



Network 101: Making The Connection

An Illustrated Walk Through of Network Traffic

Trevor DeCamp (@TheDerpySage)

About Me

- 2021 AAS Cyber Security from DMACC
- Systems Administrator at Iowa State University
- Combined 8+ Years of IT Experience
- Compulsive Home Labber (VoIP, Email, etc)
- Returning to School at ISU
- Punk Rock/Italian American Upbringing



www.thederpysage.com

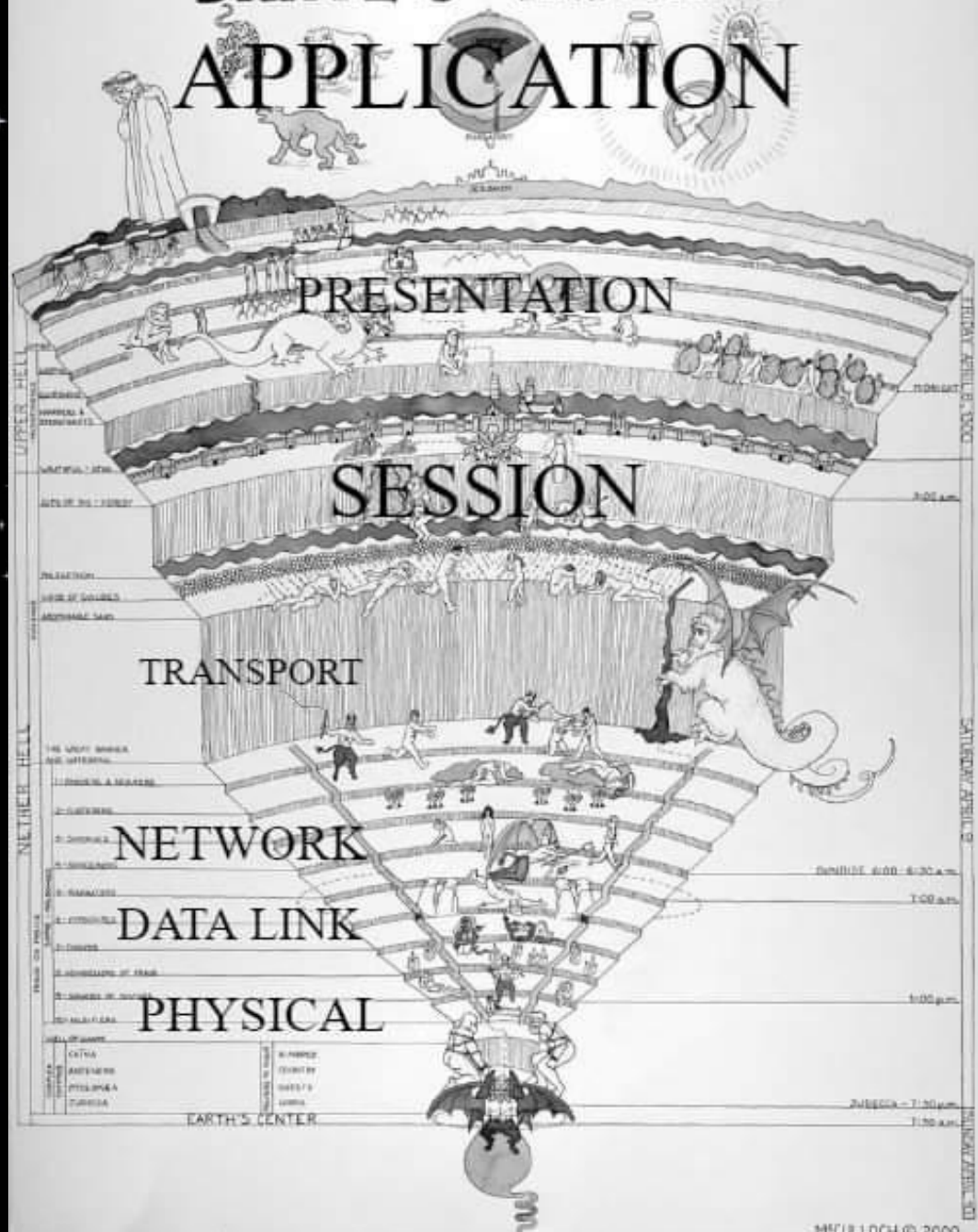
What we're talking about today...

- Network Traffic (duh)
- “Why can't my computer connect to the internet?”
- “What does a connection look like?”
- “How do I know someone can't read my connection?”

What is this?

Frame 923: 465 bytes on wire (3720 bits), 465 bytes captured (3720 bits) on interface eth0, id 0
Ethernet II, Src: VMware_41:6b:4e (00:0c:29:41:6b:4e), Dst: VMware_b8:c6:d8 (00:0c:29:b8:c6:d8)
Internet Protocol Version 4, Src: 192.168.99.12, Dst: 192.168.99.11
Transmission Control Protocol, Src Port: 80, Dst Port: 49861, Seq: 1, Ack: 429, Len: 411
Hypertext Transfer Protocol, HTTP/1.1 200 OK\r\n
Line-based text data: text/html (13 lines)

DANTE'S INFERNO APPLICATION



OSI model

Layer		Protocol data unit (PDU)	Function ^[26]	
Host layers	7	Application	Data	High-level protocols such as for resource sharing or remote file access, e.g. HTTP .
	6	Presentation		Translation of data between a networking service and an application; including character encoding , data compression and encryption/decryption
	5	Session		Managing communication sessions , i.e., continuous exchange of information in the form of multiple back-and-forth transmissions between two nodes
	4	Transport	Segment, Datagram	Reliable transmission of data segments between points on a network, including segmentation , acknowledgement and multiplexing
Media layers	3	Network	Packet	Structuring and managing a multi-node network, including addressing , routing and traffic control
	2	Data link	Frame	Transmission of data frames between two nodes connected by a physical layer
	1	Physical	Bit, Symbol	Transmission and reception of raw bit streams over a physical medium

OSI model

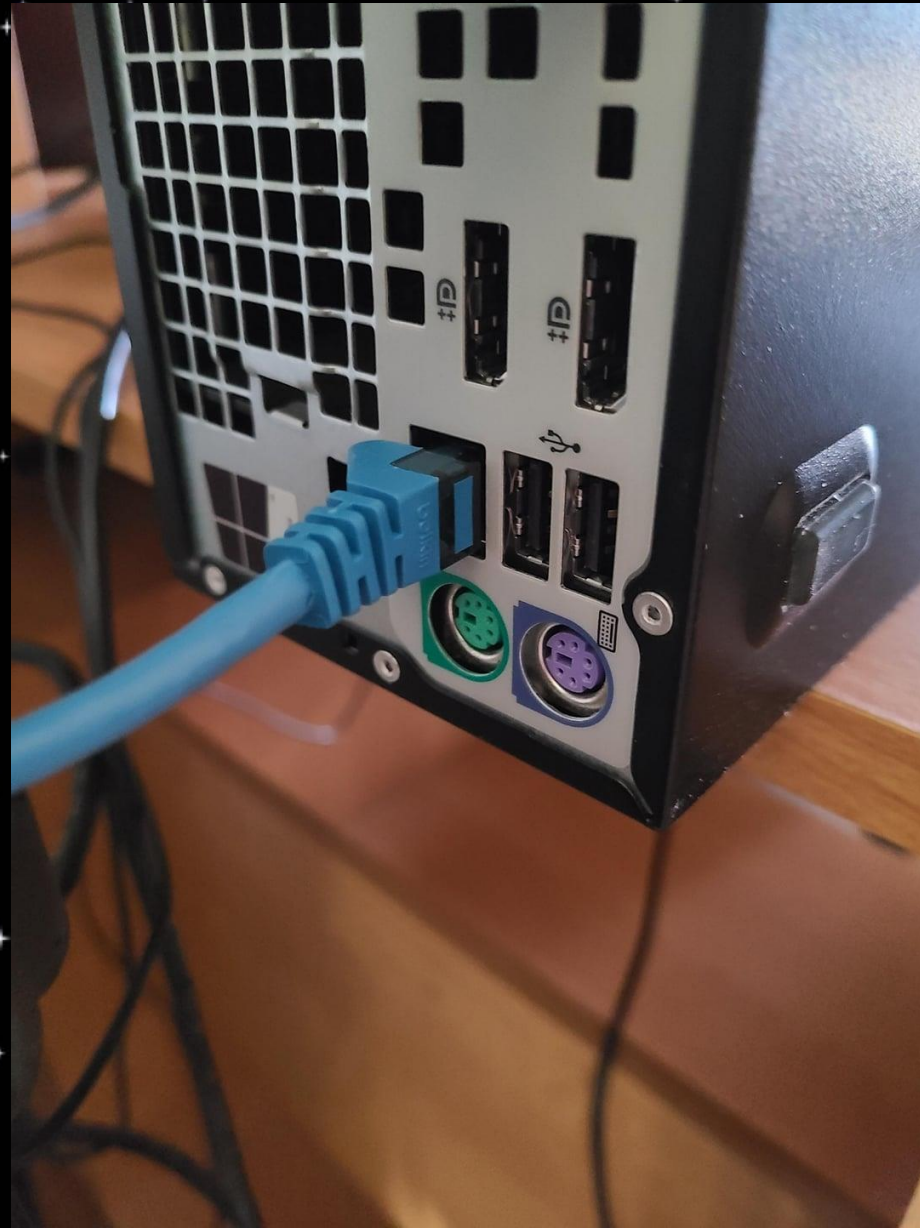
Layer		Protocol data unit (PDU)	Function ^[26]	
Host layers	7	Application	High-level protocols such as for resource sharing or remote file access, e.g. HTTP .	
	6	Presentation	Data	
	5	Session		Translation of data between a networking service and an application; including character encoding , data compression and encryption/decryption
	4	Transport		Managing communication sessions , i.e., continuous exchange of information in the form of multiple back-and-forth transmissions between two nodes
Media layers	3	Network	Reliable transmission of data segments between points on a network, including segmentation , acknowledgement and multiplexing	
	2	Data link	Structuring and managing a multi-node network, including addressing , routing and traffic control	
	1	Physical	Transmission of data frames between two nodes connected by a physical layer	
		Bit, Symbol	Transmission and reception of raw bit streams over a physical medium	

