Qassim University
Collage Of Computer
Computer Science dept.

جامعة القصيم
كلية الحاسب
قسم علوم الحاسب

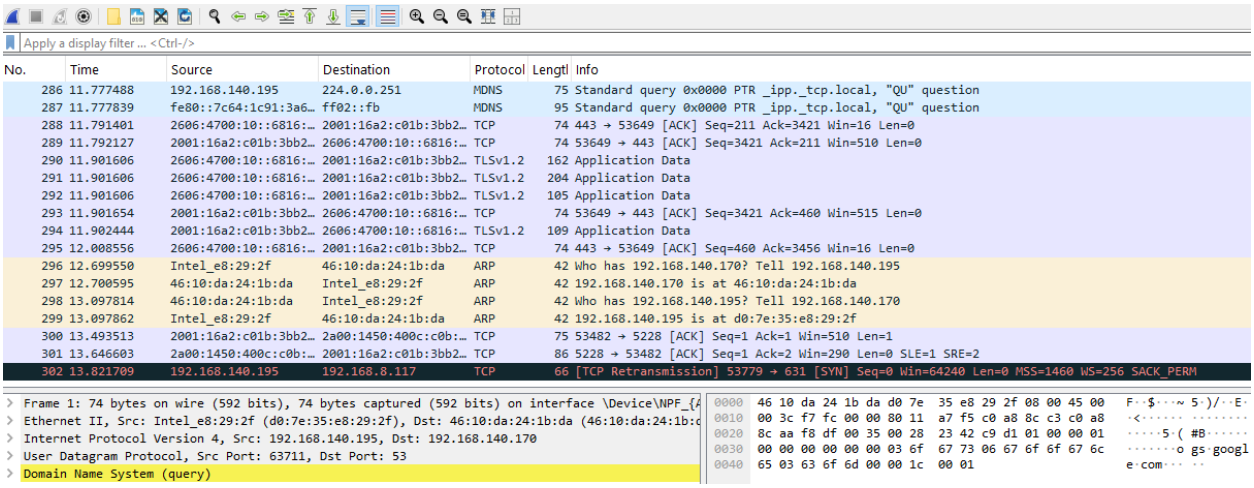
CS 471 – Web Technologies
The Internet Protocols – Lab Week 2
Semester (461)
2024/1445

Student:

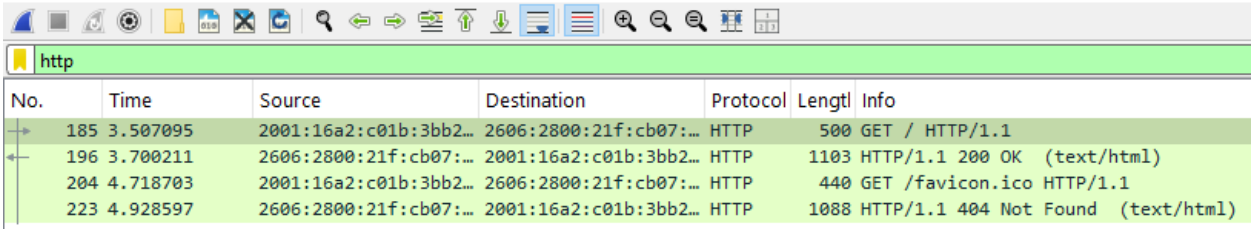
عبدالله البشري – 411109672

Part 1: Capturing HTTP Traffic.

Task 1: Start Wireshark and capture packets.



Task 2: Filter HTTP packets and analyze them.

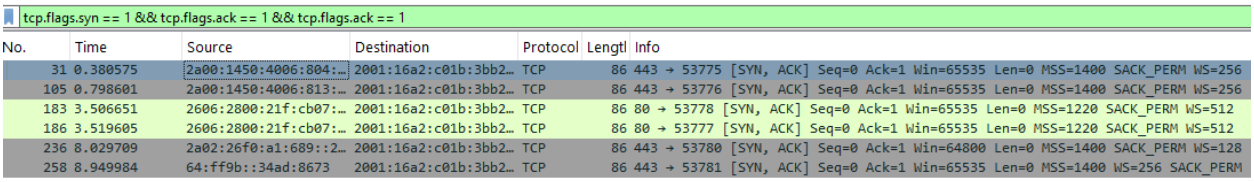


Part 2: Analyzing TCP/IP Traffic.

