Decrypting TLS and HTTP(s) using Wireshark ++

Assignment 4

Kamal Shrestha

CS21MTECH16001

Feb 27, 2022

PART - A

Decrypt TLS handshake and HTTPS messages between your browser and the web server of Bank X

Steps performed:

1. Setting SSLKEYLOGFILE environment variable, launched google chrome and Wireshark:

```
export SSLKEYLOGFILE="/home/kamal/sslkeyfile.log"

google-chrome

sudo wireshark
```

2. Packet Capturing started in Wireshark

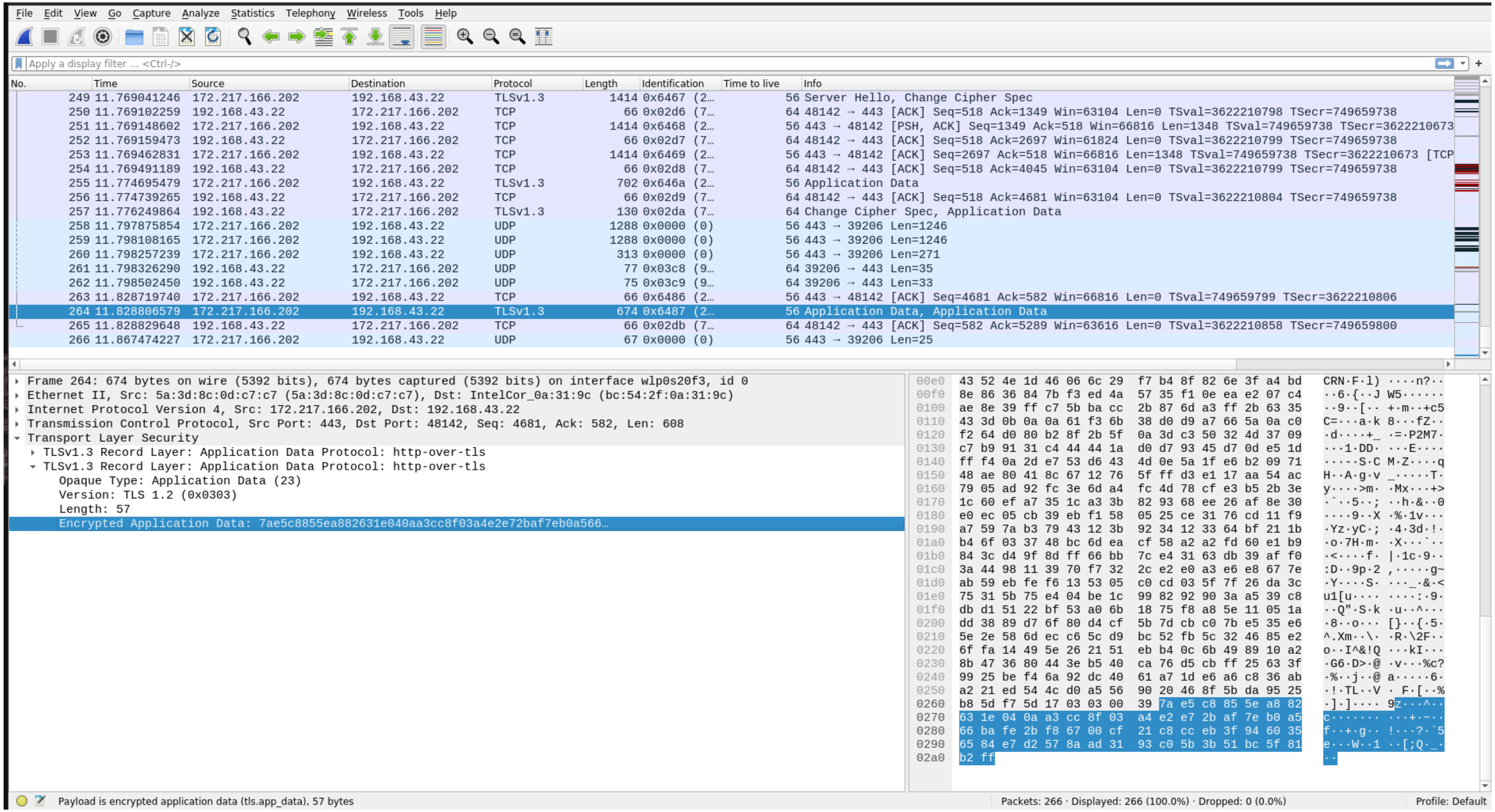
3. Opened http://netbanking.hdfcbank.com/ in opened chrome browser as, 16001%4 + 1 = 2 => HDFC

4. Entered random Username and Password.

5. Packet capture stopped and saved the trace files (CS21MTECH16001.pcapng).

6. Added the SSL Key log file in Wireshark to decrypt the TLS and HTTPs messages.

Before adding in the key log file, all the messages (handshake messages, Application data) were in encrypted format as shown with the ss below:



```
No.     Time         Source           Destination      Protocol  Length Identification Time to live Info
    249 11.769041246 172.217.166.202  192.168.43.22    TLSv1.3    1414 0x6467 (2…        56 Server Hello, Change Cipher Spec
    250 11.769102259 192.168.43.22    172.217.166.202  TCP          66 0x02d6 (7…        64 48142 → 443 [ACK] Seq=518 Ack=1349 Win=63104 Len=0 TSval=3622210798 TSecr=749659738
    251 11.769148602 172.217.166.202  192.168.43.22    TCP        1414 0x6468 (2…        56 443 → 48142 [PSH, ACK] Seq=1349 Ack=518 Win=66816 Len=1348 TSval=749659738 TSecr=3622210673
    252 11.769159473 192.168.43.22    172.217.166.202  TCP          66 0x02d7 (7…        64 48142 → 443 [ACK] Seq=518 Ack=2697 Win=61824 Len=0 TSval=3622210799 TSecr=749659738
    253 11.769462831 172.217.166.202  192.168.43.22    TCP        1414 0x6469 (2…        56 443 → 48142 [ACK] Seq=2697 Ack=518 Win=66816 Len=1348 TSval=749659738 TSecr=3622210673 [TCP
    254 11.769491189 192.168.43.22    172.217.166.202  TCP          66 0x02d8 (7…        64 48142 → 443 [ACK] Seq=518 Ack=4045 Win=63104 Len=0 TSval=3622210799 TSecr=749659738
    255 11.774695479 172.217.166.202  192.168.43.22    TLSv1.3     702 0x646a (2…        56 Application Data
    256 11.774739265 192.168.43.22    172.217.166.202  TCP          66 0x02d9 (7…        64 48142 → 443 [ACK] Seq=518 Ack=4681 Win=63104 Len=0 TSval=3622210804 TSecr=749659738
    257 11.776249864 192.168.43.22    172.217.166.202  TLSv1.3     130 0x02da (7…        64 Change Cipher Spec, Application Data
    258 11.797875854 172.217.166.202  192.168.43.22    UDP        1288 0x0000 (0)        56 443 → 39206 Len=1246
    259 11.798108165 172.217.166.202  192.168.43.22    UDP        1288 0x0000 (0)        56 443 → 39206 Len=1246
    260 11.798257239 172.217.166.202  192.168.43.22    UDP         313 0x0000 (0)        56 443 → 39206 Len=271
    261 11.798326290 192.168.43.22    172.217.166.202  UDP          77 0x03c8 (9…        64 39206 → 443 Len=35
    262 11.798502450 192.168.43.22    172.217.166.202  UDP          75 0x03c9 (9…        64 39206 → 443 Len=33
    263 11.828719740 172.217.166.202  192.168.43.22    TCP          66 0x6486 (2…        56 443 → 48142 [ACK] Seq=4681 Ack=582 Win=66816 Len=0 TSval=749659799 TSecr=3622210806
    264 11.828806579 172.217.166.202  192.168.43.22    TLSv1.3     674 0x6487 (2…        56 Application Data, Application Data
    265 11.828829648 192.168.43.22    172.217.166.202  TCP          66 0x02db (7…        64 48142 → 443 [ACK] Seq=582 Ack=5289 Win=63616 Len=0 TSval=3622210858 TSecr=749659800
    266 11.867474227 172.217.166.202  192.168.43.22    UDP          67 0x0000 (0)        56 443 → 39206 Len=25
```

