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

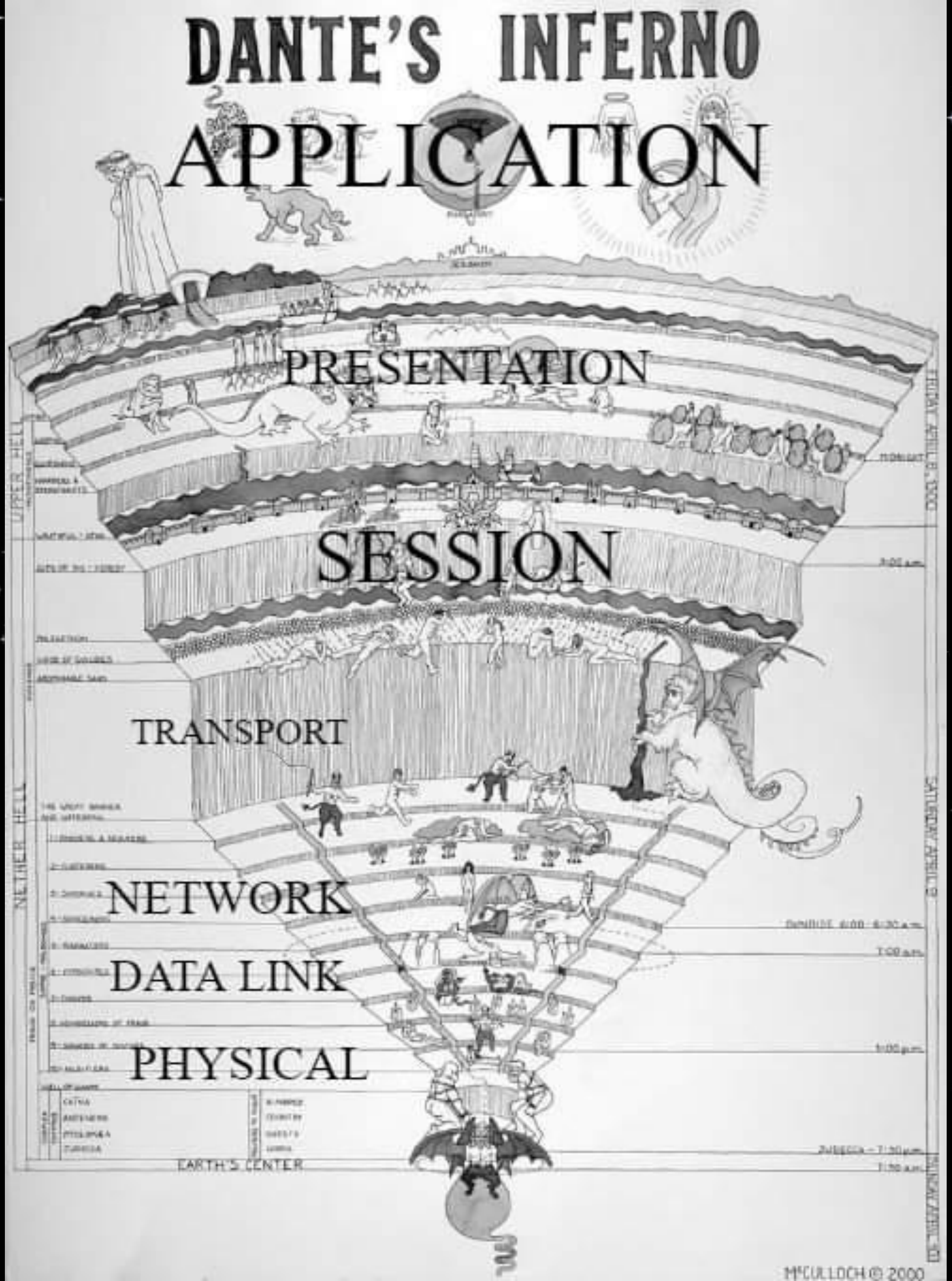
www.thederpysage.com

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
PRESENTATION
SESSION
TRANSPORT
NETWORK
DATA LINK
PHYSICAL
EARTH'S CENTER
McCULLOCH © 2000

