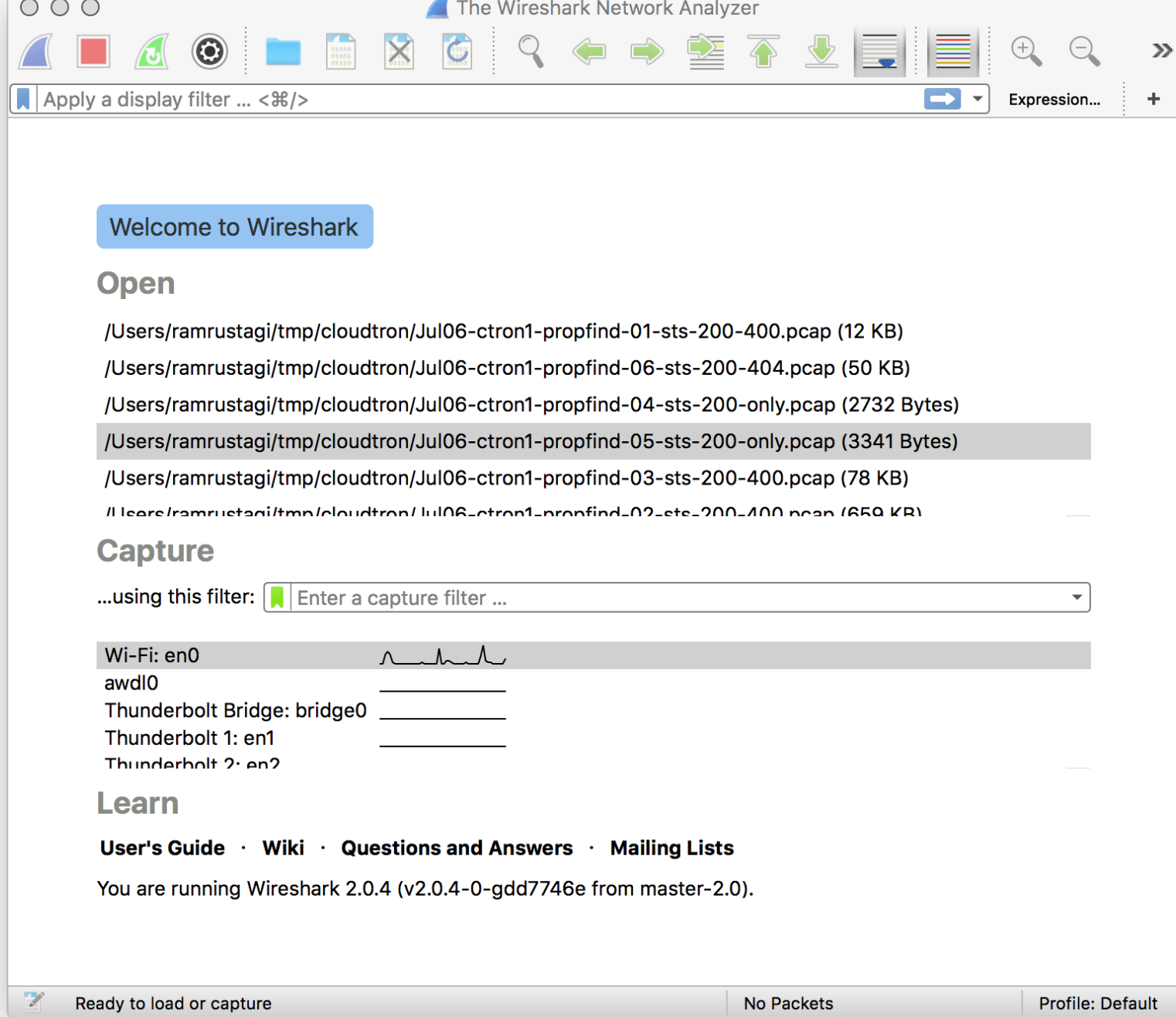
```
10 packets transmitted, 10 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 93.346/197.103/312.594/84.298 ms
```