Frame 923: 465 bytes on wire (3720 bits), 465 bytes captured (3720 bits) on interface eth0, id 0
 Ethernet II, Src: VMware_41:6b:4e (00:0c:29:41:6b:4e), Dst: VMware_b8:c6:d8 (00:0c:29:b8:c6:d8)
 Internet Protocol Version 4, Src: 192.168.99.12, Dst: 192.168.99.11
 Transmission Control Protocol, Src Port: 80, Dst Port: 49861, Seq: 1, Ack: 429, Len: 411
 Hypertext Transfer Protocol, HTTP/1.1 200 OK\r\n
 Line-based text data: text/html (13 lines)

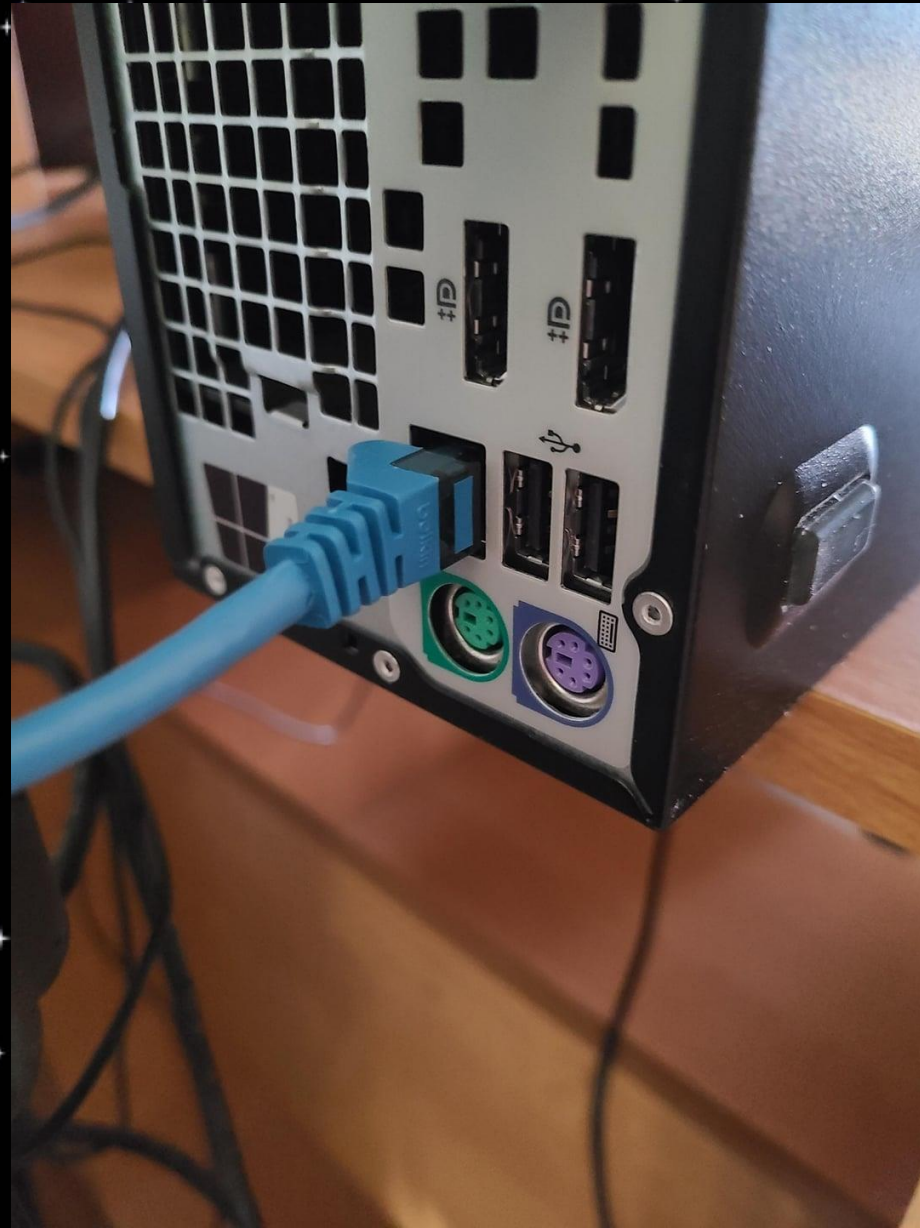


Why can't my computer connect to
the internet?

Physical



Data-Link



Network

```
C:\Users\admin>ipconfig
```

```
Windows IP Configuration
```

```
Ethernet adapter Ethernet0:
```

```
Connection-specific DNS Suffix . : test.local  
Link-local IPv6 Address . . . . . : fe80::27f2:8a5c:b9f8:a797%5  
IPv4 Address. . . . . : 192.168.99.11  
Subnet Mask . . . . . : 255.255.255.0  
Default Gateway . . . . . : 192.168.99.1
```

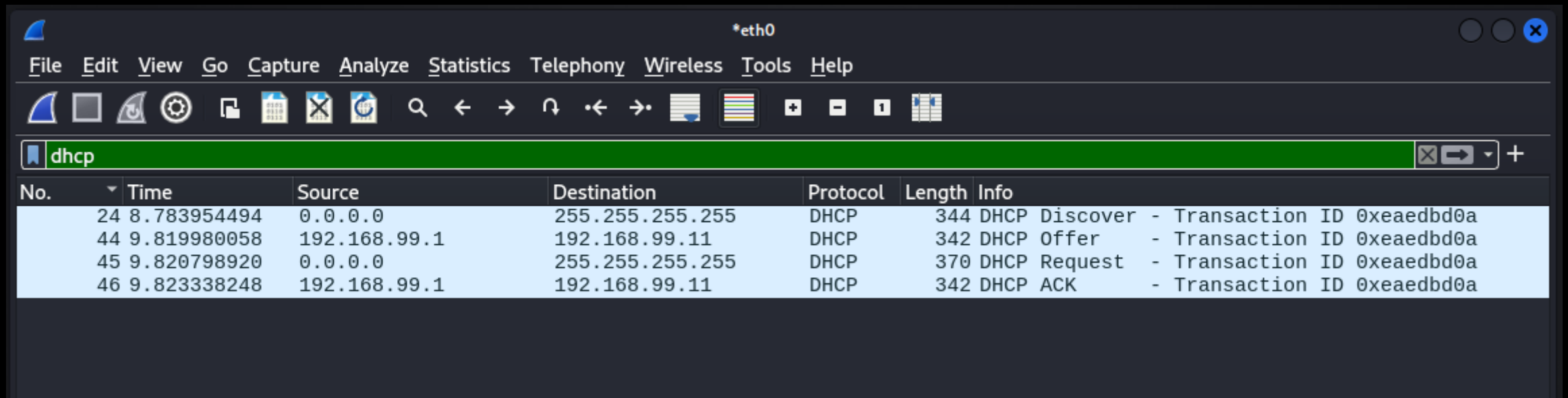
Network – IPv4

- Still the most common IP standard.
- Private ranges include 192.168.x.x, 172.16.x.x, 10.x.x.x

```
IPv4 Address . . . . . : 192.168.99.11
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 192.168.99.1
```

Network - DHCP

- Dynamic Host Configuration Protocol is a network management protocol for the automatic assignment of IPs to devices of a network.