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

Source: https://en.wikipedia.org/wiki/OSI_model

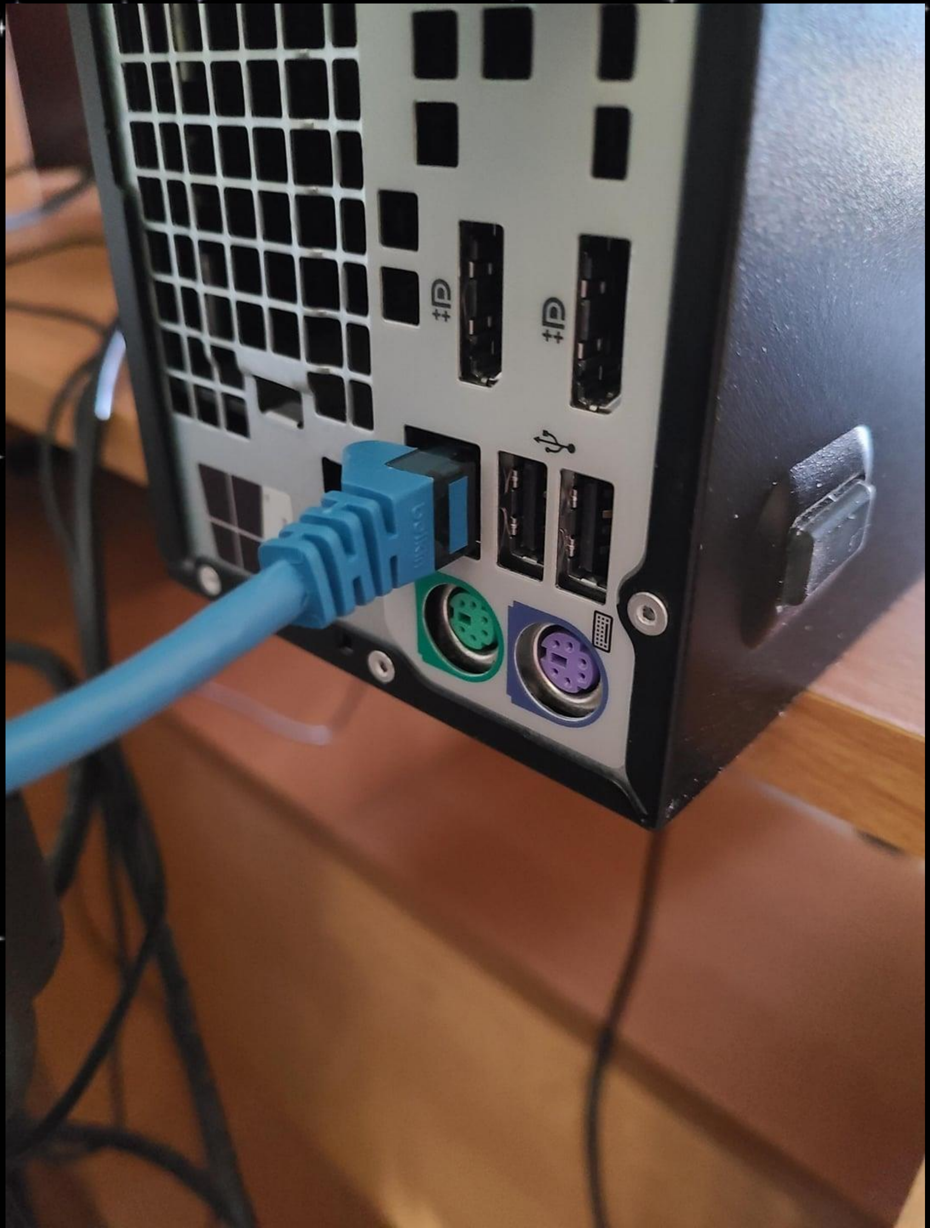
OSI model

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

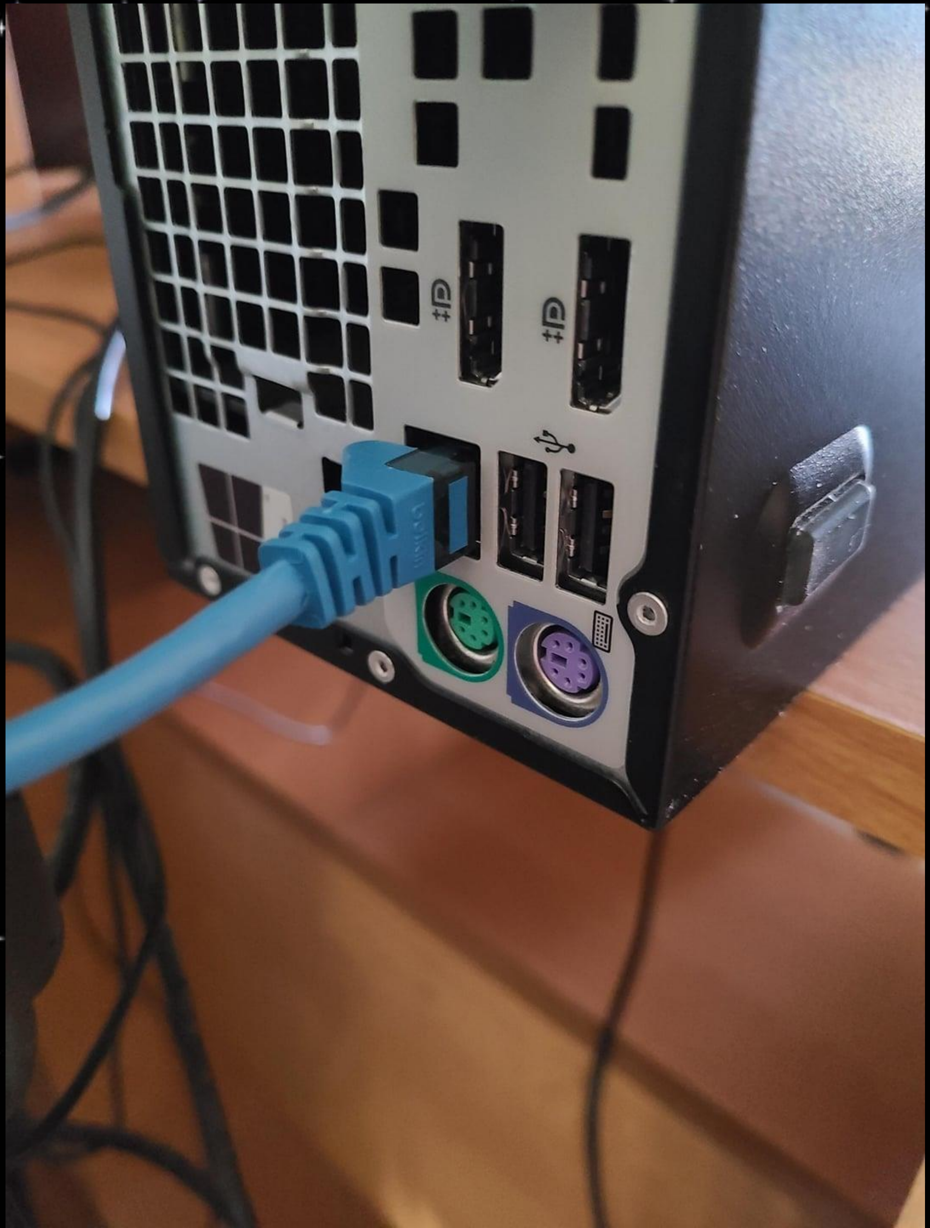
Frame 923: 465 bytes on wire (3720 bits), 465 bytes captured (3720 bits) on interface eth0, id 0
Ethernet II, Src: VMware_41:6b:4e (00:0c:29:41:6b:4e), Dst: VMware_b8:c6:d8 (00:0c:29:b8:c6:d8)
Internet Protocol Version 4, Src: 192.168.99.12, Dst: 192.168.99.11