Task 1: Filter TCP packets

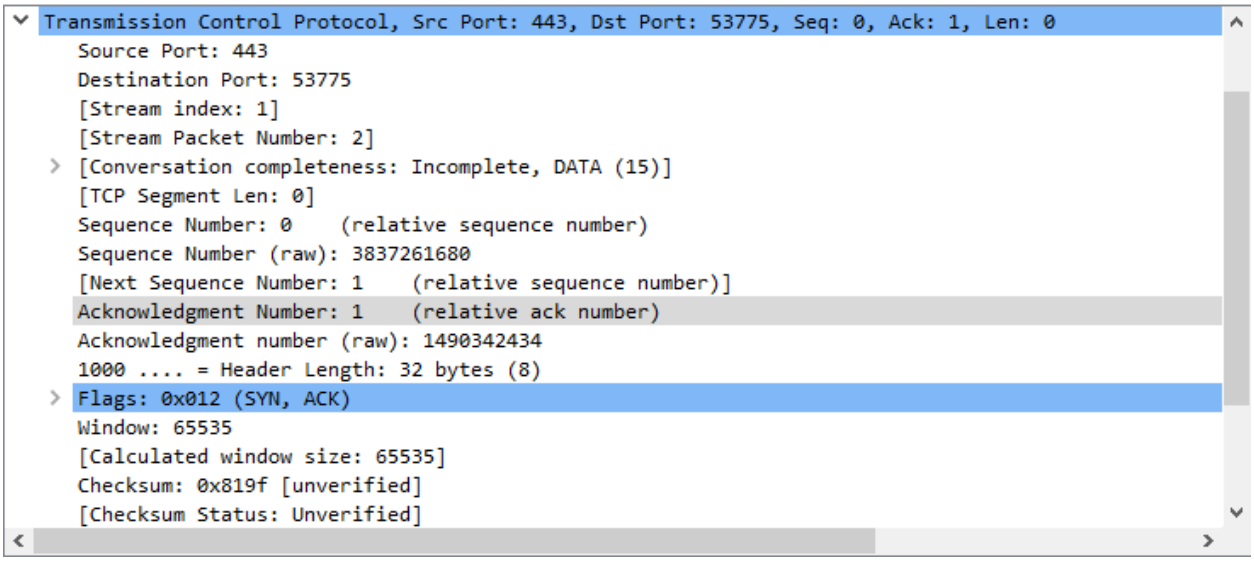
```
GET / HTTP/1.1
Host: example.com
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/128.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed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


HTTP/1.1 200 OK
Content-Encoding: gzip
Accept-Ranges: bytes
Age: 261566
Cache-Control: max-age=604800
Content-Type: text/html; charset=UTF-8
Date: Fri, 06 Sep 2024 17:01:36 GMT
Etag: "3147526947+gzip"
Expires: Fri, 13 Sep 2024 17:01:36 GMT
Last-Modified: Thu, 17 Oct 2019 07:18:26 GMT
Server: ECAcc (dcd/7D62)
Vary: Accept-Encoding
X-Cache: HIT
Content-Length: 648
```