The image shows a Wireshark network traffic capture window titled '*eth0'. The interface includes a menu bar (File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, Help) and a toolbar with various icons. A filter bar at the top shows 'dhcp' selected. Below the filter bar is a table of captured packets. The table has columns for No., Time, Source, Destination, Protocol, Length, and Info. The captured packets are:

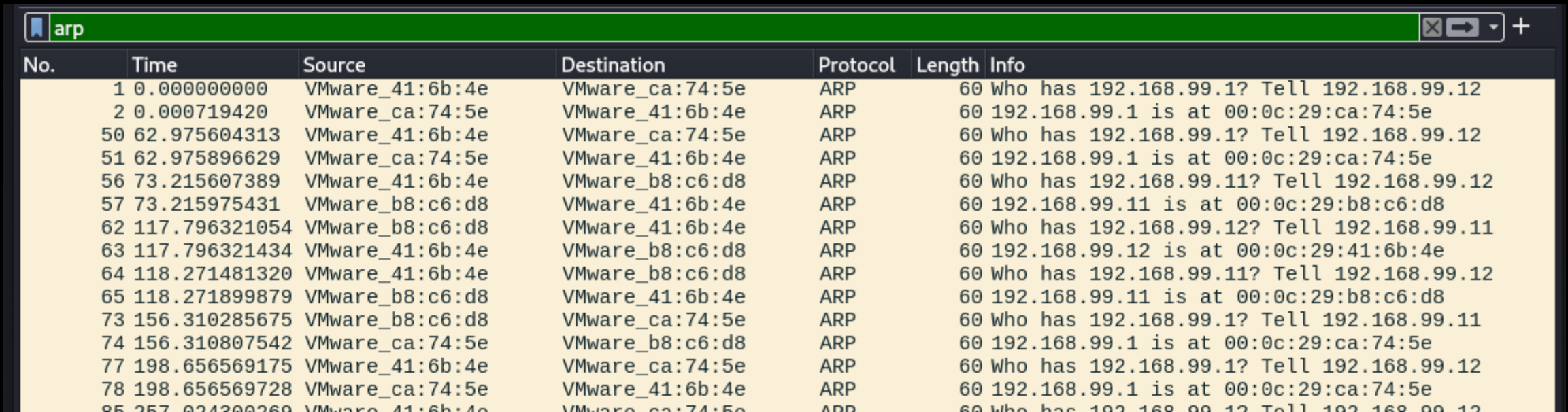
No.	Time	Source	Destination	Protocol	Length	Info
24	8.783954494	0.0.0.0	255.255.255.255	DHCP	344	DHCP Discover - Transaction ID 0xeaedbd0a
44	9.819980058	192.168.99.1	192.168.99.11	DHCP	342	DHCP Offer - Transaction ID 0xeaedbd0a
45	9.820798920	0.0.0.0	255.255.255.255	DHCP	370	DHCP Request - Transaction ID 0xeaedbd0a
46	9.823338248	192.168.99.1	192.168.99.11	DHCP	342	DHCP ACK - Transaction ID 0xeaedbd0a

Network - ARP

Ethernet II, Src: VMware_41:6b:4e (00:0c:29:41:6b:4e), Dst: VMware_b8:c6:d8 (00:0c:29:b8:c6:d8)

Internet Protocol Version 4, Src: 192.168.99.12, Dst: 192.168.99.11

- Address Resolution Protocol is a communication protocol used to associate link layer (MAC) addresses with an IP.



No.	Time	Source	Destination	Protocol	Length	Info
1	0.0000000000	VMware_41:6b:4e	VMware_ca:74:5e	ARP	60	Who has 192.168.99.1? Tell 192.168.99.12
2	0.000719420	VMware_ca:74:5e	VMware_41:6b:4e	ARP	60	192.168.99.1 is at 00:0c:29:ca:74:5e
50	62.975604313	VMware_41:6b:4e	VMware_ca:74:5e	ARP	60	Who has 192.168.99.1? Tell 192.168.99.12
51	62.975896629	VMware_ca:74:5e	VMware_41:6b:4e	ARP	60	192.168.99.1 is at 00:0c:29:ca:74:5e
56	73.215607389	VMware_41:6b:4e	VMware_b8:c6:d8	ARP	60	Who has 192.168.99.11? Tell 192.168.99.12
57	73.215975431	VMware_b8:c6:d8	VMware_41:6b:4e	ARP	60	192.168.99.11 is at 00:0c:29:b8:c6:d8
62	117.796321054	VMware_b8:c6:d8	VMware_41:6b:4e	ARP	60	Who has 192.168.99.12? Tell 192.168.99.11
63	117.796321434	VMware_41:6b:4e	VMware_b8:c6:d8	ARP	60	192.168.99.12 is at 00:0c:29:41:6b:4e
64	118.271481320	VMware_41:6b:4e	VMware_b8:c6:d8	ARP	60	Who has 192.168.99.11? Tell 192.168.99.12
65	118.271899879	VMware_b8:c6:d8	VMware_41:6b:4e	ARP	60	192.168.99.11 is at 00:0c:29:b8:c6:d8
73	156.310285675	VMware_b8:c6:d8	VMware_ca:74:5e	ARP	60	Who has 192.168.99.1? Tell 192.168.99.11
74	156.310807542	VMware_ca:74:5e	VMware_b8:c6:d8	ARP	60	192.168.99.1 is at 00:0c:29:ca:74:5e
77	198.656569175	VMware_41:6b:4e	VMware_ca:74:5e	ARP	60	Who has 192.168.99.1? Tell 192.168.99.12
78	198.656569728	VMware_ca:74:5e	VMware_41:6b:4e	ARP	60	192.168.99.1 is at 00:0c:29:ca:74:5e
85	257.024200260	VMware_41:6b:4e	VMware_ca:74:5e	ARP	60	Who has 192.168.99.12? Tell 192.168.99.11

Network - ICMP

- Internet Control Message Protocol is a communication protocol used to verify data transmission by indicating success or failure when communicating with another device.

```
C:\Users\admin>ping 8.8.8.8
```

```
Pinging 8.8.8.8 with 32 bytes of data:
```

```
Reply from 8.8.8.8: bytes=32 time=13ms TTL=48
```

```
Reply from 8.8.8.8: bytes=32 time=13ms TTL=48
```

```
Reply from 8.8.8.8: bytes=32 time=13ms TTL=48
```

```
Reply from 8.8.8.8: bytes=32 time=16ms TTL=48
```

```
Ping statistics for 8.8.8.8:
```

```
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
```

```
Approximate round trip times in milli-seconds:
```

```
    Minimum = 13ms, Maximum = 16ms, Average = 13ms
```

2	3.097182840	192.168.99.11	8.8.8.8	ICMP	74 Echo (ping) request	id=0x0001, seq=5/1280, ttl=128
3	3.110681849	8.8.8.8	192.168.99.11	ICMP	74 Echo (ping) reply	id=0x0001, seq=5/1280, ttl=48
4	4.105955303	192.168.99.11	8.8.8.8	ICMP	74 Echo (ping) request	id=0x0001, seq=6/1536, ttl=128
5	4.119145076	8.8.8.8	192.168.99.11	ICMP	74 Echo (ping) reply	id=0x0001, seq=6/1536, ttl=48
6	5.121776640	192.168.99.11	8.8.8.8	ICMP	74 Echo (ping) request	id=0x0001, seq=7/1792, ttl=128
7	5.135501510	8.8.8.8	192.168.99.11	ICMP	74 Echo (ping) reply	id=0x0001, seq=7/1792, ttl=48
8	6.137308486	192.168.99.11	8.8.8.8	ICMP	74 Echo (ping) request	id=0x0001, seq=8/2048, ttl=128
9	6.153824286	8.8.8.8	192.168.99.11	ICMP	74 Echo (ping) reply	id=0x0001, seq=8/2048, ttl=48

Google Search Engine

This is a demo of the Google Search Engine. Note, it is research in progress so expect some downtimes and malfunctions. You can find the older [Backrub web page here](#).

Google is being developed by [Larry Page](#) and [Sergey Brin](#) with very talented implementation help by [Scott Hassan](#) and [Alan Sterenberg](#).



Search Stanford

Search The Web



What does a connection look like?