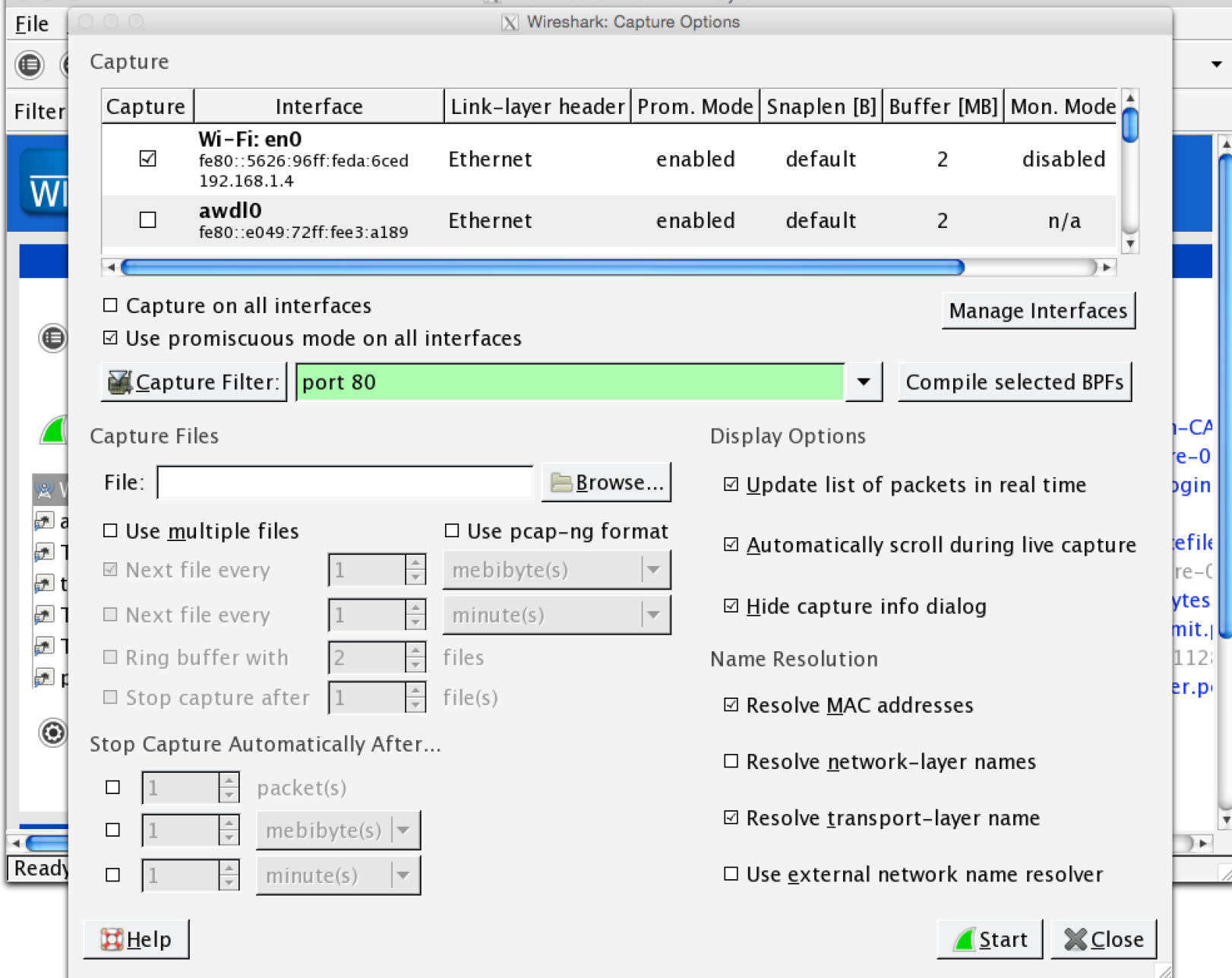

Tools - traceroute

- traceroute (Windows: `tracert`)
 - Checking reachability
 - `traceroute hostname`
 - ??

