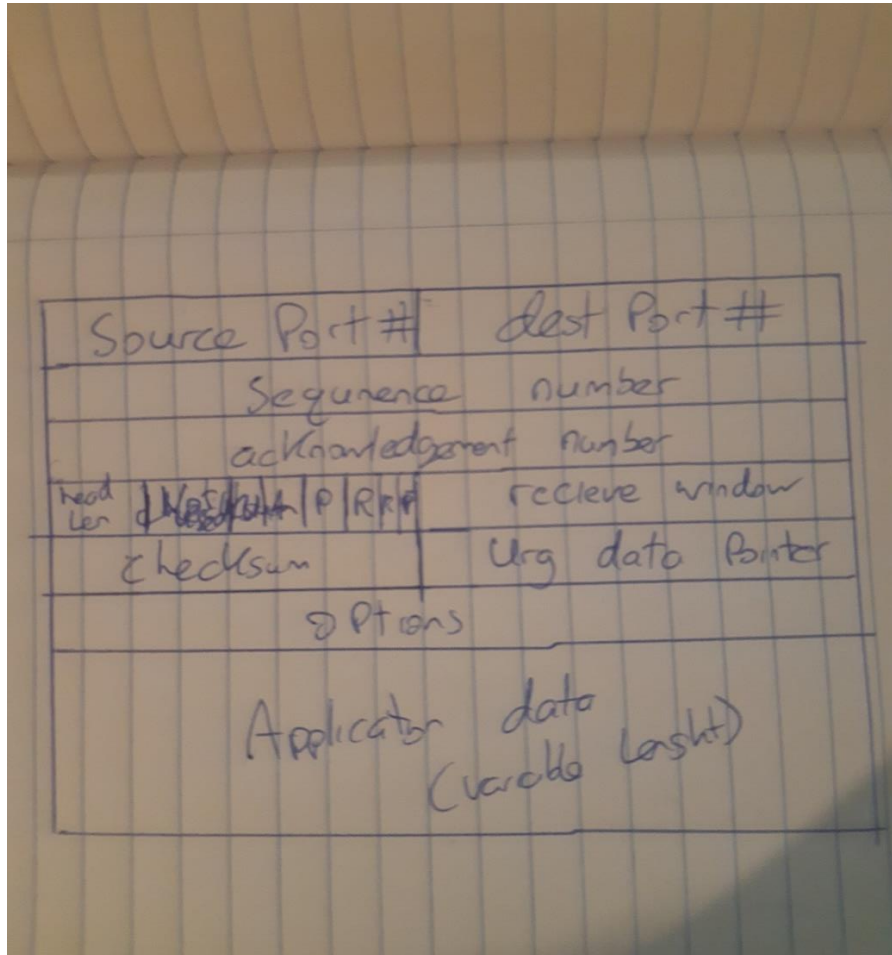


# Lab 5

## Q 1. TCP Header



## WireShark of TCP:

28	34.996408	192.168.0.100	64.185.181.238	TCP	66	50407	- 80	[FIN, ACK]	Seq=1	Ack=1	Win=65535	Len=0	TSval=115690168	TSecr=972288728
29	34.996443	192.168.0.100	23.15.7.51	TCP	66	50396	- 80	[FIN, ACK]	Seq=1	Ack=1	Win=65535	Len=0	TSval=115690168	TSecr=555799464
30	34.996478	192.168.0.100	23.15.7.51	TCP	66	50379	- 80	[FIN, ACK]	Seq=1	Ack=1	Win=65535	Len=0	TSval=115690168	TSecr=555804464
31	35.096566	192.168.0.100	72.21.91.19	TCP	54	50413	- 80	[FIN, ACK]	Seq=1	Ack=1	Win=65535	Len=0		
32	35.096593	192.168.0.100	205.251.253.243	TCP	66	50410	- 80	[FIN, ACK]	Seq=1	Ack=1	Win=65535	Len=0	TSval=115690169	TSecr=2924770787

<

> Frame 29: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface en0, id 0

> Ethernet II, Src: Apple\_35:f3:94 (c4:2c:03:35:f3:94), Dst: Netgear\_ca:25:9a (00:26:f2:ca:25:9a)