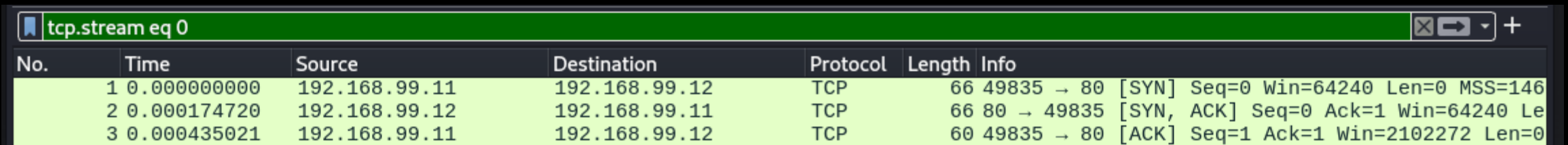
Transport – TCP (A Segment of a Stream)

- Transmission Control Protocol is a connection-oriented protocol.
- A Client and Server first must establish communication with each other and agree how they will communicate application data.
- Ideal for sending a lot of data (segmented) and ensuring integrity of the data sent.

Transmission Control Protocol, Src Port: 80, Dst Port: 49861, Seq: 1, Ack: 429, Len: 411

Transport – TCP – SYN-ACK Three-Way Handshake

- 1. SYN:** A Server will listen for SYN request from a Client to SYNchronize.
- 2. SYN-ACK:** The Server will respond with a SYN-ACK back to the client with an ACKnowledgement of the Client's SYN request, and its own request to SYNchronize with the Client.
- 3. ACK:** The Client will then respond to the Servers SYN with its own ACKnowledgement.



The image shows a Wireshark packet capture window titled "tcp.stream eq 0". It displays three packets in a table format, illustrating the TCP three-way handshake process.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.0000000000	192.168.99.11	192.168.99.12	TCP	66	49835 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=146
2	0.000174720	192.168.99.12	192.168.99.11	TCP	66	80 → 49835 [SYN, ACK] Seq=0 Ack=1 Win=64240 Le
3	0.000435021	192.168.99.11	192.168.99.12	TCP	60	49835 → 80 [ACK] Seq=1 Ack=1 Win=2102272 Len=0

Transport – TCP – FIN-ACK Four-Way Handshake

- 1. FIN-ACK:** Sending the Server a FIN-ACK to FINish this connection.
- 2. ACK:** The Server will respond with an ACKnowledgement, and send its own FIN-ACK to finish to the client, which the Client will then ACKnowledge.
- 3. Other side Repeats**

```
13 5.030730739 192.168.99.12 192.168.99.11 TCP 60 80 → 49835 [FIN, ACK] Seq=449 Ack=429 Win=6412
14 5.030932638 192.168.99.11 192.168.99.12 TCP 60 49835 → 80 [ACK] Seq=429 Ack=450 Win=2101760 L
18 8.155883978 192.168.99.11 192.168.99.12 TCP 60 49835 → 80 [FIN, ACK] Seq=429 Ack=450 Win=2101
19 8.156199219 192.168.99.12 192.168.99.11 TCP 60 80 → 49835 [ACK] Seq=450 Ack=430 Win=64128 Len
```

Data – TCP – HTTP

- 1. The Client will send an HTTP Request Message to the server.** For longer messages, Data can be sent in multiple packets, and those packets are sent ordered by a Sequence Number, which is then arranged by the Server in the proper order
- 2. The Server then sends back an Acknowledgement of every Sequence Numbered Packet** it gets, ensuring the Client knows if something was lost in transit and if it must retransmit some data.
- 3. The Client can send a PSH or Push to the Server to indicate that it has completed sending** and that anything the Server has received should be pushed to the Server Application.
- 4. The process repeats with the Server response.**

Hypertext Transfer Protocol, HTTP/1.1 200 OK\r\n

Line-based text data: text/html (13 lines)

Data – TCP – HTTP

The image shows a Wireshark packet capture window titled "tcp.stream eq 0". The main pane displays a list of 19 network packets. Packet 6 is highlighted in blue, representing the HTTP response. The details pane below shows the structure of this packet, including Ethernet II, Internet Protocol Version 4, Transmission Control Protocol, and Hypertext Transfer Protocol. The HTML content of the response is displayed in a line-based text format.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.99.11	192.168.99.12	TCP	66	49835 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=146
2	0.000174720	192.168.99.12	192.168.99.11	TCP	66	80 → 49835 [SYN, ACK] Seq=0 Ack=1 Win=64240 Le
3	0.000435021	192.168.99.11	192.168.99.12	TCP	60	49835 → 80 [ACK] Seq=1 Ack=1 Win=2102272 Len=0
4	0.023549278	192.168.99.11	192.168.99.12	HTTP	482	GET / HTTP/1.1
5	0.023960738	192.168.99.12	192.168.99.11	TCP	60	80 → 49835 [ACK] Seq=1 Ack=429 Win=64128 Len=0
6	0.025102439	192.168.99.12	192.168.99.11	HTTP	502	HTTP/1.1 200 OK (text/html)
9	0.074263499	192.168.99.11	192.168.99.12	TCP	60	49835 → 80 [ACK] Seq=429 Ack=449 Win=2101760 L
13	5.030730739	192.168.99.12	192.168.99.11	TCP	60	80 → 49835 [FIN, ACK] Seq=449 Ack=429 Win=6412
14	5.030932638	192.168.99.11	192.168.99.12	TCP	60	49835 → 80 [ACK] Seq=429 Ack=450 Win=2101760 L
18	8.155883978	192.168.99.11	192.168.99.12	TCP	60	49835 → 80 [FIN, ACK] Seq=429 Ack=450 Win=2101
19	8.156199219	192.168.99.12	192.168.99.11	TCP	60	80 → 49835 [ACK] Seq=450 Ack=430 Win=64128 Len

Frame 6: 502 bytes on wire (4016 bits), 502 bytes captured (4016 bits) on interface e
Ethernet II, Src: VMware_41:6b:4e (00:0c:29:41:6b:4e), Dst: VMware_b8:c6:d8 (00:0c:29:41:6b:4e)
Destination: VMware_b8:c6:d8 (00:0c:29:b8:c6:d8)
Source: VMware_41:6b:4e (00:0c:29:41:6b:4e)
Type: IPv4 (0x0800)
Internet Protocol Version 4, Src: 192.168.99.12, Dst: 192.168.99.11
Transmission Control Protocol, Src Port: 80, Dst Port: 49835, Seq: 1, Ack: 429, Len: 502
Hypertext Transfer Protocol
Line-based text data: text/html (13 lines)
<!DOCTYPE html>\n
<html>\n
<head>\n
<title>Test Page</title>\n
</head>\n
\n
<body>\n
\n
<h1>Test Page</h1>\n
<p>Please Ignore</p>\n
\n
</body>\n
</html>\n

0000 00 0c 29 b8 c6 d8 00 0c 29 41
0010 01 e8 e1 73 40 00 40 06 10 34
0020 63 0b 00 50 c2 ab e7 43 2a 45
0030 01 f5 ef 25 00 00 48 54 54 50
0040 30 30 20 4f 4b 0d 0a 44 61 74
0050 2c 20 32 39 20 53 65 70 20 32
0060 3a 32 33 3a 33 37 20 47 4d 54
0070 65 72 3a 20 41 70 61 63 68 65
0080 32 20 28 46 65 64 6f 72 61 20
0090 20 4f 70 65 6e 53 53 4c 2f 33
00a0 4c 61 73 74 2d 4d 6f 64 69 66
00b0 75 6e 2c 20 32 39 20 53 65 70
00c0 31 39 3a 32 33 3a 31 38 20 47
00d0 61 67 3a 20 22 38 31 2d 36 32
00e0 32 66 31 65 34 22 0d 0a 41 63
00f0 61 6e 67 65 73 3a 20 62 79 74
0100 6e 74 65 6e 74 2d 4c 65 6e 67
0110 39 0d 0a 4b 65 65 70 2d 41 6c
0120 69 6d 65 6f 75 74 3d 35 2c 20
0130 30 0d 0a 43 6f 6e 6e 65 63 74
0140 65 65 70 2d 41 6c 69 76 65 0d
0150 6e 74 2d 54 79 70 65 3a 20 74
0160 6d 6c 3b 20 63 68 61 72 73 65

To Review...