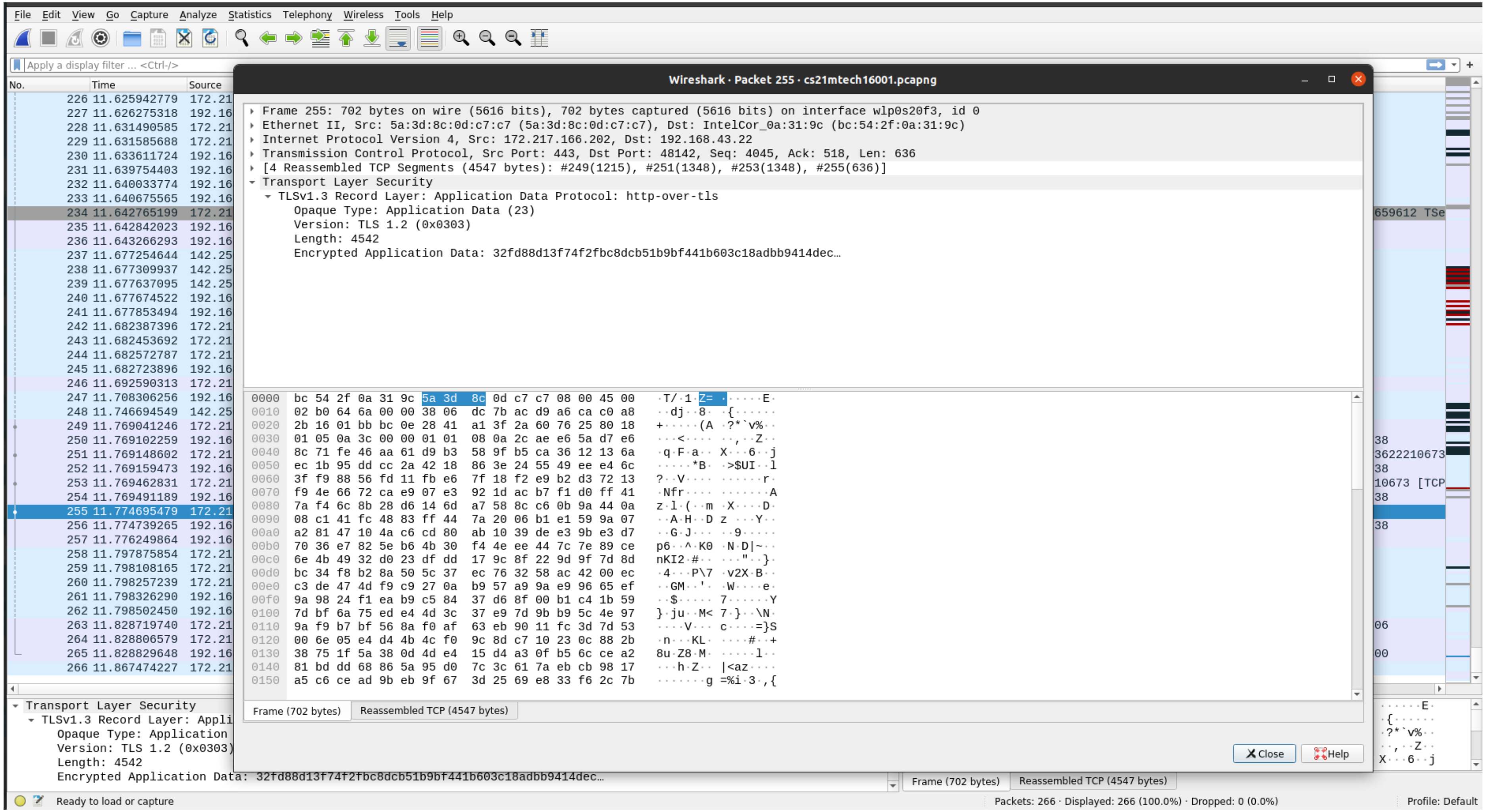
```
› Frame 264: 674 bytes on wire (5392 bits), 674 bytes captured (5392 bits) on interface wlp0s20f3, id 0
› Ethernet II, Src: 5a:3d:8c:0d:c7:c7 (5a:3d:8c:0d:c7:c7), Dst: IntelCor_0a:31:9c (bc:54:2f:0a:31:9c)
› Internet Protocol Version 4, Src: 172.217.166.202, Dst: 192.168.43.22
› Transmission Control Protocol, Src Port: 443, Dst Port: 48142, Seq: 4681, Ack: 582, Len: 608
⌄ Transport Layer Security
  › TLSv1.3 Record Layer: Application Data Protocol: http-over-tls
  ⌄ TLSv1.3 Record Layer: Application Data Protocol: http-over-tls
      Opaque Type: Application Data (23)
      Version: TLS 1.2 (0x0303)
      Length: 57
      Encrypted Application Data: 7ae5c8855ea882631e040aa3cc8f03a4e2e72baf7eb0a566…
```

```
00e0  43 52 4e 1d 46 06 6c 29  f7 b4 8f 82 6e 3f a4 bd   CRN·F·l) ····n?··
00f0  8e 86 36 84 7b f3 ed 4a  57 35 f1 0e ea e2 07 c4   ··6·{··J W5······
0100  ae 8e 39 ff c7 5b ba cc  2b 87 6d a3 ff 2b 63 35   ··9··[·· +·m··+c5
0110  43 3d 0b 0a 0a 61 f3 6b  38 d0 d9 a7 66 5a 0a c0   C=···a·k 8···fZ··
0120  f2 64 d0 80 b2 8f 2b 5f  0a 3d c3 50 32 4d 37 09   ·d····+_ ·=·P2M7·
0130  c7 b9 91 31 c4 44 44 1a  d0 d7 93 45 d7 0d e5 1d   ···1·DD· ···E····
0140  ff f4 0a 2d e7 53 d6 43  4d 0e 5a 1f e6 b2 09 71   ···-·S·C M·Z····q
0150  48 ae 80 41 8c 67 12 76  5f ff d3 e1 17 aa 54 ac   H··A·g·v _·····T·
0160  79 05 ad 92 fc 3e 6d a4  fc 4d 78 cf e3 b5 2b 3e   y····>m· ·Mx···+>
0170  1c 60 ef a7 35 1c a3 3b  82 93 68 ee 26 af 8e 30   ·`··5··; ··h·&··0
0180  e0 ec 05 cb 39 eb f1 58  05 25 ce 31 76 cd 11 f9   ····9··X ·%·1v···
0190  a7 59 7a b3 79 43 12 3b  92 34 12 33 64 bf 21 1b   ·Yz·yC·; ·4·3d·!·
01a0  b4 6f 03 37 48 bc 6d ea  cf 58 a2 a2 fd 60 e1 b9   ·o·7H·m· ·X···`··
01b0  84 3c d4 9f 8d ff 66 bb  7c e4 31 63 db 39 af f0   ·<····f· |·1c·9··
01c0  3a 44 98 11 39 70 f7 32  2c e2 e0 a3 e6 e8 67 7e   :D··9p·2 ,·····g~
01d0  ab 59 eb fe f6 13 53 05  c0 cd 03 5f 7f 26 da 3c   ·Y····S· ···_·&·<
01e0  75 31 5b 75 e4 04 be 1c  99 82 92 90 3a a5 39 c8   u1[u···· ····:·9·
01f0  db d1 51 22 bf 53 a0 6b  18 75 f8 a8 5e 11 05 1a   ··Q"·S·k ·u··^···
0200  dd 38 89 d7 6f 80 d4 cf  5b 7d cb c0 7b e5 35 e6   ·8··o··· [}··{·5·
0210  5e 2e 58 6d ec c6 5c d9  bc 52 fb 5c 32 46 85 e2   ^.Xm··\· ·R·\2F··
0220  6f fa 14 49 5e 26 21 51  eb b4 0c 6b 49 89 10 a2   o··I^&!Q ···kI···
0230  8b 47 36 80 44 3e b5 40  ca 76 d5 cb ff 25 63 3f   ·G6·D>·@ ·v···%c?
0240  99 25 be f4 6a 92 dc 40  61 a7 1d e6 a6 c8 36 ab   ·%··j··@ a·····6·
0250  a2 21 ed 54 4c d0 a5 56  90 20 46 8f 5b da 95 25   ·!·TL··V · F·[··%
0260  b8 5d f7 5d 17 03 03 00  39 7a e5 c8 85 5e a8 82   ·]·]···· 9z···^··
0270  63 1e 04 0a a3 cc 8f 03  a4 e2 e7 2b af 7e b0 a5   c······· ···+·~··
0280  66 ba fe 2b f8 67 00 cf  21 c8 cc eb 3f 94 60 35   f··+·g·· !···?·`5
0290  65 84 e7 d2 57 8a ad 31  93 c0 5b 3b 51 bc 5f 81   e···W··1 ··[;Q·_·
02a0  b2 ff                                              ··
```

Payload is encrypted application data (tls.app_data), 57 bytes

Packets: 266 · Displayed: 266 (100.0%) · Dropped: 0 (0.0%)

Profile: Default



File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Wireshark · Packet 255 · cs21mtech16001.pcapng
Close
Help

```
No.   Time          Source
  226 11.625942779  172.21
  227 11.626275318  192.16
  228 11.631490585  172.21
  229 11.631585688  172.21
  230 11.633611724  192.16
  231 11.639754403  192.16
  232 11.640033774  192.16
  233 11.640675565  192.16
  234 11.642765199  172.21
  235 11.642842023  192.16
  236 11.643266293  192.16
  237 11.677254644  142.25
  238 11.677309937  142.25
  239 11.677637095  142.25
  240 11.677674522  192.16
  241 11.677853494  192.16
  242 11.682387396  172.21
  243 11.682453692  172.21
  244 11.682572787  172.21
  245 11.682723896  192.16
  246 11.692590313  172.21
  247 11.708306256  192.16
  248 11.746694549  142.25
  249 11.769041246  172.21
  250 11.769102259  192.16
  251 11.769148602  172.21
  252 11.769159473  192.16
  253 11.769462831  172.21
  254 11.769491189  192.16
  255 11.774695479  172.21
  256 11.774739265  192.16
  257 11.776249864  192.16
  258 11.797875854  172.21
  259 11.798108165  172.21
  260 11.798257239  172.21
  261 11.798326290  192.16
  262 11.798502450  192.16
  263 11.828719740  172.21
  264 11.828806579  172.21
  265 11.828829648  192.16
  266 11.867474227  172.21
```

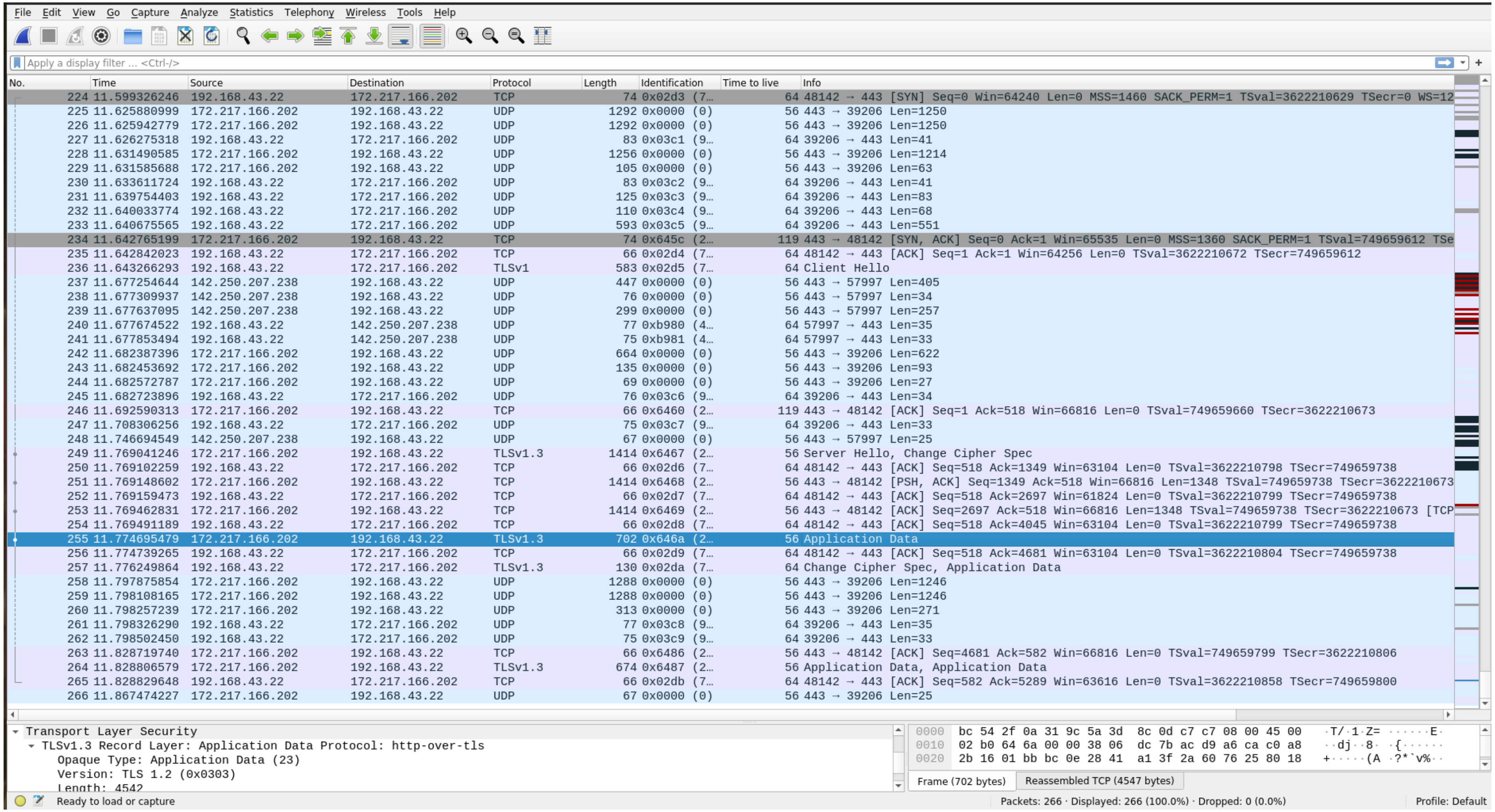
```
▸ Frame 255: 702 bytes on wire (5616 bits), 702 bytes captured (5616 bits) on interface wlp0s20f3, id 0
▸ Ethernet II, Src: 5a:3d:8c:0d:c7:c7 (5a:3d:8c:0d:c7:c7), Dst: IntelCor_0a:31:9c (bc:54:2f:0a:31:9c)
▸ Internet Protocol Version 4, Src: 172.217.166.202, Dst: 192.168.43.22
▸ Transmission Control Protocol, Src Port: 443, Dst Port: 48142, Seq: 4045, Ack: 518, Len: 636
▸ [4 Reassembled TCP Segments (4547 bytes): #249(1215), #251(1348), #253(1348), #255(636)]
▾ Transport Layer Security
  ▾ TLSv1.3 Record Layer: Application Data Protocol: http-over-tls
      Opaque Type: Application Data (23)
      Version: TLS 1.2 (0x0303)
      Length: 4542
      Encrypted Application Data: 32fd88d13f74f2fbc8dcb51b9bf441b603c18adbb9414dec…
```

```
0000  bc 54 2f 0a 31 9c 5a 3d  8c 0d c7 c7 08 00 45 00   ·T/·1·Z= ······E·
0010  02 b0 64 6a 00 00 38 06  dc 7b ac d9 a6 ca c0 a8   ··dj··8· ·{······
0020  2b 16 01 bb bc 0e 28 41  a1 3f 2a 60 76 25 80 18   +·····(A ·?*`v%··
0030  01 05 0a 3c 00 00 01 01  08 0a 2c ae e6 5a d7 e6   ···<···· ··,··Z··
0040  8c 71 fe 46 aa 61 d9 b3  58 9f b5 ca 36 12 13 6a   ·q·F·a·· X···6··j
0050  ec 1b 95 dd cc 2a 42 18  86 3e 24 55 49 ee e4 6c   ·····*B· ·>$UI··l
0060  3f f9 88 56 fd 11 fb e6  7f 18 f2 e9 b2 d3 72 13   ?··V···· ······r·
0070  f9 4e 66 72 ca e9 07 e3  92 1d ac b7 f1 d0 ff 41   ·Nfr···· ·······A
0080  7a f4 6c 8b 28 d6 14 6d  a7 58 8c c6 0b 9a 44 0a   z·l·(··m ·X····D·
0090  08 c1 41 fc 48 83 ff 44  7a 20 06 b1 e1 59 9a 07   ··A·H··D z ···Y··
00a0  a2 81 47 10 4a c6 cd 80  ab 10 39 de e3 9b e3 d7   ··G·J··· ··9·····
00b0  70 36 e7 82 5e b6 4b 30  f4 4e ee 44 7c 7e 89 ce   p6··^·K0 ·N·D|~··
00c0  6e 4b 49 32 d0 23 df dd  17 9c 8f 22 9d 9f 7d 8d   nKI2·#·· ···"··}·
00d0  bc 34 f8 b2 8a 50 5c 37  ec 76 32 58 ac 42 00 ec   ·4···P\7 ·v2X·B··
00e0  c3 de 47 4d f9 c9 27 0a  b9 57 a9 9a e9 96 65 ef   ··GM··'· ·W····e·
00f0  9a 98 24 f1 ea b9 c5 84  37 d6 8f 00 b1 c4 1b 59   ··$····· 7······Y
0100  7d bf 6a 75 ed e4 4d 3c  37 e9 7d 9b b9 5c 4e 97   }·ju··M< 7·}··\N·
0110  9a f9 b7 bf 56 8a f0 af  63 eb 90 11 fc 3d 7d 53   ····V··· c····=}S
0120  00 6e 05 e4 d4 4b 4c f0  9c 8d c7 10 23 0c 88 2b   ·n···KL· ····#··+
0130  38 75 1f 5a 38 0d 4d e4  15 d4 a3 0f b5 6c ce a2   8u·Z8·M· ·····l··
0140  81 bd dd 68 86 5a 95 d0  7c 3c 61 7a eb cb 98 17   ···h·Z·· |<az····
0150  a5 c6 ce ad 9b eb 9f 67  3d 25 69 e8 33 f6 2c 7b   ·······g =%i·3·,{
```