Task 2: Analyze TCP handshake and investigate Data Transfer and Termination



tcp.flags.syn == 1 && tcp.flags.ack == 1 && tcp.flags.ack == 1

No.	Time	Source	Destination	Protocol	Lengtl	Info
31	0.380575	2a00:1450:4006:804:…	2001:16a2:c01b:3bb2…	TCP	86	443 → 53775 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1400 SACK_PERM WS=256
105	0.798601	2a00:1450:4006:813:…	2001:16a2:c01b:3bb2…	TCP	86	443 → 53776 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1400 SACK_PERM WS=256
183	3.506651	2606:2800:21f:cb07:…	2001:16a2:c01b:3bb2…	TCP	86	80 → 53778 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1220 SACK_PERM WS=512
186	3.519605	2606:2800:21f:cb07:…	2001:16a2:c01b:3bb2…	TCP	86	80 → 53777 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1220 SACK_PERM WS=512
236	8.029709	2a02:26f0:a1:689::2…	2001:16a2:c01b:3bb2…	TCP	86	443 → 53780 [SYN, ACK] Seq=0 Ack=1 Win=64800 Len=0 MSS=1400 SACK_PERM WS=128
258	8.949984	64:ff9b::34ad:8673	2001:16a2:c01b:3bb2…	TCP	86	443 → 53781 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1400 WS=256 SACK_PERM



```
Transmission Control Protocol, Src Port: 443, Dst Port: 53775, Seq: 0, Ack: 1, Len: 0
    Source Port: 443
    Destination Port: 53775
    [Stream index: 1]
    [Stream Packet Number: 2]
  > [Conversation completeness: Incomplete, DATA (15)]
    [TCP Segment Len: 0]
    Sequence Number: 0    (relative sequence number)
    Sequence Number (raw): 3837261680
    [Next Sequence Number: 1    (relative sequence number)]
    Acknowledgment Number: 1    (relative ack number)
    Acknowledgment number (raw): 1490342434
    1000 .... = Header Length: 32 bytes (8)
  > Flags: 0x012 (SYN, ACK)
    Window: 65535
    [Calculated window size: 65535]
    Checksum: 0x819f [unverified]
    [Checksum Status: Unverified]
```

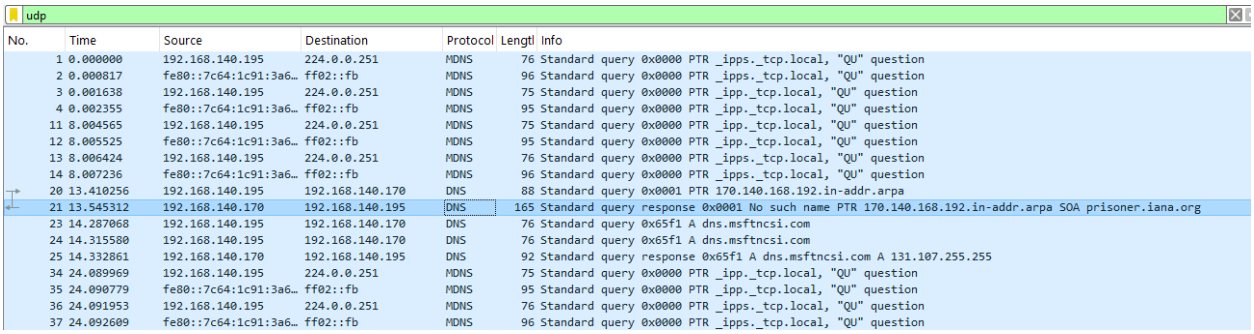
Part 3: Capturing and Analyzing UDP Traffic

Task 1: Generate UDP traffic and capture packets

Generating UDP traffic by using DNS queries:

```
C:\Users\Abod>nslookup
Default Server:  UnKnown
Address:  192.168.140.170

> google.com
Server:  UnKnown
Address:  192.168.140.170

Non-authoritative answer:
Name:    google.com
Addresses:  2a00:1450:4006:805::200e
          172.217.19.142

> _
```