Transmission Control Protocol, Src Port: 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
```

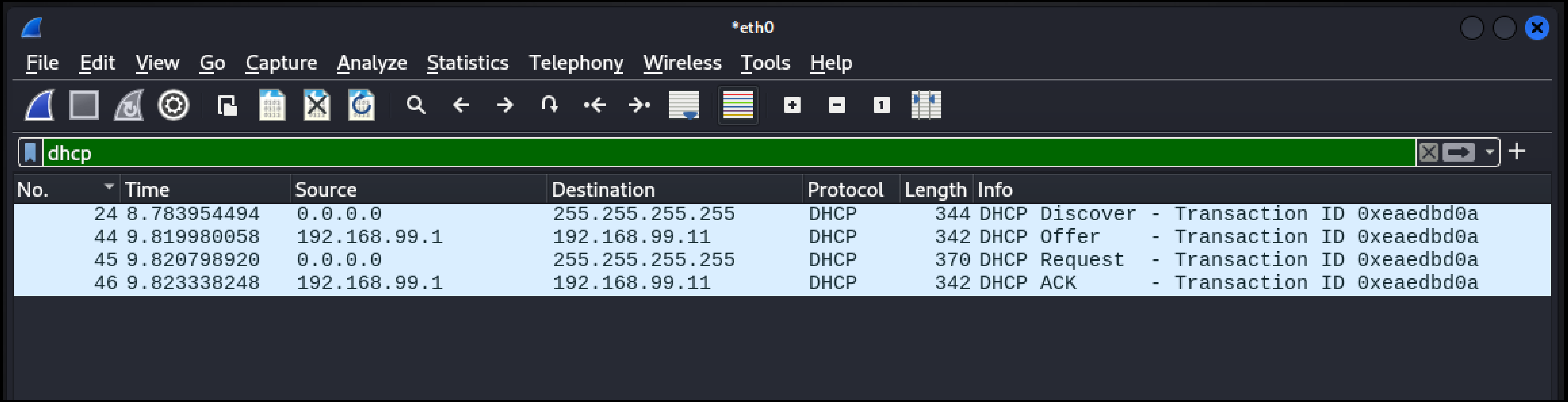
Network – IPv4

- 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.



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.

arp

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	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	257.024300269	VMware_41:6b:4e	VMware_ca:74:5e	ARP	60	Who has 192.168.99.12? Tell 192.168.99.12

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 Steremberg.

Search Stanford