Frame (702 bytes) Reassembled TCP (4547 bytes)

```
▾ Transport Layer Security
  ▾ TLSv1.3 Record Layer: Appli
      Opaque Type: Application
      Version: TLS 1.2 (0x0303)
      Length: 4542
      Encrypted Application Data: 32fd88d13f74f2fbc8dcb51b9bf441b603c18adbb9414dec…
```

Ready to load or capture   Packets: 266 · Displayed: 266 (100.0%) · Dropped: 0 (0.0%)   Profile: Default


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>

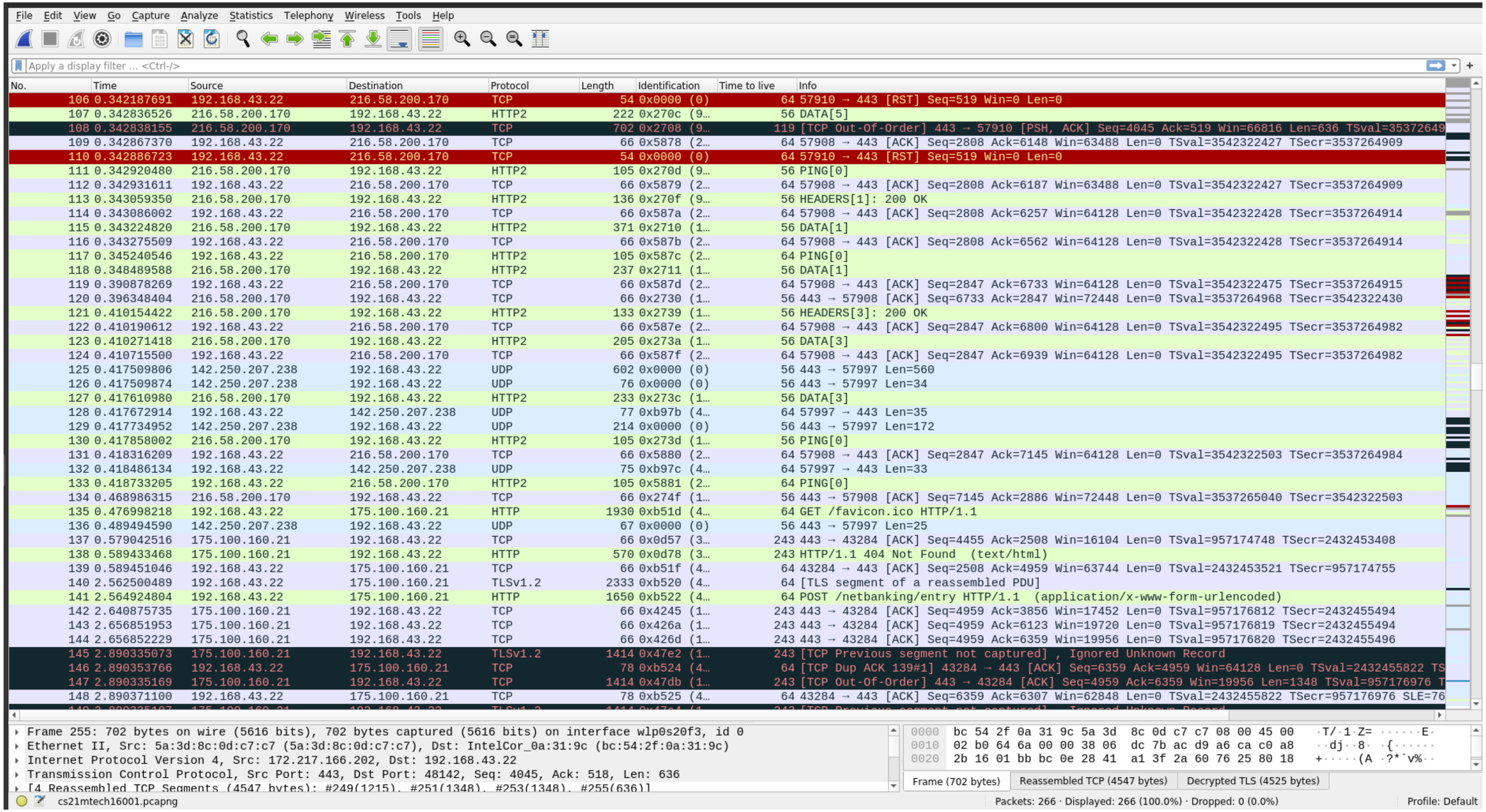
```
No.   Time         Source           Destination      Protocol  Length Identification Time to live Info
  224 11.599326246 192.168.43.22    172.217.166.202  TCP         74 0x02d3 (7…    64 48142 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=3622210629 TSecr=0 WS=12
  225 11.625880999 172.217.166.202  192.168.43.22    UDP       1292 0x0000 (0)    56 443 → 39206 Len=1250
  226 11.625942779 172.217.166.202  192.168.43.22    UDP       1292 0x0000 (0)    56 443 → 39206 Len=1250
  227 11.626275318 192.168.43.22    172.217.166.202  UDP         83 0x03c1 (9…    64 39206 → 443 Len=41
  228 11.631490585 172.217.166.202  192.168.43.22    UDP       1256 0x0000 (0)    56 443 → 39206 Len=1214
  229 11.631585688 172.217.166.202  192.168.43.22    UDP        105 0x0000 (0)    56 443 → 39206 Len=63
  230 11.633611724 192.168.43.22    172.217.166.202  UDP         83 0x03c2 (9…    64 39206 → 443 Len=41
  231 11.639754403 192.168.43.22    172.217.166.202  UDP        125 0x03c3 (9…    64 39206 → 443 Len=83
  232 11.640033774 192.168.43.22    172.217.166.202  UDP        110 0x03c4 (9…    64 39206 → 443 Len=68
  233 11.640675565 192.168.43.22    172.217.166.202  UDP        593 0x03c5 (9…    64 39206 → 443 Len=551
  234 11.642765199 172.217.166.202  192.168.43.22    TCP         74 0x645c (2…   119 443 → 48142 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1360 SACK_PERM=1 TSval=749659612 TSe
  235 11.642842023 192.168.43.22    172.217.166.202  TCP         66 0x02d4 (7…    64 48142 → 443 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=3622210672 TSecr=749659612
  236 11.643266293 192.168.43.22    172.217.166.202  TLSv1      583 0x02d5 (7…    64 Client Hello
  237 11.677254644 142.250.207.238  192.168.43.22    UDP        447 0x0000 (0)    56 443 → 57997 Len=405
  238 11.677309937 142.250.207.238  192.168.43.22    UDP         76 0x0000 (0)    56 443 → 57997 Len=34
  239 11.677637095 142.250.207.238  192.168.43.22    UDP        299 0x0000 (0)    56 443 → 57997 Len=257
  240 11.677674522 192.168.43.22    142.250.207.238  UDP         77 0xb980 (4…    64 57997 → 443 Len=35
  241 11.677853494 192.168.43.22    142.250.207.238  UDP         75 0xb981 (4…    64 57997 → 443 Len=33
  242 11.682387396 172.217.166.202  192.168.43.22    UDP        664 0x0000 (0)    56 443 → 39206 Len=622
  243 11.682453692 172.217.166.202  192.168.43.22    UDP        135 0x0000 (0)    56 443 → 39206 Len=93
  244 11.682572787 172.217.166.202  192.168.43.22    UDP         69 0x0000 (0)    56 443 → 39206 Len=27
  245 11.682723896 192.168.43.22    172.217.166.202  UDP         76 0x03c6 (9…    64 39206 → 443 Len=34
  246 11.692590313 172.217.166.202  192.168.43.22    TCP         66 0x6460 (2…   119 443 → 48142 [ACK] Seq=1 Ack=518 Win=66816 Len=0 TSval=749659660 TSecr=3622210673
  247 11.708306256 192.168.43.22    172.217.166.202  UDP         75 0x03c7 (9…    64 39206 → 443 Len=33
  248 11.746694549 142.250.207.238  192.168.43.22    UDP         67 0x0000 (0)    56 443 → 57997 Len=25
  249 11.769041246 172.217.166.202  192.168.43.22    TLSv1.3   1414 0x6467 (2…    56 Server Hello, Change Cipher Spec
  250 11.769102259 192.168.43.22    172.217.166.202  TCP         66 0x02d6 (7…    64 48142 → 443 [ACK] Seq=518 Ack=1349 Win=63104 Len=0 TSval=3622210798 TSecr=749659738
  251 11.769148602 172.217.166.202  192.168.43.22    TCP       1414 0x6468 (2…    56 443 → 48142 [PSH, ACK] Seq=1349 Ack=518 Win=66816 Len=1348 TSval=749659738 TSecr=3622210673
  252 11.769159473 192.168.43.22    172.217.166.202  TCP         66 0x02d7 (7…    64 48142 → 443 [ACK] Seq=518 Ack=2697 Win=61824 Len=0 TSval=3622210799 TSecr=749659738
  253 11.769462831 172.217.166.202  192.168.43.22    TCP       1414 0x6469 (2…    56 443 → 48142 [ACK] Seq=2697 Ack=518 Win=66816 Len=1348 TSval=749659738 TSecr=3622210673 [TCP
  254 11.769491189 192.168.43.22    172.217.166.202  TCP         66 0x02d8 (7…    64 48142 → 443 [ACK] Seq=518 Ack=4045 Win=63104 Len=0 TSval=3622210799 TSecr=749659738
  255 11.774695479 172.217.166.202  192.168.43.22    TLSv1.3    702 0x646a (2…    56 Application Data
  256 11.774739265 192.168.43.22    172.217.166.202  TCP         66 0x02d9 (7…    64 48142 → 443 [ACK] Seq=518 Ack=4681 Win=63104 Len=0 TSval=3622210804 TSecr=749659738
  257 11.776249864 192.168.43.22    172.217.166.202  TLSv1.3    130 0x02da (7…    64 Change Cipher Spec, Application Data
  258 11.797875854 172.217.166.202  192.168.43.22    UDP       1288 0x0000 (0)    56 443 → 39206 Len=1246
  259 11.798108165 172.217.166.202  192.168.43.22    UDP       1288 0x0000 (0)    56 443 → 39206 Len=1246
  260 11.798257239 172.217.166.202  192.168.43.22    UDP        313 0x0000 (0)    56 443 → 39206 Len=271
  261 11.798326290 192.168.43.22    172.217.166.202  UDP         77 0x03c8 (9…    64 39206 → 443 Len=35
  262 11.798502450 192.168.43.22    172.217.166.202  UDP         75 0x03c9 (9…    64 39206 → 443 Len=33
  263 11.828719740 172.217.166.202  192.168.43.22    TCP         66 0x6486 (2…    56 443 → 48142 [ACK] Seq=4681 Ack=582 Win=66816 Len=0 TSval=749659799 TSecr=3622210806
  264 11.828806579 172.217.166.202  192.168.43.22    TLSv1.3    674 0x6487 (2…    56 Application Data, Application Data
  265 11.828829648 192.168.43.22    172.217.166.202  TCP         66 0x02db (7…    64 48142 → 443 [ACK] Seq=582 Ack=5289 Win=63616 Len=0 TSval=3622210858 TSecr=749659800
  266 11.867474227 172.217.166.202  192.168.43.22    UDP         67 0x0000 (0)    56 443 → 39206 Len=25
```

```
▾ Transport Layer Security
  ▾ TLSv1.3 Record Layer: Application Data Protocol: http-over-tls
      Opaque Type: Application Data (23)
      Version: TLS 1.2 (0x0303)
      Length: 4542
```

```
0000  bc 54 2f 0a 31 9c 5a 3d  8c 0d c7 c7 08 00 45 00   ·T/·1·Z= ······E·
0010  02 b0 64 6a 00 00 38 06  dc 7b ac d9 a6 ca c0 a8   ··dj··8· ·{······
0020  2b 16 01 bb bc 0e 28 41  a1 3f 2a 60 76 25 80 18   +·····(A ·?*`v%··
```