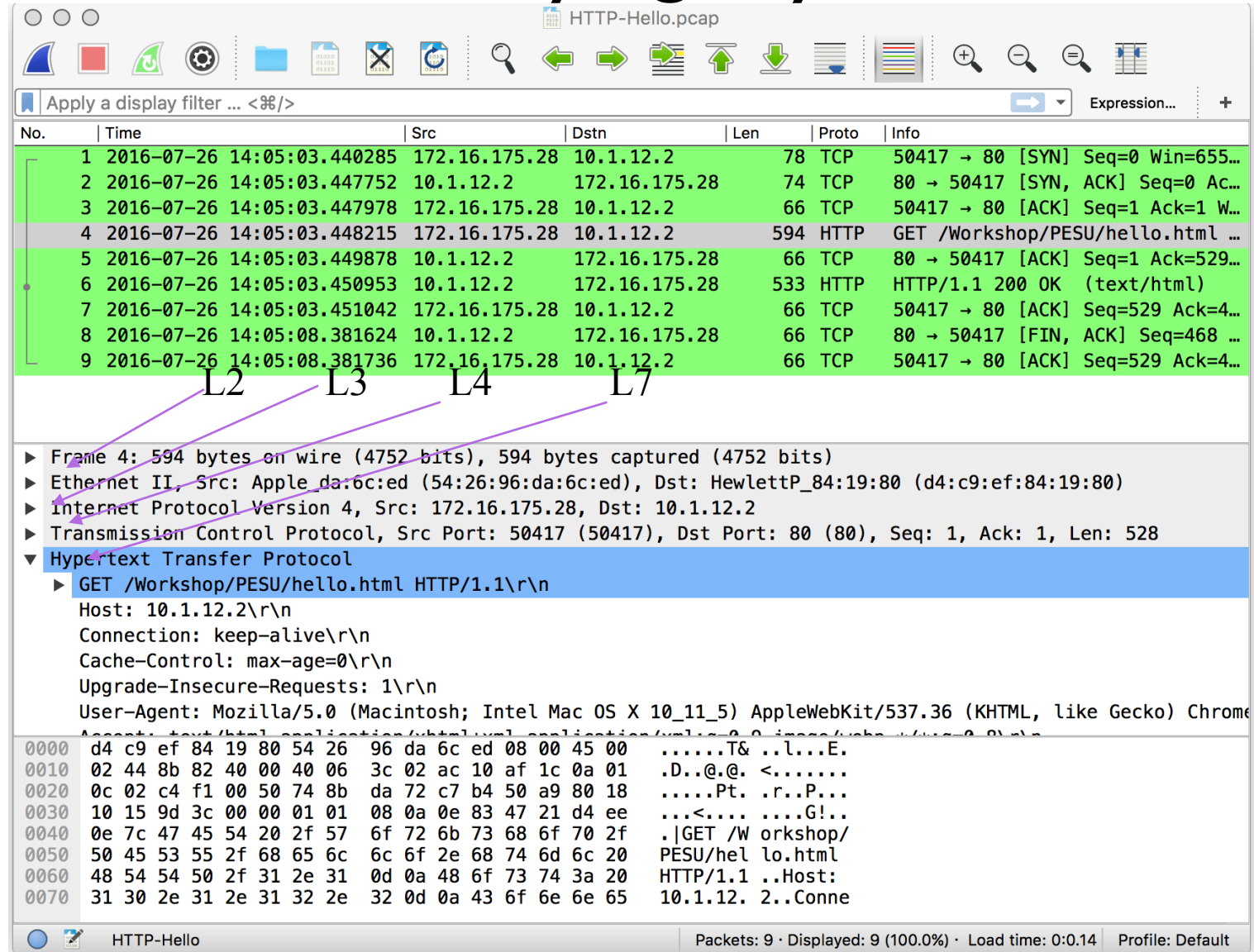
Tools

- `wireshark`
 - https://www.wireshark.org/docs/wsug_html_chunked/
 - Enables viewing of bytes sent/recd on network
 - Capture and Display filters
 - Graphical, built on `tcpdump`
 - TCP session display
 - Changing UI options
- `tcpdump`: command line capture tool
 - `-o` output file
 - `-c` packet count
 - `-i` interface names





Identifying Layers



HTTP-Hello.pcap

Apply a display filter ... <%%/> Expression...