> Internet Protocol Version 4, Src: 192.168.0.100, Dst: 23.15.7.51

> Transmission Control Protocol, Src Port: 50396, Dst Port: 80, Seq: 1, Ack: 1, Len: 0

0000	00 26 f2 ca 25 9a c4 2c 03 35 f3 94 00 00 45 00	8 % , 5 ... E
0010	00 34 47 9d 40 00 40 06 00 09 c0 a8 00 64 17 0f	40 @ @ .....d ..
0020	07 33 c4 dc 00 50 ca ac 7a 0b 7c 89 48 30 80 11	3 ... P ... z   H0 ..
0030	ff ff df 74 00 00 01 01 08 0a 06 e5 4a b8 21 20	...t .....J ! ..
0040	d3 a8	..

**Source Port:** Port number used by the computer sending the TCP segment.

**Destination Port:** Port number used by the computer receiving the TCP segment.

**Sequence Number:** Used for segmenting data into TCP segments and reassembling on the other side.

**Data Offset/head Len:** Number of bytes into the TCP packet where the data can be found.

**U/A/P/R/S/F:** Are flags used to indicate parts of the 3 way handshake and the end of the TCP session.

**Window:** Number of octets in the TCP header

**Checksum:** A Cyclic Redundancy Check, used to verify the integrity of the data in the TCP

**Urgent Pointer:** Points to the end of the "Urgent" data in the packet (if URG flag is set)

## Q2. UDP Header

UDP: Segment Header

Source Port #	Destination Port #
Length	Checksum
data	

35	9.403626	172.253.116.188	192.168.1.106	TCP	66	5228 → 53469 [ACK] Seq=1 Ack=2 Win=265 Len=0 SLE=1 SRE=2
36	9.992050	192.168.1.106	74.125.193.93	UDP	1392	55458 → 443 Len=1350
37	9.993651	192.168.1.106	74.125.193.93	UDP	99	55458 → 443 Len=57
38	10.023599	74.125.193.93	192.168.1.106	UDP	67	443 → 55458 Len=25
39	10.043660	74.125.193.93	192.168.1.106	UDP	110	443 → 55458 Len=68
40	10.043660	74.125.193.93	192.168.1.106	UDP	67	443 → 55458 Len=25
41	10.045872	192.168.1.106	74.125.193.93	UDP	75	55458 → 443 Len=33
42	15.752213	ASUSTekC_38:d4:b2	Broadcast	ARP	42	Who has 192.168.1.1? Tell 192.168.1.106
43	15.752923	Tp-LinkT_e4:11:8c	ASUSTekC_38:d4:b2	ARP	60	192.168.1.1 is at 30:b5:c2:e4:11:8c
44	16.752335	192.168.1.102	239.255.255.250	SSDP	446	NOTIFY * HTTP/1.1
45	16.752335	192.168.1.102	239.255.255.250	SSDP	455	NOTIFY * HTTP/1.1
46	16.753126	192.168.1.102	239.255.255.250	SSDP	490	NOTIFY * HTTP/1.1
47	16.753126	192.168.1.102	239.255.255.250	SSDP	490	NOTIFY * HTTP/1.1
48	16.753865	192.168.1.102	239.255.255.250	SSDP	494	NOTIFY * HTTP/1.1

> Frame 36: 1392 bytes on wire (11136 bits), 1392 bytes captured (11136 bits) on interface \Device\NPF\_{71B2E5E4-3513-4F08-9D89-D60747C708F9}, id 0

> Ethernet II, Src: ASUSTekC\_38:d4:b2 (b0:6e:bf:38:d4:b2), Dst: Tp-LinkT\_e4:11:8c (30:b5:c2:e4:11:8c)