Frame (702 bytes)
Reassembled TCP (4547 bytes)
Ready to load or capture
Packets: 266 · Displayed: 266 (100.0%) · Dropped: 0 (0.0%)
Profile: Default

Now after providing the SSLKeyLog file into Wireshark, all the encrypted conversations have been decrypted and available in plain text as shown below:


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>

```
No.   Time        Source          Destination     Protocol  Length Identification Time to live Info
  106 0.342187691 192.168.43.22   216.58.200.170  TCP           54 0x0000 (0)     64 57910 → 443 [RST] Seq=519 Win=0 Len=0
  107 0.342836526 216.58.200.170  192.168.43.22   HTTP2        222 0x270c (9…     56 DATA[5]
  108 0.342838155 216.58.200.170  192.168.43.22   TCP          702 0x2708 (9…    119 [TCP Out-Of-Order] 443 → 57910 [PSH, ACK] Seq=4045 Ack=519 Win=66816 Len=636 TSval=35372649
  109 0.342867370 192.168.43.22   216.58.200.170  TCP           66 0x5878 (2…     64 57908 → 443 [ACK] Seq=2808 Ack=6148 Win=63488 Len=0 TSval=3542322427 TSecr=3537264909
  110 0.342886723 192.168.43.22   216.58.200.170  TCP           54 0x0000 (0)     64 57910 → 443 [RST] Seq=519 Win=0 Len=0
  111 0.342920480 216.58.200.170  192.168.43.22   HTTP2        105 0x270d (9…     56 PING[0]
  112 0.342931611 192.168.43.22   216.58.200.170  TCP           66 0x5879 (2…     64 57908 → 443 [ACK] Seq=2808 Ack=6187 Win=63488 Len=0 TSval=3542322427 TSecr=3537264909
  113 0.343059350 216.58.200.170  192.168.43.22   HTTP2        136 0x270f (9…     56 HEADERS[1]: 200 OK
  114 0.343086002 192.168.43.22   216.58.200.170  TCP           66 0x587a (2…     64 57908 → 443 [ACK] Seq=2808 Ack=6257 Win=64128 Len=0 TSval=3542322428 TSecr=3537264914
  115 0.343224820 216.58.200.170  192.168.43.22   HTTP2        371 0x2710 (1…     56 DATA[1]
  116 0.343275509 192.168.43.22   216.58.200.170  TCP           66 0x587b (2…     64 57908 → 443 [ACK] Seq=2808 Ack=6562 Win=64128 Len=0 TSval=3542322428 TSecr=3537264914
  117 0.345240546 192.168.43.22   216.58.200.170  HTTP2        105 0x587c (2…     64 PING[0]
  118 0.348489588 216.58.200.170  192.168.43.22   HTTP2        237 0x2711 (1…     56 DATA[1]
  119 0.390878269 192.168.43.22   216.58.200.170  TCP           66 0x587d (2…     64 57908 → 443 [ACK] Seq=2847 Ack=6733 Win=64128 Len=0 TSval=3542322475 TSecr=3537264915
  120 0.396348404 216.58.200.170  192.168.43.22   TCP           66 0x2730 (1…     56 443 → 57908 [ACK] Seq=6733 Ack=2847 Win=72448 Len=0 TSval=3537264968 TSecr=3542322430
  121 0.410154422 216.58.200.170  192.168.43.22   HTTP2        133 0x2739 (1…     56 HEADERS[3]: 200 OK
  122 0.410190612 192.168.43.22   216.58.200.170  TCP           66 0x587e (2…     64 57908 → 443 [ACK] Seq=2847 Ack=6800 Win=64128 Len=0 TSval=3542322495 TSecr=3537264982
  123 0.410271418 216.58.200.170  192.168.43.22   HTTP2        205 0x273a (1…     56 DATA[3]
  124 0.410715500 192.168.43.22   216.58.200.170  TCP           66 0x587f (2…     64 57908 → 443 [ACK] Seq=2847 Ack=6939 Win=64128 Len=0 TSval=3542322495 TSecr=3537264982
  125 0.417509806 142.250.207.238 192.168.43.22   UDP          602 0x0000 (0)     56 443 → 57997 Len=560
  126 0.417509874 142.250.207.238 192.168.43.22   UDP           76 0x0000 (0)     56 443 → 57997 Len=34
  127 0.417610980 216.58.200.170  192.168.43.22   HTTP2        233 0x273c (1…     56 DATA[3]
  128 0.417672914 192.168.43.22   142.250.207.238 UDP           77 0xb97b (4…     64 57997 → 443 Len=35
  129 0.417734952 142.250.207.238 192.168.43.22   UDP          214 0x0000 (0)     56 443 → 57997 Len=172
  130 0.417858002 216.58.200.170  192.168.43.22   HTTP2        105 0x273d (1…     56 PING[0]
  131 0.418316209 192.168.43.22   216.58.200.170  TCP           66 0x5880 (2…     64 57908 → 443 [ACK] Seq=2847 Ack=7145 Win=64128 Len=0 TSval=3542322503 TSecr=3537264984
  132 0.418486134 192.168.43.22   142.250.207.238 UDP           75 0xb97c (4…     64 57997 → 443 Len=33
  133 0.418733205 192.168.43.22   216.58.200.170  HTTP2        105 0x5881 (2…     64 PING[0]
  134 0.468986315 216.58.200.170  192.168.43.22   TCP           66 0x274f (1…     56 443 → 57908 [ACK] Seq=7145 Ack=2886 Win=72448 Len=0 TSval=3537265040 TSecr=3542322503
  135 0.476998218 192.168.43.22   175.100.160.21  HTTP        1930 0xb51d (4…     64 GET /favicon.ico HTTP/1.1
  136 0.489494590 142.250.207.238 192.168.43.22   UDP           67 0x0000 (0)     56 443 → 57997 Len=25
  137 0.579042516 175.100.160.21  192.168.43.22   TCP           66 0x0d57 (3…    243 443 → 43284 [ACK] Seq=4455 Ack=2508 Win=16104 Len=0 TSval=957174748 TSecr=2432453408
  138 0.589433468 175.100.160.21  192.168.43.22   HTTP         570 0x0d78 (3…    243 HTTP/1.1 404 Not Found  (text/html)
  139 0.589451046 192.168.43.22   175.100.160.21  TCP           66 0xb51f (4…     64 43284 → 443 [ACK] Seq=2508 Ack=4959 Win=63744 Len=0 TSval=2432453521 TSecr=957174755
  140 2.562500489 192.168.43.22   175.100.160.21  TLSv1.2     2333 0xb520 (4…     64 [TLS segment of a reassembled PDU]
  141 2.564924804 192.168.43.22   175.100.160.21  HTTP        1650 0xb522 (4…     64 POST /netbanking/entry HTTP/1.1  (application/x-www-form-urlencoded)
  142 2.640875735 175.100.160.21  192.168.43.22   TCP           66 0x4245 (1…    243 443 → 43284 [ACK] Seq=4959 Ack=3856 Win=17452 Len=0 TSval=957176812 TSecr=2432455494
  143 2.656851953 175.100.160.21  192.168.43.22   TCP           66 0x426a (1…    243 443 → 43284 [ACK] Seq=4959 Ack=6123 Win=19720 Len=0 TSval=957176819 TSecr=2432455494
  144 2.656852229 175.100.160.21  192.168.43.22   TCP           66 0x426d (1…    243 443 → 43284 [ACK] Seq=4959 Ack=6359 Win=19956 Len=0 TSval=957176820 TSecr=2432455496
  145 2.890335073 175.100.160.21  192.168.43.22   TLSv1.2     1414 0x47e2 (1…    243 [TCP Previous segment not captured] , Ignored Unknown Record
  146 2.890353766 192.168.43.22   175.100.160.21  TCP           78 0xb524 (4…     64 [TCP Dup ACK 139#1] 43284 → 443 [ACK] Seq=6359 Ack=4959 Win=64128 Len=0 TSval=2432455822 TS
  147 2.890335169 175.100.160.21  192.168.43.22   TCP         1414 0x47db (1…    243 [TCP Out-Of-Order] 443 → 43284 [ACK] Seq=4959 Ack=6359 Win=19956 Len=1348 TSval=957176976 T
  148 2.890371100 192.168.43.22   175.100.160.21  TCP           78 0xb525 (4…     64 43284 → 443 [ACK] Seq=6359 Ack=6307 Win=62848 Len=0 TSval=2432455822 TSecr=957176976 SLE=76
```