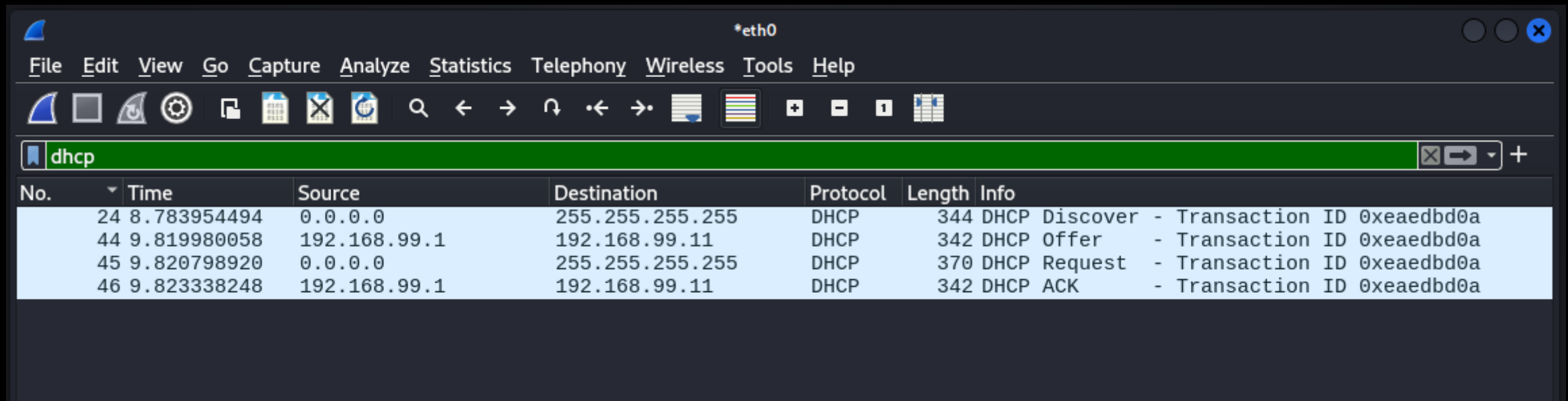
Frame 923: 465 bytes on wire (3720 bits), 465 bytes captured (3720 bits) on interface eth0, id 0
Ethernet II, Src: VMware_41:6b:4e (00:0c:29:41:6b:4e), Dst: VMware_b8:c6:d8 (00:0c:29:b8:c6:d8)
Internet Protocol Version 4, Src: 192.168.99.12, Dst: 192.168.99.11
Transmission Control Protocol, Src Port: 80, Dst Port: 49861, Seq: 1, Ack: 429, Len: 411
Hypertext Transfer Protocol, HTTP/1.1 200 OK\r\n
Line-based text data: text/html (13 lines)

Transport – UDP (A Datagram)

- User Datagram Protocol is a connectionless protocol.
- Responses are totally dependent on the Application at hand.
- Ideal for things like Gaming, VoIP, Streaming, or Transactional Messaging

Data - UDP – DHCP (again)

- Being connectionless also means it can handle Broadcast messages, or messages sent to every host on a subnet.



The image shows a Wireshark network traffic capture window titled '*eth0'. The interface includes a menu bar (File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, Help) and a toolbar with various icons. A filter bar at the top shows 'dhcp' selected. Below the filter bar is a table of captured packets. The table has columns for No., Time, Source, Destination, Protocol, Length, and Info. Four packets are listed, all related to DHCP with a Transaction ID of 0xeaedbd0a. The first packet is a DHCP Discover message from 0.0.0.0 to 255.255.255.255. The second is a DHCP Offer from 192.168.99.1 to 192.168.99.11. The third is a DHCP Request from 0.0.0.0 to 255.255.255.255. The fourth is a DHCP ACK from 192.168.99.1 to 192.168.99.11.

No.	Time	Source	Destination	Protocol	Length	Info
24	8.783954494	0.0.0.0	255.255.255.255	DHCP	344	DHCP Discover - Transaction ID 0xeaedbd0a
44	9.819980058	192.168.99.1	192.168.99.11	DHCP	342	DHCP Offer - Transaction ID 0xeaedbd0a
45	9.820798920	0.0.0.0	255.255.255.255	DHCP	370	DHCP Request - Transaction ID 0xeaedbd0a
46	9.823338248	192.168.99.1	192.168.99.11	DHCP	342	DHCP ACK - Transaction ID 0xeaedbd0a

- via 255.255.255.255

Data – UDP – DNS

- Purely Transactional
- Query, Query Response

```
DNS      70 Standard query 0x0004 A google.com
DNS      86 Standard query response 0x0004 A google.com A 142.250.190.110
DNS      70 Standard query 0x0005 AAAA google.com
DNS      98 Standard query response 0x0005 AAAA google.com AAAA 2607:f8b0:4009:
```

How do I know that a Threat Actor
can't read my connection?

Wireshark – Follow TCP Stream

The screenshot shows the Wireshark interface with the following details:

- Filter:** `ip.src == 192.168.99.0/24 && ip.dst == 192.168.99.0/24`
- Packet List:**

No.	Time	Source	Destination	Protocol	Length	Info
918	4.842324264	192.168.99.11	192.168.99.12	TCP	66	49861 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=0
919	4.842849557	192.168.99.12	192.168.99.11	TCP	60	80 → 49861 [ACK] Seq=2272 Win=0 Len=0
920	4.842849936	192.168.99.11	192.168.99.12	TCP	60	49861 → 80 [ACK] Seq=64128 Win=0 Len=0
921	4.848131441	192.168.99.11	192.168.99.12	HTTP	4	GET / HTTP/1.1
922	4.848638669	192.168.99.12	192.168.99.11	TCP	60	80 → 49861 [ACK] Seq=2272 Win=0 Len=0
923	4.849183025	192.168.99.12	192.168.99.11	HTTP	4	200 OK
924	4.849183313	192.168.99.12	192.168.99.11	TCP	60	80 → 49861 [ACK] Seq=2272 Win=0 Len=0
925	4.849514279	192.168.99.11	192.168.99.12	TCP	60	49861 → 80 [ACK] Seq=64128 Win=0 Len=0
926	4.866910142	192.168.99.11	192.168.99.12	TCP	60	49861 → 80 [ACK] Seq=64128 Win=0 Len=0
927	4.866910547	192.168.99.12	192.168.99.11	TCP	60	80 → 49861 [ACK] Seq=2272 Win=0 Len=0
929	5.062472786	192.168.99.11	192.168.99.12	TCP	60	49861 → 80 [ACK] Seq=64128 Win=0 Len=0
930	5.063200737	192.168.99.12	192.168.99.11	TCP	60	80 → 49861 [ACK] Seq=2272 Win=0 Len=0
931	5.063628649	192.168.99.11	192.168.99.12	TCP	60	49861 → 80 [ACK] Seq=64128 Win=0 Len=0
932	5.063629184	192.168.99.11	192.168.99.12	HTTP	4	GET / HTTP/1.1
933	5.063961334	192.168.99.12	192.168.99.11	TCP	60	80 → 49861 [ACK] Seq=2272 Win=0 Len=0
934	5.065104590	192.168.99.12	192.168.99.11	HTTP	4	200 OK
935	5.065994813	192.168.99.12	192.168.99.11	TCP	60	80 → 49861 [ACK] Seq=2272 Win=0 Len=0
936	5.065995633	192.168.99.11	192.168.99.12	TCP	60	49861 → 80 [ACK] Seq=64128 Win=0 Len=0

Packet 918 Details:

- Frame 918: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface eth0
- Ethernet II, Src: VMware_b8:c6:d8 (00:0c:29:b8:c6:d8), Dst: VMware_41:6b:4e (00:0c:29:41:6b:4e)
- Destination: VMware_41:6b:4e (00:0c:29:41:6b:4e)
- Source: VMware_b8:c6:d8 (00:0c:29:b8:c6:d8)
- Type: IPv4 (0x0800)

Follow TCP Stream Context Menu:

- Mark/Unmark Packet (Ctrl+M)
- Ignore/Unignore Packet (Ctrl+D)
- Set/Unset Time Reference (Ctrl+T)
- Time Shift... (Ctrl+Shift+T)
- Packet Comments
- Edit Resolved Name
- Apply as Filter
- Prepare as Filter
- Conversation Filter
- Colorize Conversation
- SCTP
- Follow (TCP Stream Ctrl+Alt+Shift+T)**
- Copy
- Protocol Preferences
- Decode As...