10 results | clustering on | Search

Search The Web

10 results | clustering on | Search

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.

tcp.stream eq 0

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

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

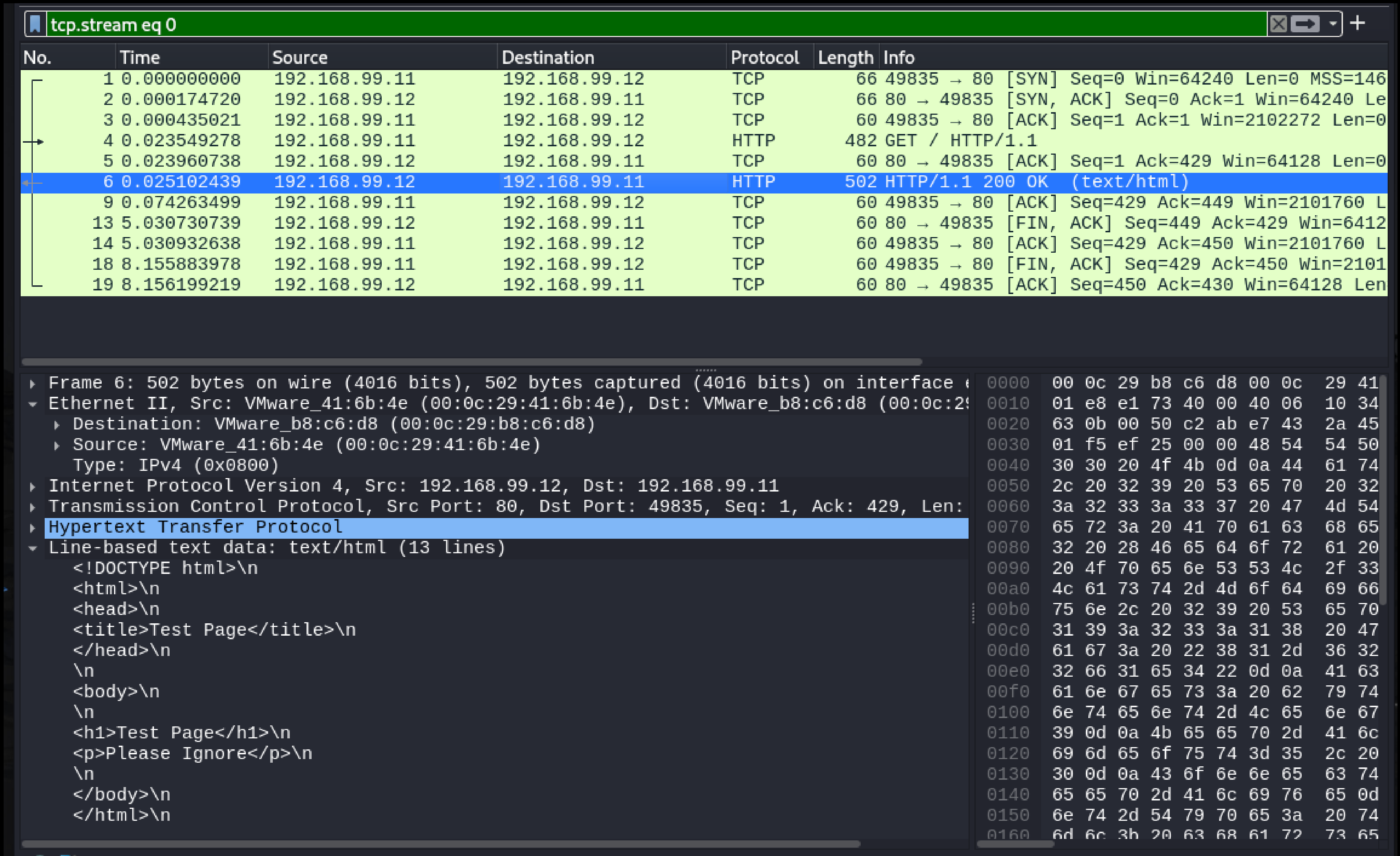
Data – TCP – HTTP

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



To Review…

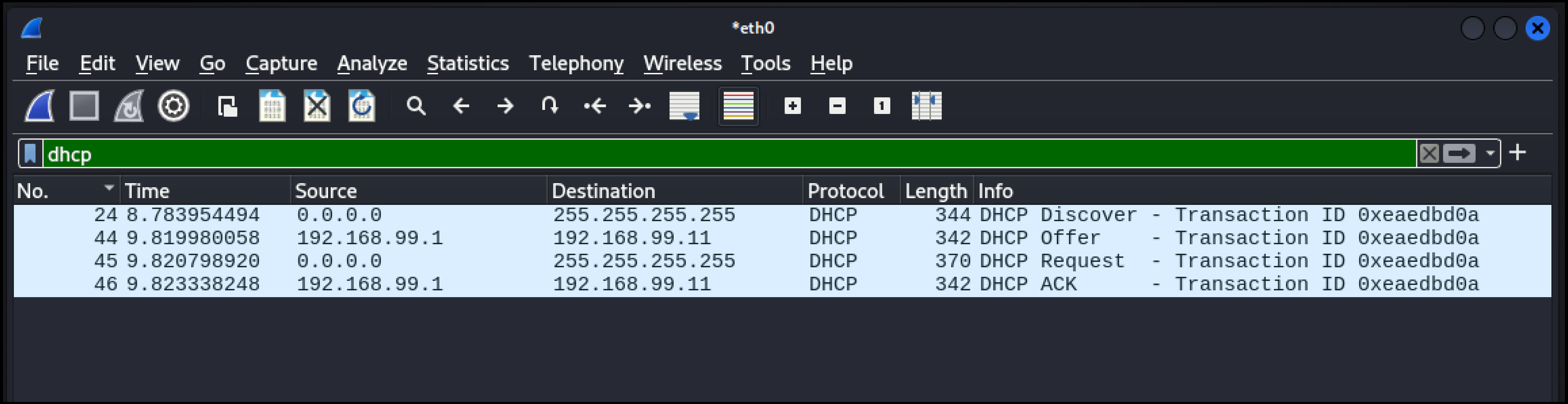
Frame 923: 465 bytes on wire (3720 bits), 465 bytes captured (3720 bits) on interface eth0, id 0