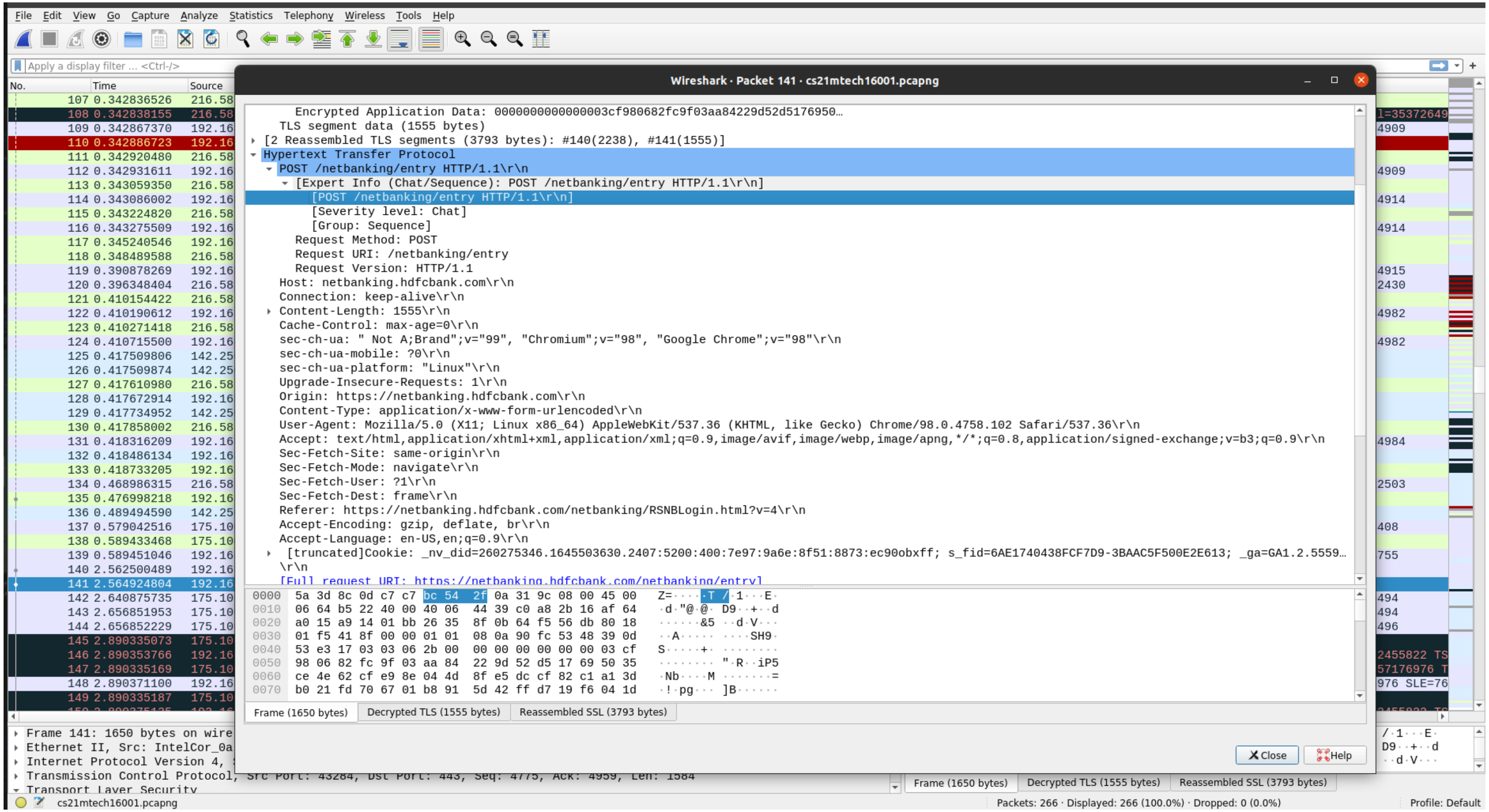
```
› Frame 255: 702 bytes on wire (5616 bits), 702 bytes captured (5616 bits) on interface wlp0s20f3, id 0
› Ethernet II, Src: 5a:3d:8c:0d:c7:c7 (5a:3d:8c:0d:c7:c7), Dst: IntelCor_0a:31:9c (bc:54:2f:0a:31:9c)
› Internet Protocol Version 4, Src: 172.217.166.202, Dst: 192.168.43.22
› Transmission Control Protocol, Src Port: 443, Dst Port: 48142, Seq: 4045, Ack: 518, Len: 636
› [4 Reassembled TCP Segments (4547 bytes): #249(1215), #251(1348), #253(1348), #255(636)]
```

```
0000  bc 54 2f 0a 31 9c 5a 3d  8c 0d c7 c7 08 00 45 00   ·T/·1·Z= ······E·
0010  02 b0 64 6a 00 00 38 06  dc 7b ac d9 a6 ca c0 a8   ··dj··8· ·{······
0020  2b 16 01 bb bc 0e 28 41  a1 3f 2a 60 76 25 80 18   +·····(A ·?*`v%··
```

cs21mtech16001.pcapng

Packets: 266 · Displayed: 266 (100.0%) · Dropped: 0 (0.0%)

Profile: Default



File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Wireshark · Packet 141 · cs21mtech16001.pcapng
Frame (1650 bytes)
Decrypted TLS (1555 bytes)
Reassembled SSL (3793 bytes)
Close
Help

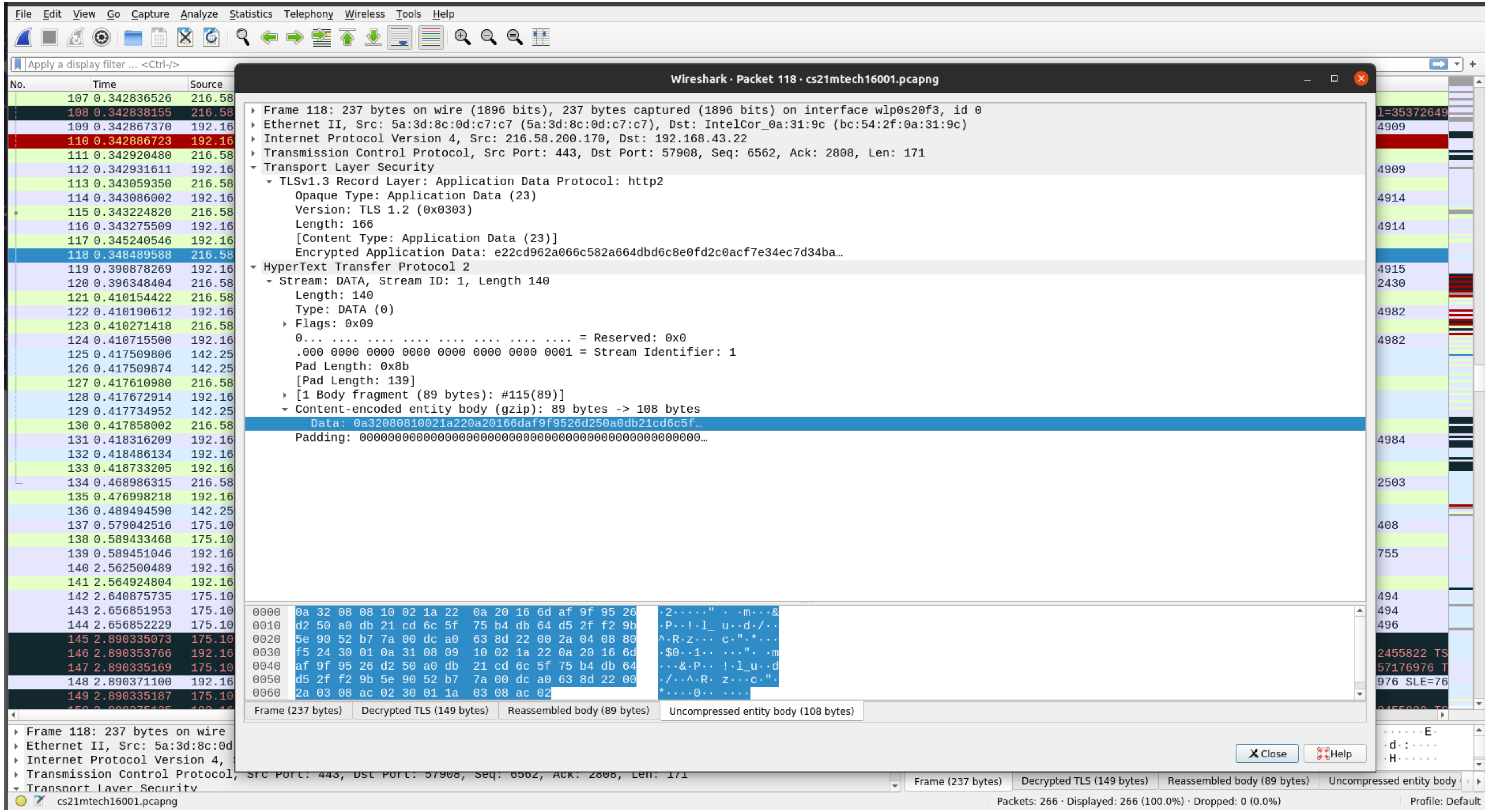
```
No.  Time        Source
107 0.342836526 216.58
108 0.342838155 216.58
109 0.342867370 192.16
110 0.342886723 192.16
111 0.342920480 216.58
112 0.342931611 192.16
113 0.343059350 216.58
114 0.343086002 192.16
115 0.343224820 216.58
116 0.343275509 192.16
117 0.345240546 192.16
118 0.348489588 216.58
119 0.390878269 192.16
120 0.396348404 216.58
121 0.410154422 216.58
122 0.410190612 192.16
123 0.410271418 216.58
124 0.410715500 192.16
125 0.417509806 142.25
126 0.417509874 142.25
127 0.417610980 216.58
128 0.417672914 192.16
129 0.417734952 142.25
130 0.417858002 216.58
131 0.418316209 192.16
132 0.418486134 192.16
133 0.418733205 192.16
134 0.468986315 216.58
135 0.476998218 192.16
136 0.489494590 142.25
137 0.579042516 175.10
138 0.589433468 175.10
139 0.589451046 192.16
140 2.562500489 192.16
141 2.564924804 192.16
142 2.640875735 175.10
143 2.656851953 175.10
144 2.656852229 175.10
145 2.890335073 175.10
146 2.890353766 192.16
147 2.890335169 175.10
148 2.890371100 192.16
149 2.890335187 175.10
```

```
    Encrypted Application Data: 0000000000000003cf980682fc9f03aa84229d52d5176950…
  TLS segment data (1555 bytes)
[2 Reassembled TLS segments (3793 bytes): #140(2238), #141(1555)]
Hypertext Transfer Protocol
  POST /netbanking/entry HTTP/1.1\r\n
    [Expert Info (Chat/Sequence): POST /netbanking/entry HTTP/1.1\r\n]
      [POST /netbanking/entry HTTP/1.1\r\n]
      [Severity level: Chat]
      [Group: Sequence]
    Request Method: POST
    Request URI: /netbanking/entry
    Request Version: HTTP/1.1
  Host: netbanking.hdfcbank.com\r\n
  Connection: keep-alive\r\n
  Content-Length: 1555\r\n
  Cache-Control: max-age=0\r\n
  sec-ch-ua: " Not A;Brand";v="99", "Chromium";v="98", "Google Chrome";v="98"\r\n
  sec-ch-ua-mobile: ?0\r\n
  sec-ch-ua-platform: "Linux"\r\n
  Upgrade-Insecure-Requests: 1\r\n
  Origin: https://netbanking.hdfcbank.com\r\n
  Content-Type: application/x-www-form-urlencoded\r\n
  User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/98.0.4758.102 Safari/537.36\r\n
  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=0.9\r\n
  Sec-Fetch-Site: same-origin\r\n
  Sec-Fetch-Mode: navigate\r\n
  Sec-Fetch-User: ?1\r\n
  Sec-Fetch-Dest: frame\r\n
  Referer: https://netbanking.hdfcbank.com/netbanking/RSNBLogin.html?v=4\r\n
  Accept-Encoding: gzip, deflate, br\r\n
  Accept-Language: en-US,en;q=0.9\r\n
   [truncated]Cookie: _nv_did=260275346.1645503630.2407:5200:400:7e97:9a6e:8f51:8873:ec90obxff; s_fid=6AE1740438FCF7D9-3BAAC5F500E2E613; _ga=GA1.2.5559…
  \r\n
  [Full request URI: https://netbanking.hdfcbank.com/netbanking/entry]
```

```
0000  5a 3d 8c 0d c7 c7 bc 54  2f 0a 31 9c 08 00 45 00   Z=·····T /·1···E·
0010  06 64 b5 22 40 00 40 06  44 39 c0 a8 2b 16 af 64   ·d·"@·@· D9··+··d
0020  a0 15 a9 14 01 bb 26 35  8f 0b 64 f5 56 db 80 18   ······&5 ··d·V···
0030  01 f5 41 8f 00 00 01 01  08 0a 90 fc 53 48 39 0d   ··A····· ····SH9·
0040  53 e3 17 03 03 06 2b 00  00 00 00 00 00 00 03 cf   S·····+· ········
0050  98 06 82 fc 9f 03 aa 84  22 9d 52 d5 17 69 50 35   ········ "·R··iP5
0060  ce 4e 62 cf e9 8e 04 4d  8f e5 dc cf 82 c1 a1 3d   ·Nb····M ·······=
0070  b0 21 fd 70 67 01 b8 91  5d 42 ff d7 19 f6 04 1d   ·!·pg··· ]B······
```

```
Frame 141: 1650 bytes on wire
Ethernet II, Src: IntelCor_0a
Internet Protocol Version 4,
Transmission Control Protocol, Src Port: 43284, Dst Port: 443, Seq: 4775, Ack: 4959, Len: 1584
Transport Layer Security
```