Data – TCP – HTTP (again)

```
Wireshark · Follow TCP Stream (tcp.stream eq 2) · eth0

GET / HTTP/1.1
Host: 192.168.99.12
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/128.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9

HTTP/1.1 200 OK
Date: Sun, 29 Sep 2024 22:13:07 GMT
Server: Apache/2.4.62 (Fedora Linux) OpenSSL/3.2.2
Last-Modified: Sun, 29 Sep 2024 19:23:18 GMT
ETag: "81-623470542f1e4"
Accept-Ranges: bytes
Content-Length: 129
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8

<!DOCTYPE html>
<html>
<head>
<title>Test Page</title>
</head>

<body>

<h1>Test Page</h1>
<p>Please Ignore</p>

</body>
</html>
GET /favicon.ico HTTP/1.1
```

Transport – TCP – Transport Layer Security

- Implemented as part of the TCP process.
- Negotiates a Cipher and a Public Key Exchange to encrypt Data
- The result (should be) something entirely undecipherable

W2dmfC9mfGEoey9sfXZ/e2BofW
5/Z2ZsbmNjdi98YHphay9tensvZ2
B/aml6Y2N2L2hqe3wve2dqL39g
ZmF7L25sfWB8fCEvRmkvdmB6L
3x/amF7L3tnai97ZmJqL3tgL2x9b
mxkL3tnZnwve2dgemhnlly9pamp
jL2l9amove2AvY2p7L2JqL2RhYH
gvY2BjIQ==

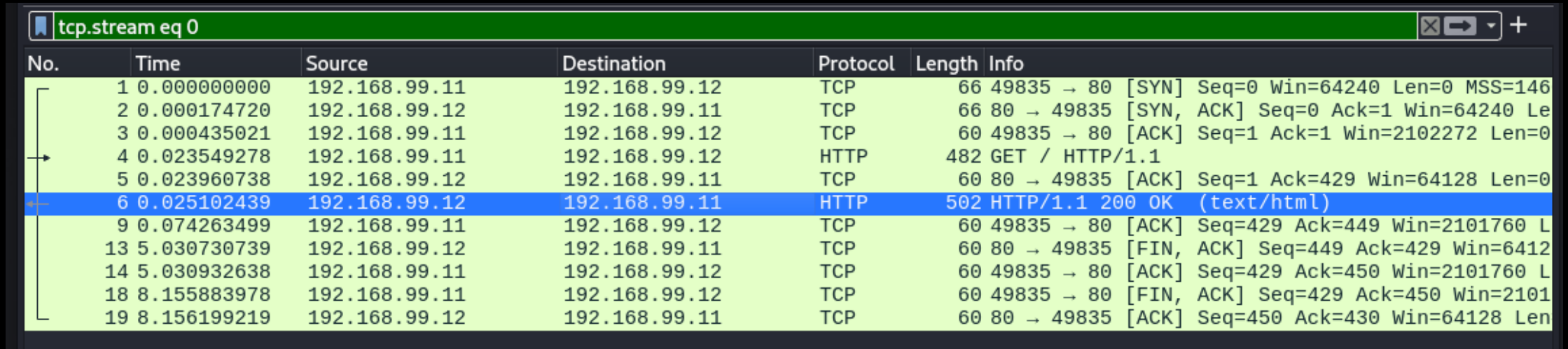


All right, then. Keep your secrets.

Data – TCP – TLS (v1.2)

- 1. ClientHello:** The Client will send a ClientHello which specifies the highest version of TLS the application on the Client side will support, suggested ciphers, and suggested compression.
- 2. ServerHello:** The Server responds with a ServerHello (shocker) containing the chosen TLS version, the cipher, and the compression method. The Server will also send its Certificate, which the Client will verify against its own Certificate Authority Chain. Every Browser used on the modern web comes with a package of Trusted Certificate Authorities that dictate which Certificates it will accept based on the Signing CA.
- 3. ClientKeyExchange:** The Client will then, based on the selected Cipher, send its Public Key back in a ClientKeyExchange message, and a ChangeCipherSpec message, which indicates that the conversation from then on will be authenticated and encrypted, along with an encrypted Finished message. All this in one packet.
- 4. NewSessionTicket:** Server responds NewSessionTicket, and its own ChangeCipherSpec and encrypted Finish. Likewise, all in one packet. From there the Encrypted Data will flow.

Data – TCP – HTTP



The image shows a Wireshark packet capture window titled "tcp.stream eq 0". The window displays a list of 11 network packets. The columns are: No., Time, Source, Destination, Protocol, Length, and Info. Packet 6 is highlighted in blue. The packets show a sequence of TCP and HTTP traffic between 192.168.99.11 and 192.168.99.12.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.99.11	192.168.99.12	TCP	66	49835 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=146
2	0.000174720	192.168.99.12	192.168.99.11	TCP	66	80 → 49835 [SYN, ACK] Seq=0 Ack=1 Win=64240 Le
3	0.000435021	192.168.99.11	192.168.99.12	TCP	60	49835 → 80 [ACK] Seq=1 Ack=1 Win=2102272 Len=0
4	0.023549278	192.168.99.11	192.168.99.12	HTTP	482	GET / HTTP/1.1
5	0.023960738	192.168.99.12	192.168.99.11	TCP	60	80 → 49835 [ACK] Seq=1 Ack=429 Win=64128 Len=0
6	0.025102439	192.168.99.12	192.168.99.11	HTTP	502	HTTP/1.1 200 OK (text/html)
9	0.074263499	192.168.99.11	192.168.99.12	TCP	60	49835 → 80 [ACK] Seq=429 Ack=449 Win=2101760 L
13	5.030730739	192.168.99.12	192.168.99.11	TCP	60	80 → 49835 [FIN, ACK] Seq=449 Ack=429 Win=6412
14	5.030932638	192.168.99.11	192.168.99.12	TCP	60	49835 → 80 [ACK] Seq=429 Ack=450 Win=2101760 L
18	8.155883978	192.168.99.11	192.168.99.12	TCP	60	49835 → 80 [FIN, ACK] Seq=429 Ack=450 Win=2101
19	8.156199219	192.168.99.12	192.168.99.11	TCP	60	80 → 49835 [ACK] Seq=450 Ack=430 Win=64128 Len

Total Packets: 11

Data – TCP – HTTPS (TLS v1.2)

No.	Time	Source	Destination	Protocol	Length	Info
14	0.017337908	192.168.99.11	192.168.99.12	TCP	66	49812 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=14
15	0.017579910	192.168.99.12	192.168.99.11	TCP	66	443 → 49812 [SYN, ACK] Seq=0 Ack=1 Win=64240 L
16	0.017795374	192.168.99.11	192.168.99.12	TCP	60	49812 → 443 [ACK] Seq=1 Ack=1 Win=2102272 Len=
17	0.018536775	192.168.99.11	192.168.99.12	TLSv1.2	2059	Client Hello
18	0.018536997	192.168.99.12	192.168.99.11	TCP	60	443 → 49812 [ACK] Seq=1 Ack=1461 Win=67072 Len
19	0.018691510	192.168.99.12	192.168.99.11	TCP	60	443 → 49812 [ACK] Seq=1 Ack=2006 Win=70016 Len
20	0.020817665	192.168.99.12	192.168.99.11	TLSv1.2	3155	Server Hello, Certificate, Server Key Exchange
21	0.021126916	192.168.99.11	192.168.99.12	TCP	60	49812 → 443 [ACK] Seq=2006 Ack=3102 Win=210227
22	0.022076979	192.168.99.11	192.168.99.12	TLSv1.2	147	Client Key Exchange, Change Cipher Spec, Encry
23	0.023034273	192.168.99.11	192.168.99.12	TLSv1.2	721	Application Data
24	0.023442457	192.168.99.12	192.168.99.11	TLSv1.2	328	New Session Ticket, Change Cipher Spec, Encryp
25	0.024349749	192.168.99.12	192.168.99.11	TLSv1.2	494	Application Data
26	0.024533679	192.168.99.11	192.168.99.12	TCP	60	49812 → 443 [ACK] Seq=2766 Ack=3816 Win=210150
27	0.024533820	192.168.99.12	192.168.99.11	TLSv1.2	85	Encrypted Alert
28	0.024533863	192.168.99.12	192.168.99.11	TCP	60	443 → 49812 [FIN, ACK] Seq=3847 Ack=2766 Win=7
29	0.024722163	192.168.99.11	192.168.99.12	TCP	60	49812 → 443 [ACK] Seq=2766 Ack=3848 Win=210150
30	0.030184043	192.168.99.11	192.168.99.12	TCP	60	49812 → 443 [FIN, ACK] Seq=2766 Ack=3848 Win=2
31	0.030184405	192.168.99.12	192.168.99.11	TCP	60	443 → 49812 [ACK] Seq=3848 Ack=2767 Win=72960

Total Packets: 18

Data – TCP – HTTPS (TLS v1.3) (defined 2018)

- The client now leads with all its supported ciphers and public keys in its ClientHello, allowing the Server to start with an Encrypted ServerHello and Certificate.

The image shows a Wireshark packet capture window titled "tcp.stream eq 8". The capture shows a sequence of 17 packets between source IP 192.168.99.11 and destination IP 192.168.99.12. The handshake starts with a SYN packet (No. 135), followed by SYN-ACK (No. 136), ACK (No. 137), and ClientHello (No. 138). The server responds with ServerHello, Change Cipher Spec, and Application Data (Nos. 139-143). The client then sends ACK (No. 144), Change Cipher Spec, and Application Data (Nos. 145-147). The connection ends with FIN, ACK (No. 148), ACK (No. 149), FIN, ACK (No. 151), and ACK (No. 152).