Ethernet II, Src: VMware_41:6b:4e (00:0c:29:41:6b:4e), Dst: VMware_b8:c6:d8 (00:0c:29:b8:c6:d8)
Internet Protocol Version 4, Src: 192.168.99.12, Dst: 192.168.99.11
Transmission Control Protocol, Src Port: 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.



- via 255.255.255.255

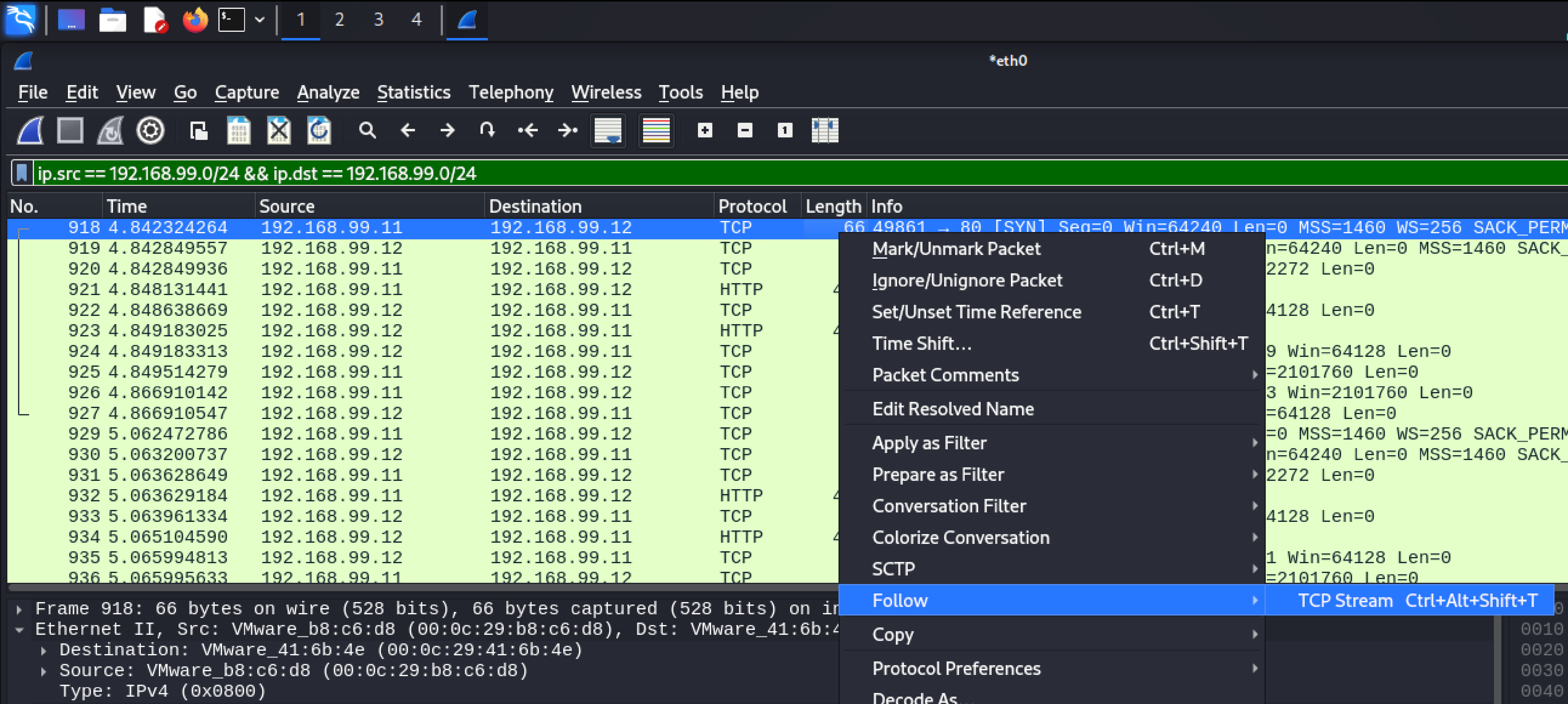
Data – UDP – DNS

- 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



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/sign
ed-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>
```