cs21mtech16001.pcapng   Packets: 266 · Displayed: 266 (100.0%) · Dropped: 0 (0.0%)   Profile: Default


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
No. Time Source
Wireshark · Packet 118 · cs21mtech16001.pcapng
Frame (237 bytes)
Decrypted TLS (149 bytes)
Reassembled body (89 bytes)
Uncompressed entity body (108 bytes)
Close
Help

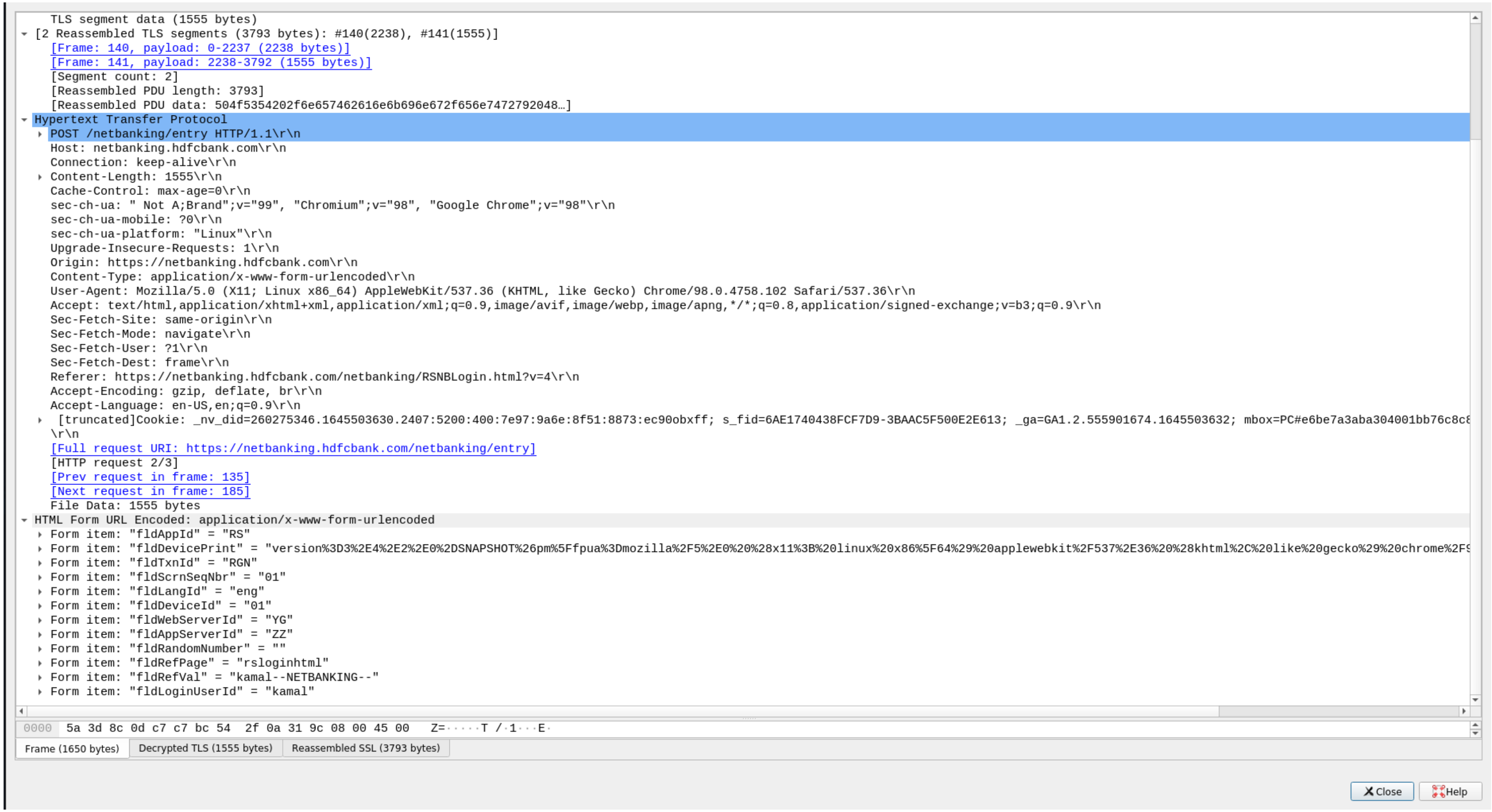
```
107 0.342836526  216.58
108 0.342838155  216.58
109 0.342867370  192.16
110 0.342886723  192.16
111 0.342920480  216.58
112 0.342931611  192.16
113 0.343059350  216.58
114 0.343086002  192.16
115 0.343224820  216.58
116 0.343275509  192.16
117 0.345240546  192.16
118 0.348489588  216.58
119 0.390878269  192.16
120 0.396348404  216.58
121 0.410154422  216.58
122 0.410190612  192.16
123 0.410271418  216.58
124 0.410715500  192.16
125 0.417509806  142.25
126 0.417509874  142.25
127 0.417610980  216.58
128 0.417672914  192.16
129 0.417734952  142.25
130 0.417858002  216.58
131 0.418316209  192.16
132 0.418486134  192.16
133 0.418733205  192.16
134 0.468986315  216.58
135 0.476998218  192.16
136 0.489494590  142.25
137 0.579042516  175.10
138 0.589433468  175.10
139 0.589451046  192.16
140 2.562500489  192.16
141 2.564924804  192.16
142 2.640875735  175.10
143 2.656851953  175.10
144 2.656852229  175.10
145 2.890335073  175.10
146 2.890353766  192.16
147 2.890335169  175.10
148 2.890371100  192.16
149 2.890335187  175.10
```

```
Frame 118: 237 bytes on wire (1896 bits), 237 bytes captured (1896 bits) on interface wlp0s20f3, id 0
Ethernet II, Src: 5a:3d:8c:0d:c7:c7 (5a:3d:8c:0d:c7:c7), Dst: IntelCor_0a:31:9c (bc:54:2f:0a:31:9c)
Internet Protocol Version 4, Src: 216.58.200.170, Dst: 192.168.43.22
Transmission Control Protocol, Src Port: 443, Dst Port: 57908, Seq: 6562, Ack: 2808, Len: 171
Transport Layer Security
  TLSv1.3 Record Layer: Application Data Protocol: http2
    Opaque Type: Application Data (23)
    Version: TLS 1.2 (0x0303)
    Length: 166
    [Content Type: Application Data (23)]
    Encrypted Application Data: e22cd962a066c582a664dbd6c8e0fd2c0acf7e34ec7d34ba…
HyperText Transfer Protocol 2
  Stream: DATA, Stream ID: 1, Length 140
    Length: 140
    Type: DATA (0)
    Flags: 0x09
    0... .... .... .... .... .... .... .... = Reserved: 0x0
    .000 0000 0000 0000 0000 0000 0000 0001 = Stream Identifier: 1
    Pad Length: 0x8b
    [Pad Length: 139]
    [1 Body fragment (89 bytes): #115(89)]
    Content-encoded entity body (gzip): 89 bytes -> 108 bytes
      Data: 0a32080810021a220a20166daf9f9526d250a0db21cd6c5f…
    Padding: 000000000000000000000000000000000000000000000000…
```

```
0000  0a 32 08 08 10 02 1a 22  0a 20 16 6d af 9f 95 26   ·2·····" · ·m···&
0010  d2 50 a0 db 21 cd 6c 5f  75 b4 db 64 d5 2f f2 9b   ·P··!·l_ u··d·/··
0020  5e 90 52 b7 7a 00 dc a0  63 8d 22 00 2a 04 08 80   ^·R·z··· c·"·*···
0030  f5 24 30 01 0a 31 08 09  10 02 1a 22 0a 20 16 6d   ·$0··1·· ···"· ·m
0040  af 9f 95 26 d2 50 a0 db  21 cd 6c 5f 75 b4 db 64   ···&·P·· !·l_u··d
0050  d5 2f f2 9b 5e 90 52 b7  7a 00 dc a0 63 8d 22 00   ·/··^·R· z···c·"·
0060  2a 03 08 ac 02 30 01 1a  03 08 ac 02               *····0·· ····
```

```
Frame 118: 237 bytes on wire
Ethernet II, Src: 5a:3d:8c:0d
Internet Protocol Version 4,
Transmission Control Protocol, Src Port: 443, Dst Port: 57908, Seq: 6562, Ack: 2808, Len: 171
Transport Layer Security
```

cs21mtech16001.pcapng   Packets: 266 · Displayed: 266 (100.0%) · Dropped: 0 (0.0%)   Profile: Default