No.	Time	Src	Dstn	Len	Proto	Info
1	2016-07-26 14:05:03.440285	172.16.175.28	10.1.12.2	78	TCP	50417 → 80 [SYN] Seq=0 Win=655...
2	2016-07-26 14:05:03.447752	10.1.12.2	172.16.175.28	74	TCP	80 → 50417 [SYN, ACK] Seq=0 Ac...
3	2016-07-26 14:05:03.447978	172.16.175.28	10.1.12.2	66	TCP	50417 → 80 [ACK] Seq=1 Ack=1 W...
4	2016-07-26 14:05:03.448215	172.16.175.28	10.1.12.2	594	HTTP	GET /Workshop/PESU/hello.html ...
5	2016-07-26 14:05:03.449878	10.1.12.2	172.16.175.28	66	TCP	80 → 50417 [ACK] Seq=1 Ack=529...
6	2016-07-26 14:05:03.450953	10.1.12.2	172.16.175.28	533	HTTP	HTTP/1.1 200 OK (text/html)
7	2016-07-26 14:05:03.451042	172.16.175.28	10.1.12.2	66	TCP	50417 → 80 [ACK] Seq=529 Ack=4...
8	2016-07-26 14:05:08.381624	10.1.12.2	172.16.175.28	66	TCP	80 → 50417 [FIN, ACK] Seq=468 ...
9	2016-07-26 14:05:08.381736	172.16.175.28	10.1.12.2	66	TCP	50417 → 80 [ACK] Seq=529 Ack=4...

L2 L3 L4 L7

- ▶ Frame 4: 594 bytes on wire (4752 bits), 594 bytes captured (4752 bits)
- ▶ Ethernet II, Src: Apple da:6c:ed (54:26:96:da:6c:ed), Dst: HewlettP_84:19:80 (d4:c9:ef:84:19:80)
- ▶ Internet Protocol Version 4, Src: 172.16.175.28, Dst: 10.1.12.2
- ▶ Transmission Control Protocol, Src Port: 50417 (50417), Dst Port: 80 (80), Seq: 1, Ack: 1, Len: 528
- ▼ Hypertext Transfer Protocol
 - ▶ GET /Workshop/PESU/hello.html HTTP/1.1\r\nHost: 10.1.12.2\r\nConnection: keep-alive\r\nCache-Control: max-age=0\r\nUpgrade-Insecure-Requests: 1\r\nUser-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome...

0000 d4 c9 ef 84 19 80 54 26 96 da 6c ed 08 00 45 00T& ..l...E.
0010 02 44 8b 82 40 00 40 06 3c 02 ac 10 af 1c 0a 01 ..D..@.@. <.....
0020 0c 02 c4 f1 00 50 74 8b da 72 c7 b4 50 a9 80 18Pt. .r..P...
0030 10 15 9d 3c 00 00 01 01 08 0a 0e 83 47 21 d4 ee ...<.... ..G!..
0040 0e 7c 47 45 54 20 2f 57 6f 72 6b 73 68 6f 70 2f .|GET /W orkshop/
0050 50 45 53 55 2f 68 65 6c 6c 6f 2e 68 74 6d 6c 20 PESU/hel lo.html
0060 48 54 54 50 2f 31 2e 31 0d 0a 48 6f 73 74 3a 20 HTTP/1.1 ..Host:
0070 31 30 2e 31 2e 31 32 2e 32 0d 0a 43 6f 6e 6e 65 10.1.12. 2..Conne

HTTP-Hello Packets: 9 · Displayed: 9 (100.0%) · Load time: 0:0:14 Profile: Default

Wireshark: UI Options

- Color coding
- Time format
- Packet reordering (in display)
- Defining protocol
- Using display filter
- Following TCP Stream
- Capture packets with proper capture filter
 - Needed for analyzing packet layers

Wireshark capture filters

- **Traffic between A and either B or C**
`host A and \ (B or C \)`
- **Traffic between A any host except B**
`host A and not B`
- **Capture web traffic**
`port 80`
- **Capture ICMP traffic (e.g. ping, traceroute)**
`icmp`
- **Capture web traffic**
`port 80`
- **Capture traffic for a subnet**
`net 10.211.55.0/24`