> Internet Protocol Version 4, Src: 192.168.1.106, Dst: 74.125.193.93

> User Datagram Protocol, Src Port: 55458, Dst Port: 443

> Data (1350 bytes)

```

0000 30 b5 c2 e4 11 8c b0 6e bf 38 d4 b2 08 00 45 00 0...n 8...E
0010 05 02 01 a4 00 00 11 00 00 c0 a8 01 6a 4a 7d b...@...:j}
0020 c1 5d d8 a2 01 bb 05 4e d3 4c 54 97 89 6d 07 b0...N...LT...m{
0030 86 2c 42 58 c8 42 f3 85 9a 18 3e 74 7c 09 fd 44...BX.B...>t}...D
0040 04 f1 35 13 69 8b 86 5a c2 20 f6 0a 2d 72 53 07...5.i...Z...-rS-
0050 19 2c 09 4f 4c fe 61 dd 92 d9 23 74 ac 64 b8 1a...OL.a...#t.d...
0060 b9 0e a4 59 b2 4e 82 0f ec 3b 95 57 2e fb 29 d7...Y.N...;W.-)
0070 26 f9 fc 3e 87 4f d0 ab e5 3c 0b 1f 1e 56 f9 33...>O...<...V-3
0080 a3 f0 52 f3 c5 c3 9a 74 46 88 43 cd 6d 8f ae 1c...R...t F.C.m...
0090 8e 67 f1 75 13 3c 84 29 10 3f 4b de 1f a9 c3 1c...g.u.<...?K....
00a0 0e 7c 02 17 7a 69 e5 78 b4 a3 e5 89 62 73 9e 3e...|...zi.x...bs>
00b0 d4 be b9 a2 78 e3 77 f8 8d e6 fd 37 e7 fd 83 c1...x.w...7....
00c0 ac b4 9c 13 6a ec 60 78 4d 79 ac 87 41 a5 cd 56...j...x My...A-V
00d0 1b 39 cc fc 4a 25 59 0d ad 58 9e 1a 95 b1 5e d1...9...7W...X...^
00e0 55 a8 70 0a f5 4a 64 b0 3e 03 27 a5 3b 71 1c 5e...U.p...3d...>...q^
00f0 03 86 ca ec 85 12 e4 0d 34 c5 80 c5 c9 6d 44 f3........4...^BD
0100 90 cd a7 51 94 80 20 cf b3 a0 f9 28 1d d0 6d 87...Q...<...(-...
0110 9e 7a b9 c5 69 6b 6c bd d9 c5 0b 56 5c 5d ba 9e...z...ik1...V\]..
0120 dd 9a 3c 77 12 49 65 41 c2 82 6e 9c aa 56 06 df...<w.IeA...n.V...
0130 6d 3e 93 5f ba 3f f4 fc c1 a8 f1 ae e5 1c 8c dc...m...?...-.....
0140 2c 34 79 6b c8 a6 4e d6 8d 8f 2c 56 2f 3b b9 bf...4yK...N...;V/;...
0150 cd 77 2b 7f 15 09 69 3e c9 96 25 b1 ee 1a 6e c5...w+...i>...%...n-
0160 7d e2 3e 39 ba df 1f 41 74 7c 66 3a f9 23 d8 96...>9...A t[f:#+...
0170 68 af 9b 57 9b 53 2d a2 11 60 e3 4b 04 35 f4 10...h...W-S...^K-S...
0180 13 cf fe 3d 34 72 fa f3 d8 67 7a f5 13 94 c1 c7...=4n...gz....
0190 16 ff b8 59 c7 9a 4e ec 5d 62 60 5e ff 5b 50 32...Y-N...jb^-[P2

```

Ethernet: <live capture in progress>
Packets: 668 · Displayed: 668 (100.0%)
Profile: Default