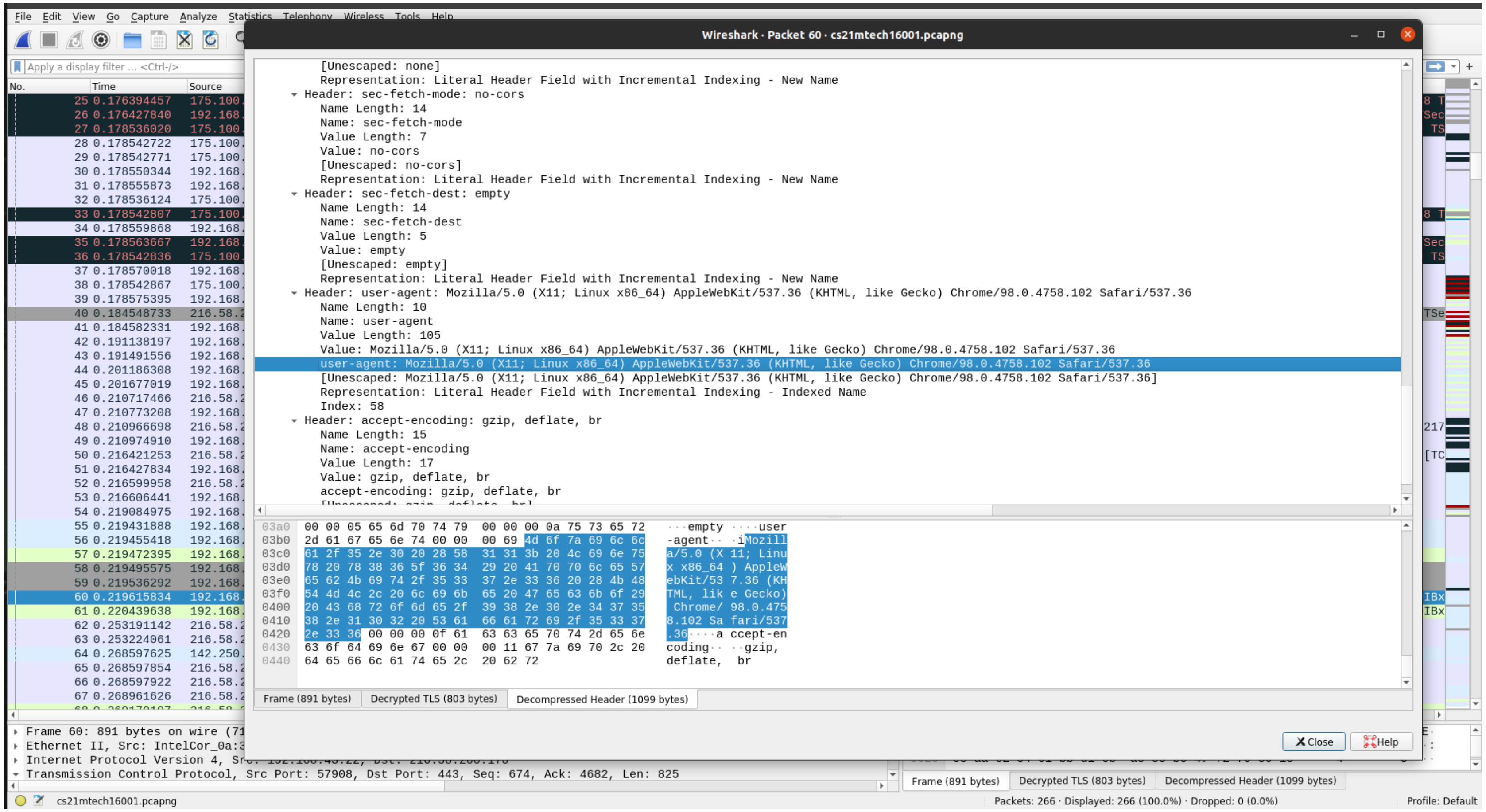
```
    TLS segment data (1555 bytes)
  [2 Reassembled TLS segments (3793 bytes): #140(2238), #141(1555)]
    [Frame: 140, payload: 0-2237 (2238 bytes)]
    [Frame: 141, payload: 2238-3792 (1555 bytes)]
    [Segment count: 2]
    [Reassembled PDU length: 3793]
    [Reassembled PDU data: 504f5354202f6e657462616e6b696e672f656e7472792048…]
  Hypertext Transfer Protocol
    POST /netbanking/entry HTTP/1.1\r\n
    Host: netbanking.hdfcbank.com\r\n
    Connection: keep-alive\r\n
    Content-Length: 1555\r\n
    Cache-Control: max-age=0\r\n
    sec-ch-ua: " Not A;Brand";v="99", "Chromium";v="98", "Google Chrome";v="98"\r\n
    sec-ch-ua-mobile: ?0\r\n
    sec-ch-ua-platform: "Linux"\r\n
    Upgrade-Insecure-Requests: 1\r\n
    Origin: https://netbanking.hdfcbank.com\r\n
    Content-Type: application/x-www-form-urlencoded\r\n
    User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/98.0.4758.102 Safari/537.36\r\n
    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=0.9\r\n
    Sec-Fetch-Site: same-origin\r\n
    Sec-Fetch-Mode: navigate\r\n
    Sec-Fetch-User: ?1\r\n
    Sec-Fetch-Dest: frame\r\n
    Referer: https://netbanking.hdfcbank.com/netbanking/RSNBLogin.html?v=4\r\n
    Accept-Encoding: gzip, deflate, br\r\n
    Accept-Language: en-US,en;q=0.9\r\n
     [truncated]Cookie: _nv_did=260275346.1645503630.2407:5200:400:7e97:9a6e:8f51:8873:ec90obxff; s_fid=6AE1740438FCF7D9-3BAAC5F500E2E613; _ga=GA1.2.555901674.1645503632; mbox=PC#e6be7a3aba304001bb76c8c8
    \r\n
    [Full request URI: https://netbanking.hdfcbank.com/netbanking/entry]
    [HTTP request 2/3]
    [Prev request in frame: 135]
    [Next request in frame: 185]
    File Data: 1555 bytes
  HTML Form URL Encoded: application/x-www-form-urlencoded
    Form item: "fldAppId" = "RS"
    Form item: "fldDevicePrint" = "version%3D3%2E4%2E2%2E0%2DSNAPSHOT%26pm%5Ffpua%3Dmozilla%2F5%2E0%20%28x11%3B%20linux%20x86%5F64%29%20applewebkit%2F537%2E36%20%28khtml%2C%20like%20gecko%29%20chrome%2F9
    Form item: "fldTxnId" = "RGN"
    Form item: "fldScrnSeqNbr" = "01"
    Form item: "fldLangId" = "eng"
    Form item: "fldDeviceId" = "01"
    Form item: "fldWebServerId" = "YG"
    Form item: "fldAppServerId" = "ZZ"
    Form item: "fldRandomNumber" = ""
    Form item: "fldRefPage" = "rsloginhtml"
    Form item: "fldRefVal" = "kamal--NETBANKING--"
    Form item: "fldLoginUserId" = "kamal"

0000  5a 3d 8c 0d c7 c7 bc 54  2f 0a 31 9c 08 00 45 00   Z=·····T /·1···E·
```

PART - B

1. What browser did you use, what's the version number?

I used Google Chrome to access the website and the version was: Google Chrome 98.0.4758.102 as shown in the screenshot below:


File
Edit
View
Go
Capture
Analyze
Statistics
Telephony
Wireless
Tools
Help
Apply a display filter ... <Ctrl-/>
Wireshark · Packet 60 · cs21mtech16001.pcapng
Close
Help

```
No.  Time          Source
  25 0.176394457   175.100.
  26 0.176427840   192.168.
  27 0.178536020   175.100.
  28 0.178542722   175.100.
  29 0.178542771   175.100.
  30 0.178550344   192.168.
  31 0.178555873   192.168.
  32 0.178536124   175.100.
  33 0.178542807   175.100.
  34 0.178559868   192.168.
  35 0.178563667   192.168.
  36 0.178542836   175.100.
  37 0.178570018   192.168.
  38 0.178542867   175.100.
  39 0.178575395   192.168.
  40 0.184548733   216.58.2
  41 0.184582331   192.168.
  42 0.191138197   192.168.
  43 0.191491556   192.168.
  44 0.201186308   192.168.
  45 0.201677019   192.168.
  46 0.210717466   216.58.2
  47 0.210773208   192.168.
  48 0.210966698   216.58.2
  49 0.210974910   192.168.
  50 0.216421253   216.58.2
  51 0.216427834   192.168.
  52 0.216599958   216.58.2
  53 0.216606441   192.168.
  54 0.219084975   192.168.
  55 0.219431888   192.168.
  56 0.219455418   192.168.
  57 0.219472395   192.168.
  58 0.219495575   192.168.
  59 0.219536292   192.168.
  60 0.219615834   192.168.
  61 0.220439638   192.168.
  62 0.253191142   216.58.2
  63 0.253224061   216.58.2
  64 0.268597625   142.250.
  65 0.268597854   216.58.2
  66 0.268597922   216.58.2
  67 0.268961626   216.58.2
```

```
        [Unescaped: none]
        Representation: Literal Header Field with Incremental Indexing - New Name
    ▾ Header: sec-fetch-mode: no-cors
        Name Length: 14
        Name: sec-fetch-mode
        Value Length: 7
        Value: no-cors
        [Unescaped: no-cors]
        Representation: Literal Header Field with Incremental Indexing - New Name
    ▾ Header: sec-fetch-dest: empty
        Name Length: 14
        Name: sec-fetch-dest
        Value Length: 5
        Value: empty
        [Unescaped: empty]
        Representation: Literal Header Field with Incremental Indexing - New Name
    ▾ Header: user-agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/98.0.4758.102 Safari/537.36
        Name Length: 10
        Name: user-agent
        Value Length: 105
        Value: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/98.0.4758.102 Safari/537.36
        user-agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/98.0.4758.102 Safari/537.36
        [Unescaped: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/98.0.4758.102 Safari/537.36]
        Representation: Literal Header Field with Incremental Indexing - Indexed Name
        Index: 58
    ▾ Header: accept-encoding: gzip, deflate, br
        Name Length: 15
        Name: accept-encoding
        Value Length: 17
        Value: gzip, deflate, br
        accept-encoding: gzip, deflate, br
```

```
03a0  00 00 05 65 6d 70 74 79  00 00 00 0a 75 73 65 72   ···empty ····user
03b0  2d 61 67 65 6e 74 00 00  00 69 4d 6f 7a 69 6c 6c   -agent·· ·iMozill
03c0  61 2f 35 2e 30 20 28 58  31 31 3b 20 4c 69 6e 75   a/5.0 (X 11; Linu
03d0  78 20 78 38 36 5f 36 34  29 20 41 70 70 6c 65 57   x x86_64 ) AppleW
03e0  65 62 4b 69 74 2f 35 33  37 2e 33 36 20 28 4b 48   ebKit/53 7.36 (KH
03f0  54 4d 4c 2c 20 6c 69 6b  65 20 47 65 63 6b 6f 29   TML, lik e Gecko)
0400  20 43 68 72 6f 6d 65 2f  39 38 2e 30 2e 34 37 35    Chrome/ 98.0.475
0410  38 2e 31 30 32 20 53 61  66 61 72 69 2f 35 33 37   8.102 Sa fari/537
0420  2e 33 36 00 00 00 0f 61  63 63 65 70 74 2d 65 6e   .36····a ccept-en
0430  63 6f 64 69 6e 67 00 00  00 11 67 7a 69 70 2c 20   coding·· ··gzip,
0440  64 65 66 6c 61 74 65 2c  20 62 72                  deflate,  br
```

Frame (891 bytes) | Decrypted TLS (803 bytes) | Decompressed Header (1099 bytes)