Task 2: Filter and analysis UDP Packets



udp

No.	Time	Source	Destination	Protocol	Lengtl	Info
1	0.000000	192.168.140.195	224.0.0.251	MDNS	76	Standard query 0x0000 PTR _ipps._tcp.local, "QU" question
2	0.000817	fe80::7c64:1c91:3a6…	ff02::fb	MDNS	96	Standard query 0x0000 PTR _ipps._tcp.local, "QU" question
3	0.001638	192.168.140.195	224.0.0.251	MDNS	75	Standard query 0x0000 PTR _ipp._tcp.local, "QU" question
4	0.002355	fe80::7c64:1c91:3a6…	ff02::fb	MDNS	95	Standard query 0x0000 PTR _ipp._tcp.local, "QU" question
11	8.004565	192.168.140.195	224.0.0.251	MDNS	75	Standard query 0x0000 PTR _ipp._tcp.local, "QU" question
12	8.005525	fe80::7c64:1c91:3a6…	ff02::fb	MDNS	95	Standard query 0x0000 PTR _ipp._tcp.local, "QU" question
13	8.006424	192.168.140.195	224.0.0.251	MDNS	76	Standard query 0x0000 PTR _ipps._tcp.local, "QU" question
14	8.007236	fe80::7c64:1c91:3a6…	ff02::fb	MDNS	96	Standard query 0x0000 PTR _ipps._tcp.local, "QU" question
20	13.410256	192.168.140.195	192.168.140.170	DNS	88	Standard query 0x0001 PTR 170.140.168.192.in-addr.arpa
21	13.545312	192.168.140.170	192.168.140.195	DNS	165	Standard query response 0x0001 No such name PTR 170.140.168.192.in-addr.arpa SOA prisoner.iana.org
23	14.287068	192.168.140.195	192.168.140.170	DNS	76	Standard query 0x65f1 A dns.msftncsi.com
24	14.315580	192.168.140.195	192.168.140.170	DNS	76	Standard query 0x65f1 A dns.msftncsi.com
25	14.332861	192.168.140.170	192.168.140.195	DNS	92	Standard query response 0x65f1 A dns.msftncsi.com A 131.107.255.255
34	24.089969	192.168.140.195	224.0.0.251	MDNS	75	Standard query 0x0000 PTR _ipp._tcp.local, "QU" question
35	24.090779	fe80::7c64:1c91:3a6…	ff02::fb	MDNS	95	Standard query 0x0000 PTR _ipp._tcp.local, "QU" question
36	24.091953	192.168.140.195	224.0.0.251	MDNS	76	Standard query 0x0000 PTR _ipps._tcp.local, "QU" question
37	24.092609	fe80::7c64:1c91:3a6…	ff02::fb	MDNS	96	Standard query 0x0000 PTR _ipps._tcp.local, "QU" question

```
User Datagram Protocol, Src Port: 53, Dst Port: 62788
    Source Port: 53
    Destination Port: 62788
    Length: 131
    Checksum: 0xf970 [unverified]
    [Checksum Status: Unverified]
    [Stream index: 2]
    [Stream Packet Number: 2]
  > [Timestamps]
    UDP payload (123 bytes)
> Domain Name System (response)
```

As we can see, the UDP header is simpler than TCP header, which is typically contains source, destination ports, length and checksum. That's make UDP a simpler but less reliable protocol compared to TCP.

Part 4: Comparing TCP and UDP by filling in the following tables. Save your work (e.g., in an MS Word document), and upload it to your online git repo.

Task 1: Fill in the following table and provide reasons.

	TCP or UDP	Reason
Reliability and Connection Establishment	TCP	TCP is connection-oriented, meaning it establishes a connection through a three-way handshake (SYN, SYN-ACK, ACK). It ensures reliable delivery of packets by retransmitting lost packets and maintaining the order of data.
Data Integrity and Ordering	TCP	TCP ensures that data is delivered in the correct order, thanks to sequence numbers and acknowledgments. It also ensures that corrupted packets are detected and retransmitted if necessary. UDP, on the other hand, does not guarantee data integrity or order, as it is connectionless.

Task 2: Identify the use Cases and Performance of TCP and UDP.

	TCP	UDP
Use Cases	TCP is used in applications that require reliability and accuracy, such as web browsing (HTTP/HTTPS), email (SMTP/IMAP), file transfers (FTP), and remote access (SSH).	UDP is used in applications where speed is more important than reliability, such as live video/audio streaming, online gaming, VoIP, and DNS queries.
Performance	TCP has higher overhead due to connection setup, error correction, and flow control mechanisms. It is slower but ensures reliable and ordered delivery.	UDP has lower overhead as it does not establish connections or guarantee delivery. It is faster but less reliable than TCP, making it suitable for real-time applications.