**Source Port:** Port number used by the computer sending the TCP segment.

**Destination Port:** Port number used by the computer receiving the TCP segment.

**Length:** Length in bytes the UDP header and UDP data. The type of IPv can determine this size.

**Checksum:** Error checking of the Header and data.

### Q3. Checksum Example UDP, image below

C0a8 = 1100 0000 1010 1000 +

0065 = 0000 0000 0110 0101

= **1100 0001 0000 1101 +**

C0a8 = 1100 0000 1010 1000

= **1000 0001 1011 0101**

**+ 1**

= **1000 0001 1011 0110 +**

000a = 0000 0000 0000 1010

= **1000 0001 1100 0000 +**

0011 = 0000 0000 0001 0001

= **1000 0001 1101 0001 +**

C2f1 = 1100 0010 1111 0001

= **0100 0100 1100 0010**

**+ 1**

= **0100 0100 1100 0011 +**

13e7 = 0001 0011 1110 0111

= **0101 1000 1010 1010 +**

0019 = 0000 0000 0001 1001

= **0101 1000 1100 0011 +**

0100 = 0000 0001 0000 0000

= **0101 1001 1100 0011 +**

0300 = 0000 0011 0000 0000

= **0101 1100 1100 0011 +**

0000 = 0000 0000 0000 0000

0800 = 0000 1000 0000 0000

= **0110 0100 1100 0011 +**

1182 = 0001 0001 1000 0010

= **0111 0110 0100 0101** +

264e = 0010 0110 0100 1110

= **1001 1100 1001 0011** +

0000 = 0000 0000 0000 0000

D20c = 1101 0010 0000 1100

= **0110 1110 1001 1111**

+ **1**

= **0110 1110 1010 0000** +

0034 = 0000 0000 0011 0100

= **0110 1110 1101 0100**

**Flip:** 1001 0001 0010 1011

**In Hexa:** 912B

12	0.200396	192.168.0.10	192.168.0.101	HART_...	59	Pass Through Response, Sequence Number 7
13	0.308435	192.168.0.101	192.168.0.10	HART_...	59	Pass Through Request, Sequence Number 8
14	0.311965	192.168.0.10	192.168.0.101	HART_...	85	Pass Through Response, Sequence Number 8
15	0.360352	192.168.0.101	192.168.0.10	HART_...	59	Pass Through Request, Sequence Number 9
16	0.364185	192.168.0.10	192.168.0.101	HART_...	82	Pass Through Response, Sequence Number 9
17	0.412411	192.168.0.101	192.168.0.10	HART_...	59	Pass Through Request, Sequence Number 10
18	0.415960	192.168.0.10	192.168.0.101	HART_...	93	Pass Through Response, Sequence Number 10
19	0.464293	192.168.0.101	192.168.0.10	HART_...	59	Pass Through Request, Sequence Number 11
20	0.467833	192.168.0.10	192.168.0.101	HART_...	74	Pass Through Response, Sequence Number 11

User Datagram Protocol, Src Port: 49905, Dst Port: 5095	
Source Port: 49905	
Destination Port: 5095	
Length: 25	
Checksum: 0x9211 [unverified]	
[Checksum Status: Unverified]	
[Stream index: 1]	
> [Timestamps]	
HART_IP Protocol, Pass Through Request, Sequence Number 8	
HART_IP Header	
Version: 1	
Message Type: Request (0)	
Message ID: Pass Through (3)	
Status: 0	
Sequence Number: 8	
Message Length: 17	
HART_IP Body, Pass Through, Request	
Frame Type: STX	
Long Address: 264e0000d2	
Command: 12	
Length: 0	
Checksum: 0x34	

0000	00 26 16 00 00 d2 00 0c	29 50 a9 fc 08 00 45 00	..&.....)P....E..
0010	00 2d 00 00 40 00 40 11	b9 00 c0 a8 00 65 c0 a8	....@.@.....e..
0020	00 0a c2 f1 13 e7 00 19	92 11 01 00 03 00 00 08	.....
0030	00 11 82 26 4e 00 00 d2	0c 00 34	...&N....4