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
mxkL3tnZnwve2dgemhnIy9pamp
jL2l9amove2AvY2p7L2JqL2RhYH
gvY2BjIQ==

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

tcp.stream eq 0

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)

tcp.stream eq 1

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)

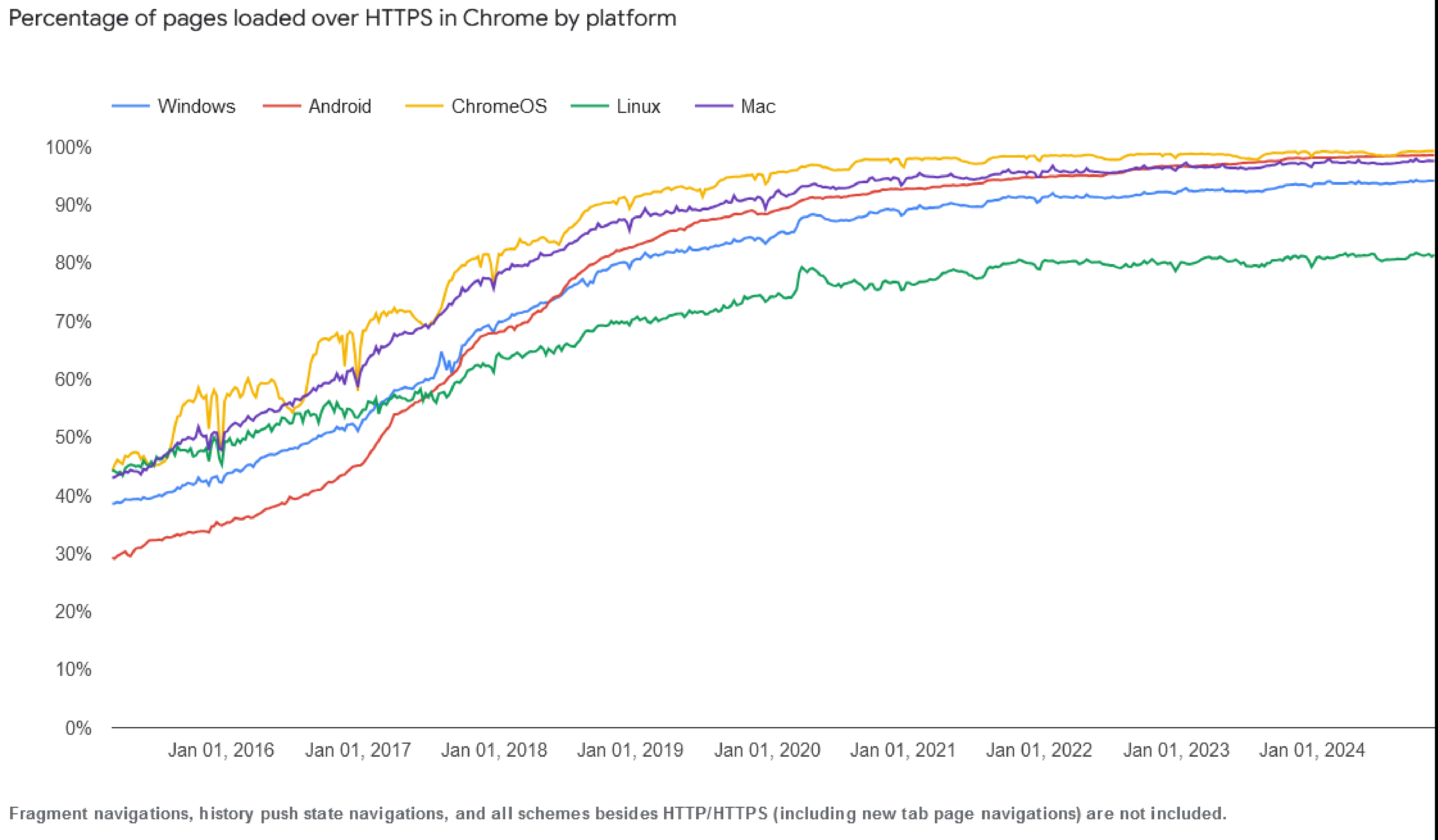
- The client now leads with all its supported ciphers and public keys in its ClientHello, allowing the Server to start with an Encrypted ServerHello and Certificate.