No.	Time	Source	Destination	Protocol	Length	Info
135	7.390563965	192.168.99.11	192.168.99.12	TCP	66	49700 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=14
136	7.390750590	192.168.99.12	192.168.99.11	TCP	66	443 → 49700 [SYN, ACK] Seq=0 Ack=1 Win=64240 L
137	7.390907842	192.168.99.11	192.168.99.12	TCP	60	49700 → 443 [ACK] Seq=1 Ack=1 Win=2102272 Len=
138	7.391228982	192.168.99.11	192.168.99.12	TLSv1.3	2090	Client Hello
139	7.391968698	192.168.99.12	192.168.99.11	TCP	60	443 → 49700 [ACK] Seq=1 Ack=1461 Win=67072 Len
140	7.391968930	192.168.99.12	192.168.99.11	TCP	60	443 → 49700 [ACK] Seq=1 Ack=2037 Win=70016 Len
141	7.394063033	192.168.99.12	192.168.99.11	TLSv1.3	310	Server Hello, Change Cipher Spec, Application
142	7.394571982	192.168.99.11	192.168.99.12	TLSv1.3	134	Change Cipher Spec, Application Data
143	7.394787039	192.168.99.11	192.168.99.12	TLSv1.3	640	Application Data
144	7.394787195	192.168.99.12	192.168.99.11	TLSv1.3	341	Application Data
145	7.395421315	192.168.99.12	192.168.99.11	TLSv1.3	472	Application Data
146	7.395421449	192.168.99.11	192.168.99.12	TCP	60	49700 → 443 [ACK] Seq=2703 Ack=962 Win=2101248
147	7.395679087	192.168.99.12	192.168.99.11	TLSv1.3	78	Application Data
148	7.395679194	192.168.99.12	192.168.99.11	TCP	60	443 → 49700 [FIN, ACK] Seq=986 Ack=2703 Win=72
149	7.395788794	192.168.99.11	192.168.99.12	TCP	60	49700 → 443 [ACK] Seq=2703 Ack=987 Win=2101248
151	7.396549138	192.168.99.11	192.168.99.12	TCP	60	49700 → 443 [FIN, ACK] Seq=2703 Ack=987 Win=21
152	7.396718173	192.168.99.12	192.168.99.11	TCP	60	443 → 49700 [ACK] Seq=987 Ack=2704 Win=72960 L

Total Packets: 17

Data – TCP – HTTPS

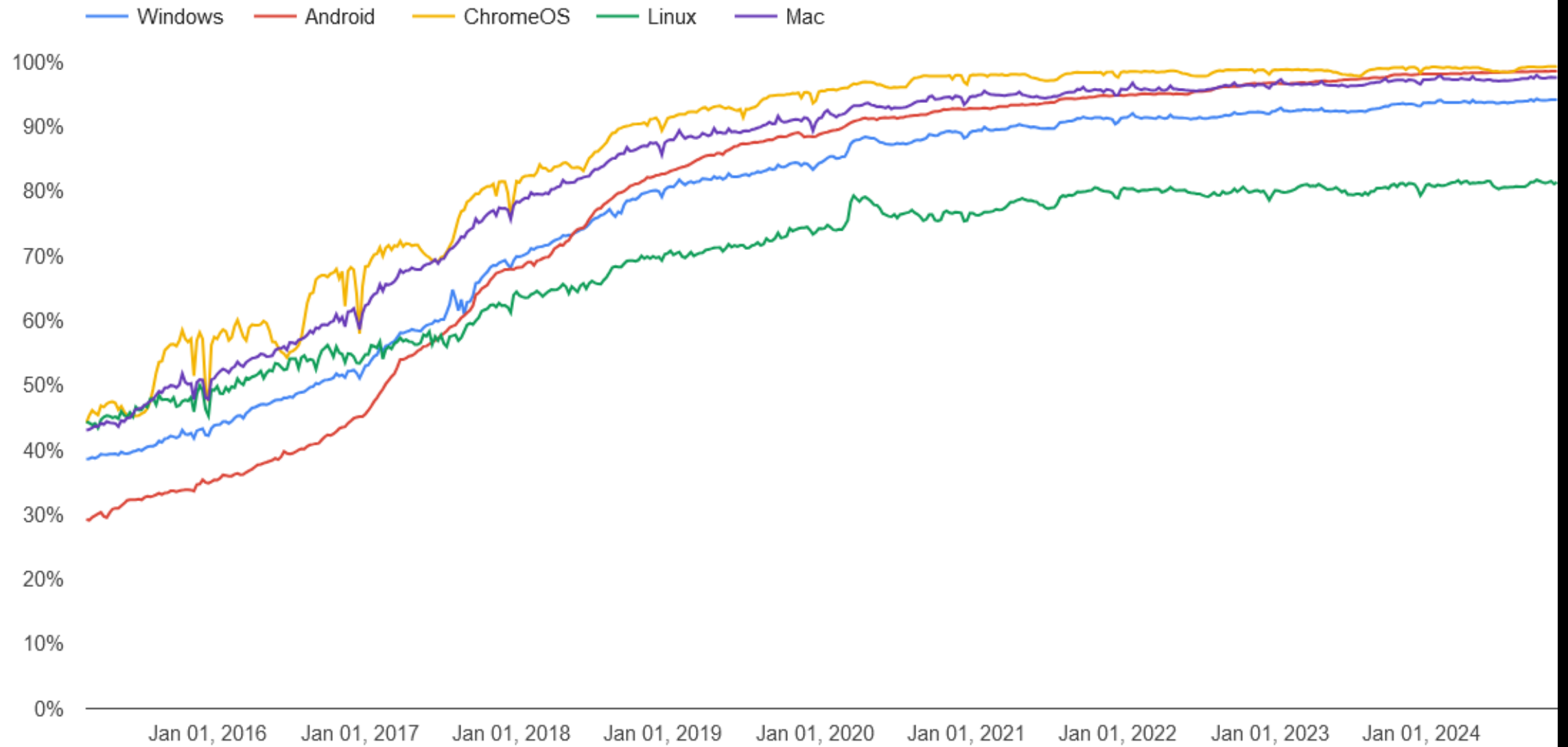
Wireshark · Follow TCP Stream (tcp.stream eq 1) · eth0

```
.....u.*.....l,-...f.Q.1..Q..{= cr0a..Y.cT.=.F.,Cr.%%9r.&....'... zz.....+./.,.0...../5...c.....
.....j. ....'.....@...
.....t.ws.....v8vl.j4...?.H...2.u0. ....`@Z.8.^....Cd.....Gl..T.....b.D;'...)p....d?/...U....a.D.p.P%[.....R..
h...<...pa.xUa.....v.....s:.,.....k5.f.....h.X.]p../4.[.D{.;.nA.....QM"z'. ..X.';Z.#.....t.z..
U....[$.-
.....3.....c.....9..(.).....a.....E.3.wIF..JP..g..V./Y....H....r#h..9..6'....Qa.....
>T.'W>.Y.....3..'.G.X.i....st.Ji".jq....;c.....f.V....\.."+*...8.I..`!D.y..h...H...Gu..
.[@.:q.^..L.....KIp4
..Zv.....4...*..p...8...j.#p.|=.4.DA\..#.....W...7...0~.`[.b.A.C.tE*...D=.;.<...=_S.."H...#S.q.z...)t...3&g:..
.....Bm...s0x.A....v.\.hel...c...d...u./k.....ZSHP.j.w.7..v.....N.< ..$......b.9...B.+Y....Y%...9...
.....@
..C...U.}"..
F...I.....4.HM....x.;i:..pN..#K.r.$....-..q.r....J)...QP...83..re....).C...,0..g"..Q.....0`f.....\a...<2&.[...C.
t.....kJ.{.<?.K.Y..N(..q..Y.F
.....4 Ru..v..{u=..~ ..)}<u<v...\.@k...S-..&.8.?w...<#.Mj&7.v...*...f.....M%gD.G...U.,.m.\...W....5aR]L.....D
....{..}.....}.....Ii.._E.2.x...I7. ....1.%..[.....<..TE...B.",..mC(3r.*...@

2.....\..~..hp...X.../..
..G#g...Q1C.>.....`&<.P..D...0<E
4.....Y'r.f!.....hW.C\..0.$..L.?..0.3T...a..Ts.C.|(k...@...:00..{q-&.....r..7+#...m..c.st...a.X....|...U...|(
KU.T>:=..o.....W. .H.T..(^.....1.)l7,..9.8.9.jg;j.R.`y.
...i{.X.G..S.lAs.68Y.. o..}\...
.5..r...!.~zx.0..}.zC...X...e..K].....l..'v...+0.....i.....%.*.._Q.....r.HE.^..RT... ..H...[}?s
Pc*...$.8.^_%. ..9.....-2otY...@.....3p.....1.
.....+.....h2.http/1.1.....#..*'..g..\..oLq..)i.&!N:.....0~.3..4...Y..
8)^.*.,.A.....B...Qj.,.m.X~+L...R.[].t.-.0]_..Z.....2.!y.....r.%g2p...h...../.....-b0.$.....i%pkc...)F..
.gx...{pS...v..m-.B.[.....q.....,+..ldi.....h2.
...
..C.....P...L..S=.=..D...s...*.|RJ*S...zq.c.....0..$......#..... .http/1.1.....
..
...
...0..0.....
...0
..*..H..
.....0m1.0 ..U....US1.0...U.
..Unspecified1.0...U....ca-91459085705860947701
0...U...web11.0.. ..*..H..
.. ..root@web10..
240929191521Z.
250929191521Z0L1.0 ..U....US1.0...U.
..Unspecified1
0...U...web11.0.. ..*..H..

3 client pkts, 4 server pkts, 3 turns.
```