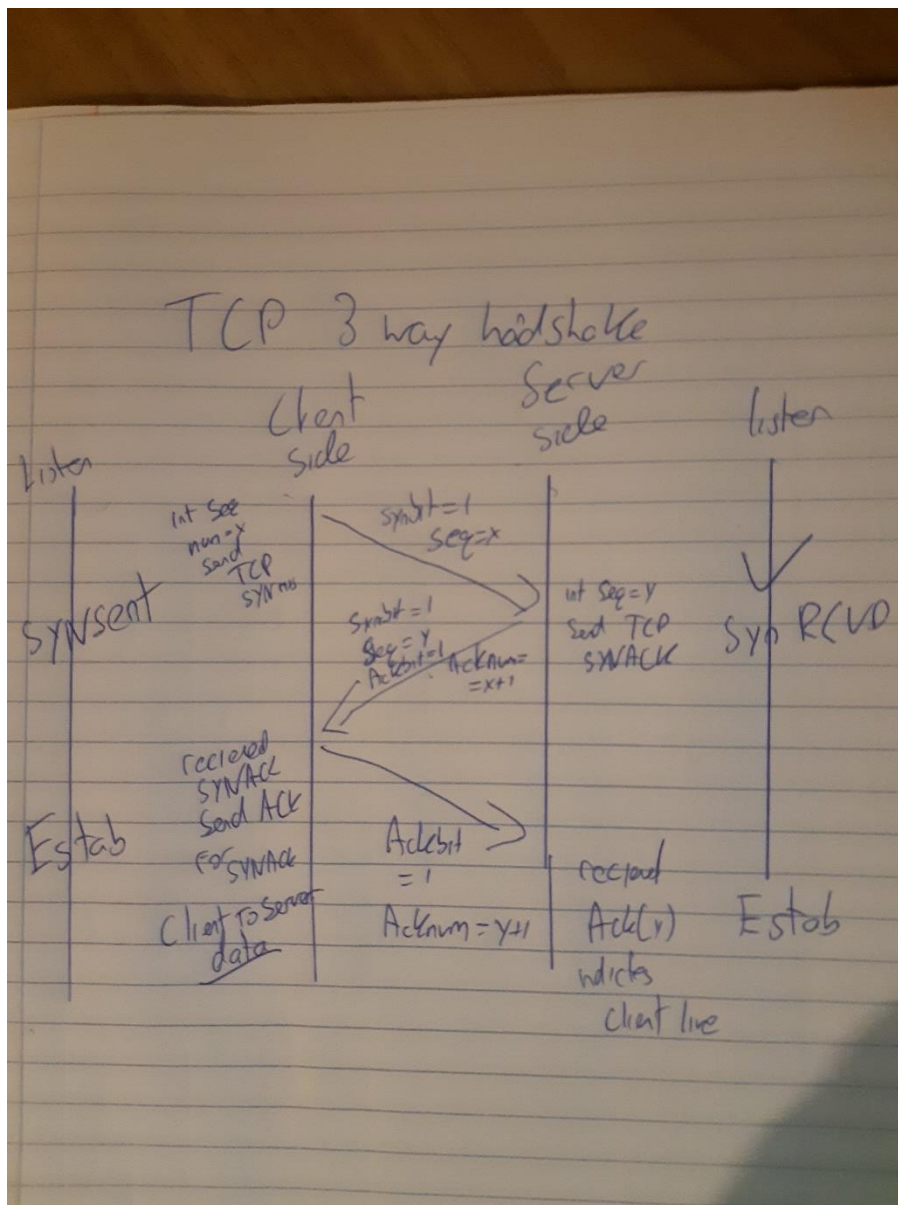
**Actual checksum: 9211**

The result I got was only marginally off(912B – 9211) so I assume it was a calculating error but the method I used was correct.

## Q4. Checksum Example UDP, image below

After capturing some streaming packets, I found that the majority of the packets were UDP. The packets were used for transferring data for the stream to the computer, while UDP may lose information or possibly display it out of order the brain and can make up for this error when looking at the frames as long as it is not that bad. UDP is used for streaming because it allows faster transfer of data.