tcp.stream eq 8

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

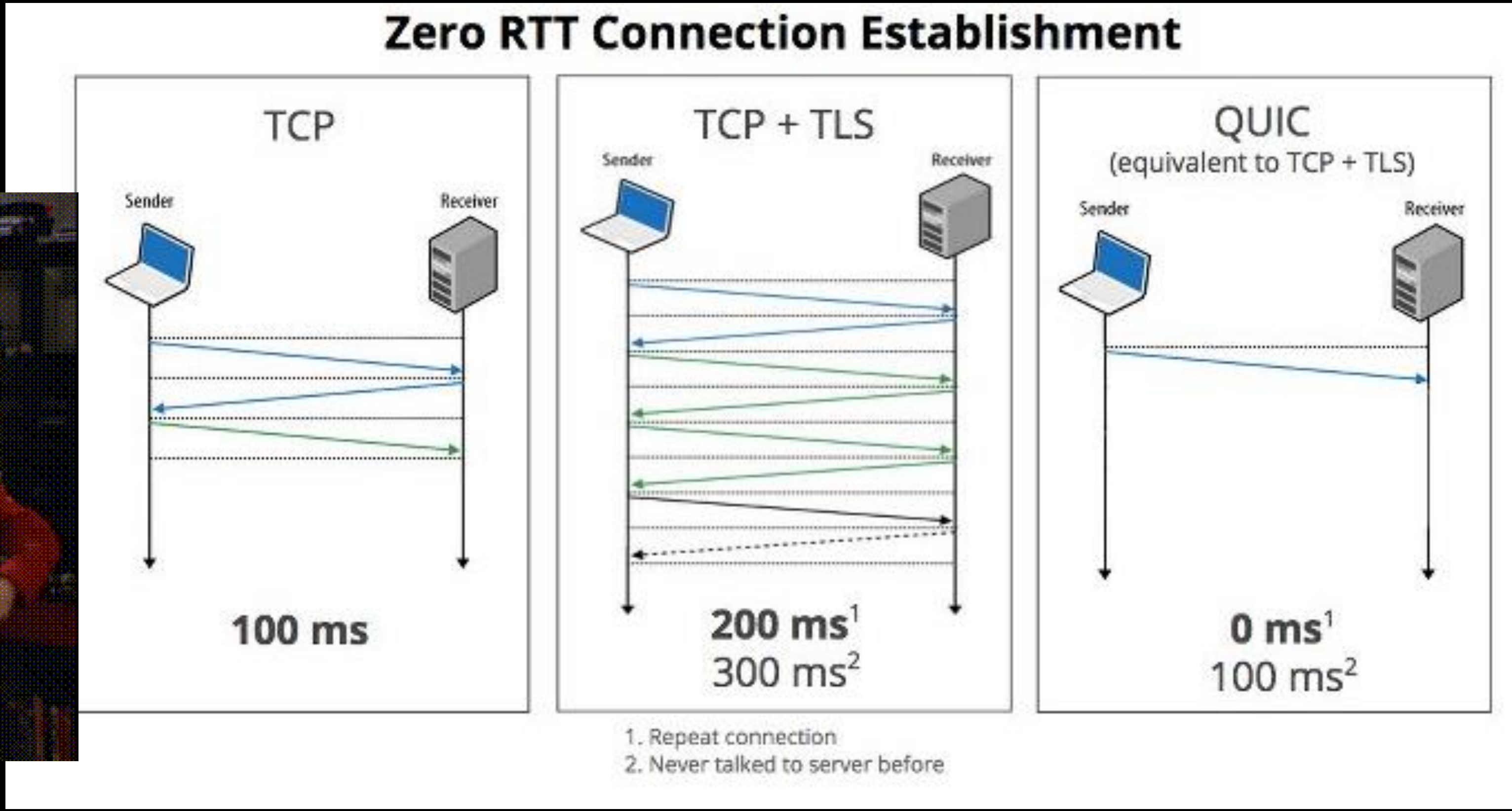
Data – TCP – HTTPS



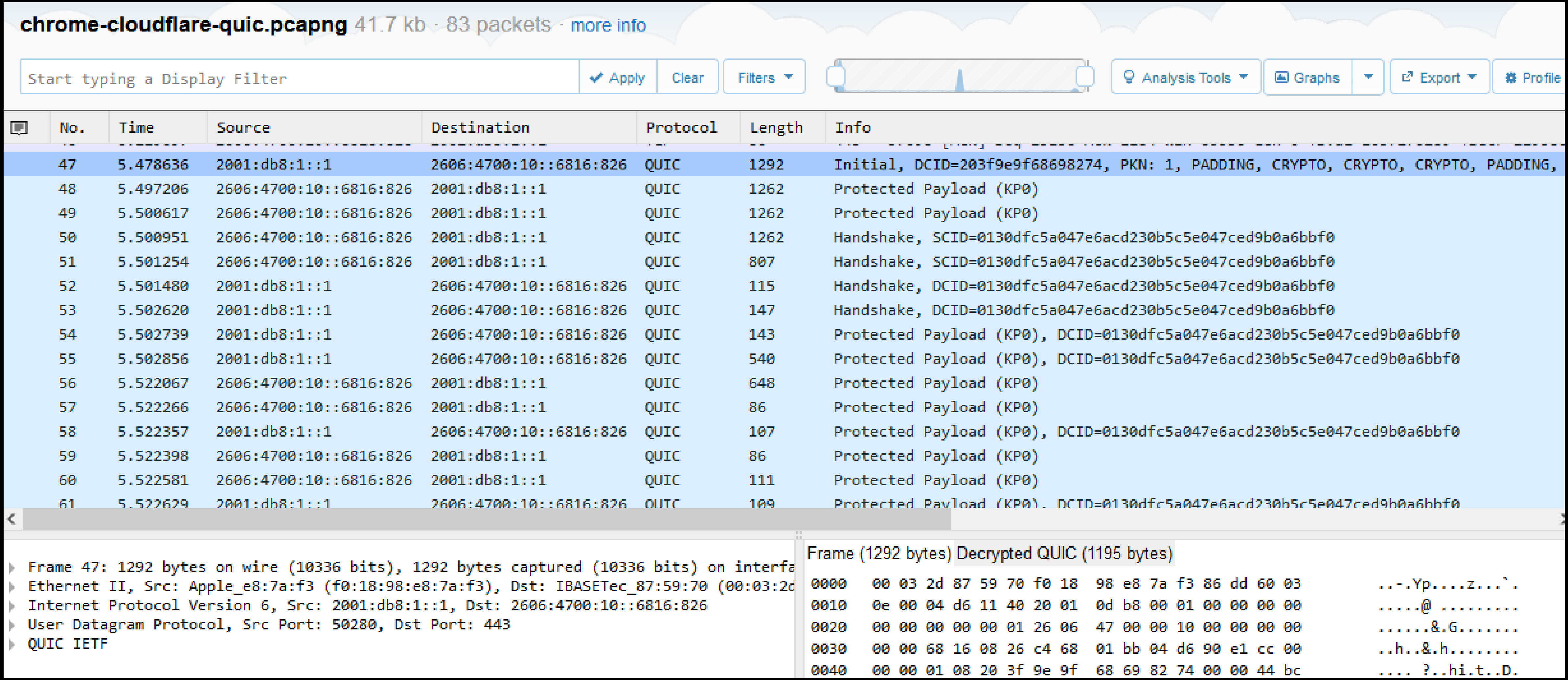
Source: https://transparencyreport.google.com/https/overview

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.



Transport – QUIC



Source: https://www.qacafe.com/resources/sample-captures-for-quic-doh-communityid-wpa3-cloudshark-3-10

Further Reading:
www.tcpipguide.com/free/

wrap it up...

Questions?

Blog version of this talk available on my website:

www.thederpysage.com/blog/network101/