Percentage of pages loaded over HTTPS in Chrome by platform



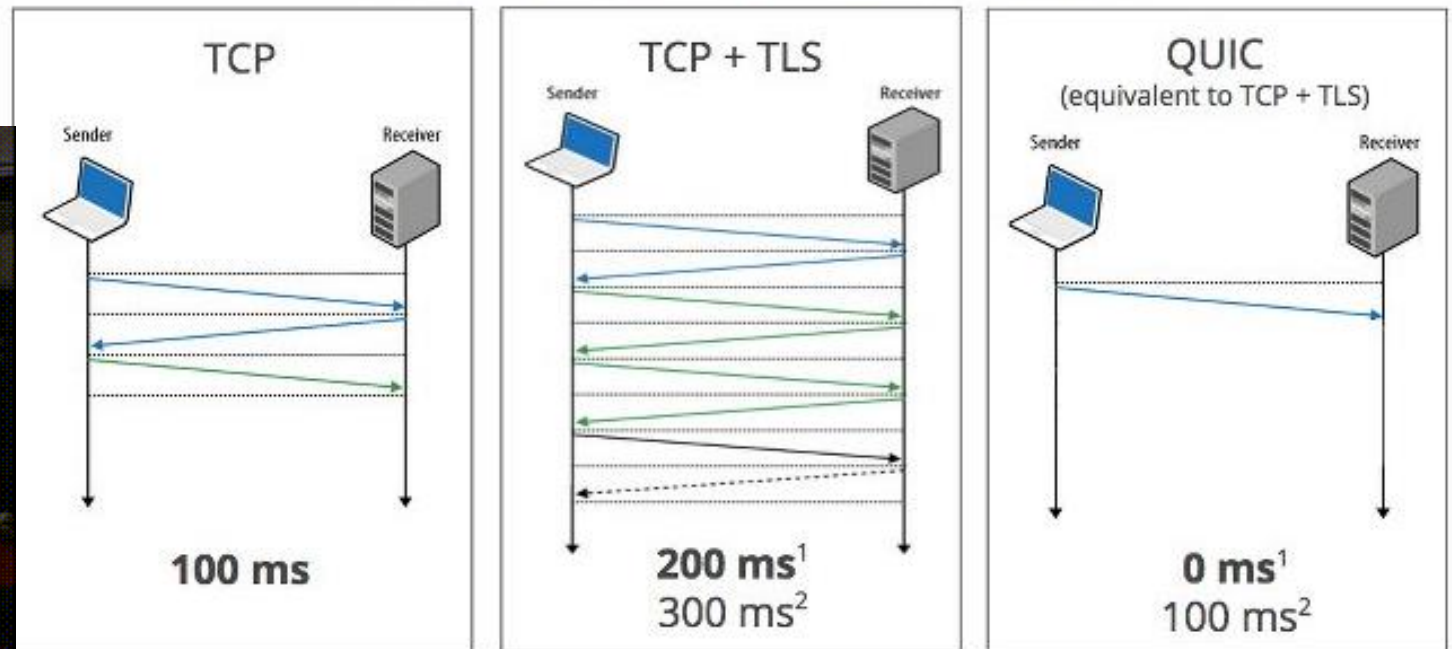
Fragment navigations, history push state navigations, and all schemes besides HTTP/HTTPS (including new tab page navigations) are not included.

Source: <https://transparencyreport.google.com/https/overview>

Transport – QUIC (UDP+TLS) (Defined 2020*)

- Initially pitched at “Quick UDP Internet Connections”
- Reduces the latency caused by cumbersome TLS handshakes
- Built with HTTP advances in mind.

Zero RTT Connection Establishment



1. Repeat connection
2. Never talked to server before



Transport – QUIC

chrome-cloudflare-quic.pcapng 41.7 kb · 83 packets · [more info](#)

Start typing a Display Filter Apply Clear

No.	Time	Source	Destination	Protocol	Length	Info
47	5.478636	2001:db8:1::1	2606:4700:10::6816:826	QUIC	1292	Initial, DCID=203f9e9f68698274, PKN: 1, PADDING, CRYPTO, CRYPTO, CRYPTO, PADDING,
48	5.497206	2606:4700:10::6816:826	2001:db8:1::1	QUIC	1262	Protected Payload (KP0)
49	5.500617	2606:4700:10::6816:826	2001:db8:1::1	QUIC	1262	Protected Payload (KP0)
50	5.500951	2606:4700:10::6816:826	2001:db8:1::1	QUIC	1262	Handshake, SCID=0130dfc5a047e6acd230b5c5e047ced9b0a6bbf0
51	5.501254	2606:4700:10::6816:826	2001:db8:1::1	QUIC	807	Handshake, SCID=0130dfc5a047e6acd230b5c5e047ced9b0a6bbf0
52	5.501480	2001:db8:1::1	2606:4700:10::6816:826	QUIC	115	Handshake, DCID=0130dfc5a047e6acd230b5c5e047ced9b0a6bbf0
53	5.502620	2001:db8:1::1	2606:4700:10::6816:826	QUIC	147	Handshake, DCID=0130dfc5a047e6acd230b5c5e047ced9b0a6bbf0
54	5.502739	2001:db8:1::1	2606:4700:10::6816:826	QUIC	143	Protected Payload (KP0), DCID=0130dfc5a047e6acd230b5c5e047ced9b0a6bbf0
55	5.502856	2001:db8:1::1	2606:4700:10::6816:826	QUIC	540	Protected Payload (KP0), DCID=0130dfc5a047e6acd230b5c5e047ced9b0a6bbf0
56	5.522067	2606:4700:10::6816:826	2001:db8:1::1	QUIC	648	Protected Payload (KP0)
57	5.522266	2606:4700:10::6816:826	2001:db8:1::1	QUIC	86	Protected Payload (KP0)
58	5.522357	2001:db8:1::1	2606:4700:10::6816:826	QUIC	107	Protected Payload (KP0), DCID=0130dfc5a047e6acd230b5c5e047ced9b0a6bbf0
59	5.522398	2606:4700:10::6816:826	2001:db8:1::1	QUIC	86	Protected Payload (KP0)
60	5.522581	2606:4700:10::6816:826	2001:db8:1::1	QUIC	111	Protected Payload (KP0)
61	5.522629	2001:db8:1::1	2606:4700:10::6816:826	QUIC	109	Protected Payload (KP0), DCID=0130dfc5a047e6acd230b5c5e047ced9b0a6bbf0

Frame 47: 1292 bytes on wire (10336 bits), 1292 bytes captured (10336 bits) on interface
Ethernet II, Src: Apple_e8:7a:f3 (f0:18:98:e8:7a:f3), Dst: IBASETEC_87:59:70 (00:03:2d:87:59:70)
Internet Protocol Version 6, Src: 2001:db8:1::1, Dst: 2606:4700:10::6816:826
User Datagram Protocol, Src Port: 50280, Dst Port: 443
QUIC IETF

Frame (1292 bytes) Decrypted QUIC (1195 bytes)

```
0000  00 03 2d 87 59 70 f0 18 98 e8 7a f3 86 dd 60 03  ...Yp....z...`
0010  0e 00 04 d6 11 40 20 01 0d b8 00 01 00 00 00 00  .....@ .....
0020  00 00 00 00 00 01 26 06 47 00 00 10 00 00 00 00  .....&.G.....
0030  00 00 68 16 08 26 c4 68 01 bb 04 d6 90 e1 cc 00  ..h..&.h.....
0040  00 00 01 08 20 3f 9e 9f 68 69 82 74 00 00 44 bc  .... ?..hi.t..D
```

Further Reading:
www.tcpipguide.com/free/

wrap it up...

Questions?



Blog version of this talk available on my website:

www.thederpysage.com/blog/network101/