## Q5. TCP – 3 way handshake

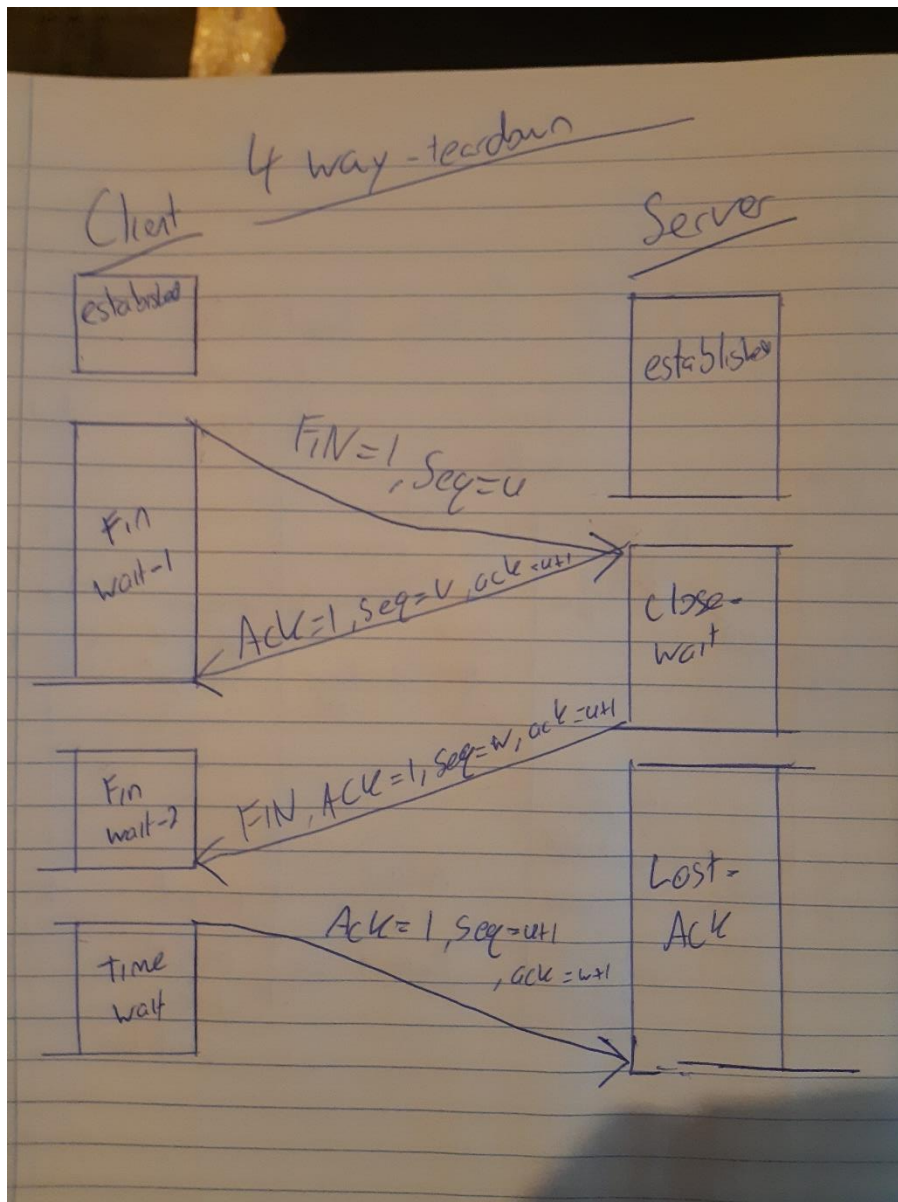


Wireshark Packet capture of a 3 way handshake:

110	67.204568	192.168.0.101	192.168.0.10	TCP	54 49559 → 5094 [ACK] Seq=175 Ack=375 Win=5888 Len=0
111	70.826237	192.168.0.101	192.168.0.10	HART_...	62 Session Close Request, Sequence Number 13
112	70.826433	192.168.0.101	192.168.0.10	TCP	54 49559 → 5094 [FIN, ACK] Seq=183 Ack=375 Win=5888 Len=0
113	70.827652	192.168.0.10	192.168.0.101	HART_...	62 Session Close Response, Sequence Number 13



## Q6. TCP – 4 way teardown:



## Wireshark Packet capture of a 4 way teardown:

72	35.468941	VMware_50:a9:fc	Rosemoun_00:00:d2	ARP	42	192.168.0.101 is at 00:0c:29:50:a9:fc
73	36.731287	192.168.0.101	192.168.0.10	TCP	66	49559 → 5094 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM=1 WS=64
74	36.731776	192.168.0.10	192.168.0.101	TCP	66	5094 → 49559 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460 SACK_PERM=1 WS=4
75	36.731914	192.168.0.101	192.168.0.10	TCP	54	49559 → 5094 [ACK] Seq=1 Ack=1 Win=5888 Len=0
76	36.731973	192.168.0.101	192.168.0.10	HART	67	Session Initiate Request. Sequence Number 2