Wireshark Display Filters

- **Source IP filter**
 - `ip.src == 10.1.1.1`
- **Destination IP filter**
 - `ip.dst == 10.1.1.101`
 - `ip.dst != 10.1.1.101`
- **Protocol filter**
 - `http || icmp`
- **Port number**
 - `tcp.port eq 80`
- **TCP Seq**
 - `tcp.stream eq 1`

Wireshark - Misc

- Saving file
 - Saving selected packets
- Reading from file
- Time display format
- Statistics
- Other options

tcpdump

- Command line interface
 - ASCII content
 - Capture full packet
 - Capture filters
 - Output file
 - Ethernet frame display

tcpdump - Usage Examples

- To capture between two specified hosts and save

```
$ sudo tcpdump -n -i eth0 -s 0 -w  
file.pcap host <A> and host <B>
```

- To capture 100 packets received from host X

```
$ sudo tcpdump -n -i eth0 -c 100  
src <X>
```

- To capture first 256 bytes and display in ASCII

```
$ sudo tcpdump -n -i eth0 -s 256 -  
A
```

- To capture with (link level) ethernet headers

```
$ sudo tcpdump -n -i eth0 -e
```

Use Case 1 : Using nc

- Take two systems in your home/lab and connect to a network. These can be connected by directly connecting them via a LAN Cable (ethernet). If LAN cable is an issue, Enable mobile hotspot on your phone and connect the laptops to your mobile hotspot.
- If windows machine, install “netcat for windows”
 - (<https://joncraton.org/blog/46/netcat-for-windows/>)
 - It is available by default on Linux/Mac.
- Identify the IP address of each system.
- Establish a chat session between and exchange few messages.

Use Case 2 : Using ping

- Connect your in your home/lab to internet.
- Check reachability of google.com by sending 10 pkts
 - ping -c10 www.google.com (linux/mac)
 - ping -n 10 (www.google.com) (windows)
- Analyze the response time (min, max, avg and standard deviation).
- Repeat the above with 100 pkts with pkt size of 1000 bytes and analyze the response.
- Repeat the above 200 pkts, break down your internet connectivity for 30 seconds after 50 pkts
 - Analyze the packet loss and response time.

Use Case 3 : Using `tracert`

- Connect your in your home/lab to internet.
- Traceoute google.com and identify all the routers in the path your home to google
- Analyze the response times of various routers.
 - Do you find any router at n^{th} hop which takes more time than router at $(n+1)^{\text{th}}$ hop
- Repeat the above for your other favourite websites.

Use Case 4 : Using `wget`/`curl`

- use `wget` to download a file using rate limit.
- Use `curl` to dump headers of a URL.

Use Case 5 : Using wireshark

- Download and install wireshark on your system (www.wireshark.org).
- Launch wireshark application, select the active internet interface (e.g. ethernet, wireless etc or even any) and start capturing with any capture filter.
 - Access your favourite website e.g. your academic institution website in the browser.
 - Stop the capture after 1 minute in wireshark.
- Analyze the number of packets received.
- Identify all the protocols that are identified by wireshark

Use Case 6 : Using `wireshark`

- Connect two systems on the network and identify their IP addresses.
- Launch `wireshark` application and specify capture filter as other system's IP address.
- Initiate a chat session (using `nc`) between two systems and exchange few messages.
- Analyze the `nc` chat packets in `wireshark`.
- Analyze the number of packets received.
- Explore why there is no layer 7 packet in `wireshark` even though you are using `nc` as an application.

Use Case 7 : Using `tcpdump`

- Connect two systems on the network and identify their IP addresses.
- Launch `tcpdump` application with capture filter as IP address of your college website with packet size of 1500
- Open a browser and access your college website.
- Analyze the first 5 packets in `tcpdump`.
 - Analyze 3-way handshake
 - Analyze application data sent
 - Analyze response received

Summary

- nc
- ping
- traceroute
- wget
- curl
- tcpdump
- wireshark