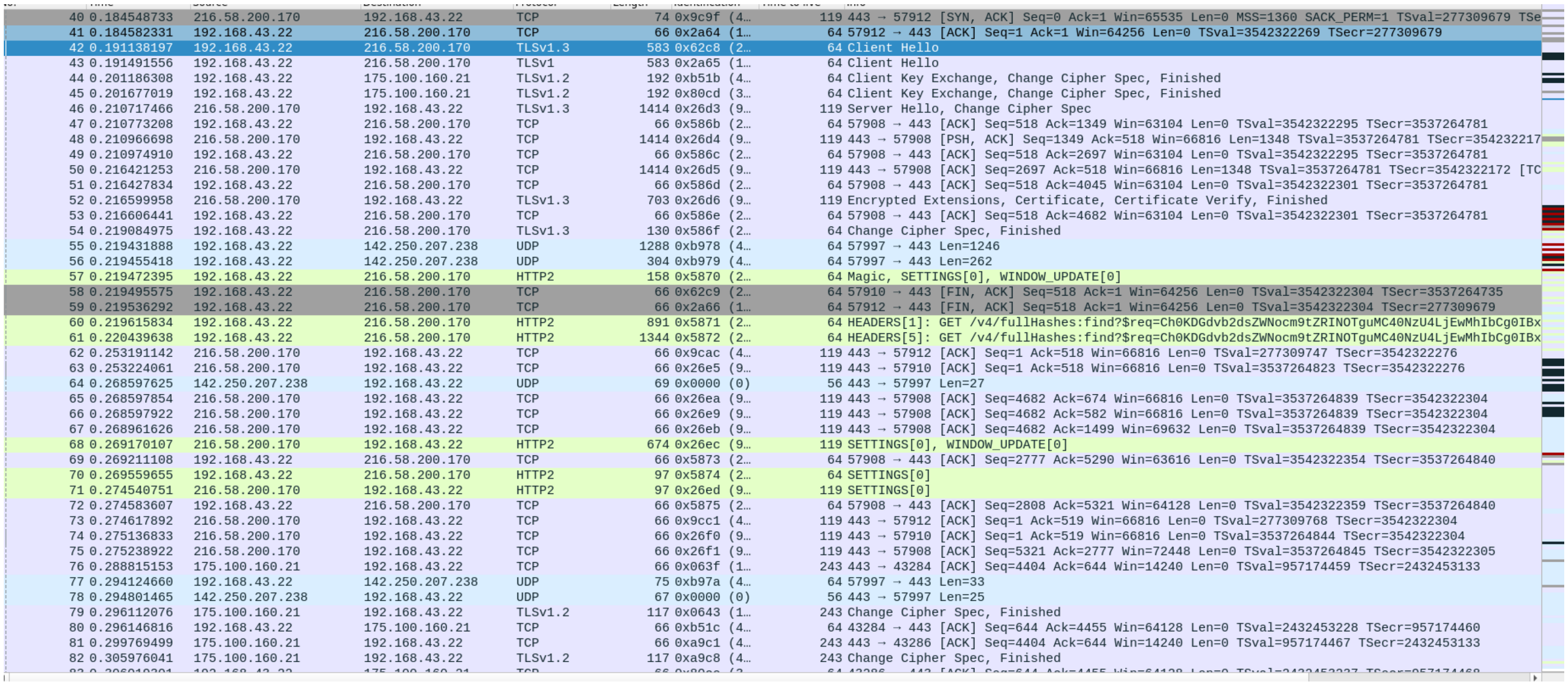
```
▸ Frame 60: 891 bytes on wire (71
▸ Ethernet II, Src: IntelCor_0a:3
▸ Internet Protocol Version 4, Src
▾ Transmission Control Protocol, Src Port: 57908, Dst Port: 443, Seq: 674, Ack: 4682, Len: 825
```

Frame (891 bytes) | Decrypted TLS (803 bytes) | Decompressed Header (1099 bytes)

cs21mtech16001.pcapng   Packets: 266 · Displayed: 266 (100.0%) · Dropped: 0 (0.0%)   Profile: Default

2. List out various protocols that you noticed in the column named “Protocol” in the Wireshark GUI from the time you keyed in the hostname of the bank in the browser till you start viewing application data. For each such protocol, mention its purpose in brief.

   The following were the protocols seen from Client Hello to Application Data:



```
40 0.184548733 216.58.200.170  192.168.43.22   TCP       74 0x9c9f (4…  119 443 → 57912 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1360 SACK_PERM=1 TSval=277309679 TSe
41 0.184582331 192.168.43.22   216.58.200.170  TCP       66 0x2a64 (1…   64 57912 → 443 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=3542322269 TSecr=277309679
42 0.191138197 192.168.43.22   216.58.200.170  TLSv1.3  583 0x62c8 (2…   64 Client Hello
43 0.191491556 192.168.43.22   216.58.200.170  TLSv1    583 0x2a65 (1…   64 Client Hello
44 0.201186308 192.168.43.22   175.100.160.21  TLSv1.2  192 0xb51b (4…   64 Client Key Exchange, Change Cipher Spec, Finished
45 0.201677019 192.168.43.22   175.100.160.21  TLSv1.2  192 0x80cd (3…   64 Client Key Exchange, Change Cipher Spec, Finished
46 0.210717466 216.58.200.170  192.168.43.22   TLSv1.3 1414 0x26d3 (9…  119 Server Hello, Change Cipher Spec
47 0.210773208 192.168.43.22   216.58.200.170  TCP       66 0x586b (2…   64 57908 → 443 [ACK] Seq=518 Ack=1349 Win=63104 Len=0 TSval=3542322295 TSecr=3537264781
48 0.210966698 216.58.200.170  192.168.43.22   TCP     1414 0x26d4 (9…  119 443 → 57908 [PSH, ACK] Seq=1349 Ack=518 Win=66816 Len=1348 TSval=3537264781 TSecr=354232217
49 0.210974910 192.168.43.22   216.58.200.170  TCP       66 0x586c (2…   64 57908 → 443 [ACK] Seq=518 Ack=2697 Win=63104 Len=0 TSval=3542322295 TSecr=3537264781
50 0.216421253 216.58.200.170  192.168.43.22   TCP     1414 0x26d5 (9…  119 443 → 57908 [ACK] Seq=2697 Ack=518 Win=66816 Len=1348 TSval=3537264781 TSecr=3542322172 [TC
51 0.216427834 192.168.43.22   216.58.200.170  TCP       66 0x586d (2…   64 57908 → 443 [ACK] Seq=518 Ack=4045 Win=63104 Len=0 TSval=3542322301 TSecr=3537264781
52 0.216599958 216.58.200.170  192.168.43.22   TLSv1.3  703 0x26d6 (9…  119 Encrypted Extensions, Certificate, Certificate Verify, Finished
53 0.216606441 192.168.43.22   216.58.200.170  TCP       66 0x586e (2…   64 57908 → 443 [ACK] Seq=518 Ack=4682 Win=63104 Len=0 TSval=3542322301 TSecr=3537264781
54 0.219084975 192.168.43.22   216.58.200.170  TLSv1.3  130 0x586f (2…   64 Change Cipher Spec, Finished
55 0.219431888 192.168.43.22   142.250.207.238 UDP     1288 0xb978 (4…   64 57997 → 443 Len=1246
56 0.219455418 192.168.43.22   142.250.207.238 UDP      304 0xb979 (4…   64 57997 → 443 Len=262
57 0.219472395 192.168.43.22   216.58.200.170  HTTP2    158 0x5870 (2…   64 Magic, SETTINGS[0], WINDOW_UPDATE[0]
58 0.219495575 192.168.43.22   216.58.200.170  TCP       66 0x62c9 (2…   64 57910 → 443 [FIN, ACK] Seq=518 Ack=1 Win=64256 Len=0 TSval=3542322304 TSecr=3537264735
59 0.219536292 192.168.43.22   216.58.200.170  TCP       66 0x2a66 (1…   64 57912 → 443 [FIN, ACK] Seq=518 Ack=1 Win=64256 Len=0 TSval=3542322304 TSecr=277309679
60 0.219615834 192.168.43.22   216.58.200.170  HTTP2    891 0x5871 (2…   64 HEADERS[1]: GET /v4/fullHashes:find?$req=Ch0KDGdvb2dsZWNocm9tZRINOTguMC40NzU4LjEwMhIbCg0IBx
61 0.220439638 192.168.43.22   216.58.200.170  HTTP2   1344 0x5872 (2…   64 HEADERS[5]: GET /v4/fullHashes:find?$req=Ch0KDGdvb2dsZWNocm9tZRINOTguMC40NzU4LjEwMhIbCg0IBx
62 0.253191142 216.58.200.170  192.168.43.22   TCP       66 0x9cac (4…  119 443 → 57912 [ACK] Seq=1 Ack=518 Win=66816 Len=0 TSval=277309747 TSecr=3542322276
63 0.253224061 216.58.200.170  192.168.43.22   TCP       66 0x26e5 (9…  119 443 → 57910 [ACK] Seq=1 Ack=518 Win=66816 Len=0 TSval=3537264823 TSecr=3542322276
64 0.268597625 142.250.207.238 192.168.43.22   UDP       69 0x0000 (0)   56 443 → 57997 Len=27
65 0.268597854 216.58.200.170  192.168.43.22   TCP       66 0x26ea (9…  119 443 → 57908 [ACK] Seq=4682 Ack=674 Win=66816 Len=0 TSval=3537264839 TSecr=3542322304
66 0.268597922 216.58.200.170  192.168.43.22   TCP       66 0x26e9 (9…  119 443 → 57908 [ACK] Seq=4682 Ack=582 Win=66816 Len=0 TSval=3537264839 TSecr=3542322304
67 0.268961626 216.58.200.170  192.168.43.22   TCP       66 0x26eb (9…  119 443 → 57908 [ACK] Seq=4682 Ack=1499 Win=69632 Len=0 TSval=3537264839 TSecr=3542322304
68 0.269170107 216.58.200.170  192.168.43.22   HTTP2    674 0x26ec (9…  119 SETTINGS[0], WINDOW_UPDATE[0]
69 0.269211108 192.168.43.22   216.58.200.170  TCP       66 0x5873 (2…   64 57908 → 443 [ACK] Seq=2777 Ack=5290 Win=63616 Len=0 TSval=3542322354 TSecr=3537264840
70 0.269559655 192.168.43.22   216.58.200.170  HTTP2     97 0x5874 (2…   64 SETTINGS[0]
71 0.274540751 216.58.200.170  192.168.43.22   HTTP2     97 0x26ed (9…  119 SETTINGS[0]
72 0.274583607 192.168.43.22   216.58.200.170  TCP       66 0x5875 (2…   64 57908 → 443 [ACK] Seq=2808 Ack=5321 Win=64128 Len=0 TSval=3542322359 TSecr=3537264840
73 0.274617892 216.58.200.170  192.168.43.22   TCP       66 0x9cc1 (4…  119 443 → 57912 [ACK] Seq=1 Ack=519 Win=66816 Len=0 TSval=277309768 TSecr=3542322304
74 0.275136833 216.58.200.170  192.168.43.22   TCP       66 0x26f0 (9…  119 443 → 57910 [ACK] Seq=1 Ack=519 Win=66816 Len=0 TSval=3537264844 TSecr=3542322304
75 0.275238922 216.58.200.170  192.168.43.22   TCP       66 0x26f1 (9…  119 443 → 57908 [ACK] Seq=5321 Ack=2777 Win=72448 Len=0 TSval=3537264845 TSecr=3542322305
76 0.288815153 175.100.160.21  192.168.43.22   TCP       66 0x063f (1…  243 443 → 43284 [ACK] Seq=4404 Ack=644 Win=14240 Len=0 TSval=957174459 TSecr=2432453133
77 0.294124660 192.168.43.22   142.250.207.238 UDP       75 0xb97a (4…   64 57997 → 443 Len=33
78 0.294801465 142.250.207.238 192.168.43.22   UDP       67 0x0000 (0)   56 443 → 57997 Len=25
79 0.296112076 175.100.160.21  192.168.43.22   TLSv1.2  117 0x0643 (1…  243 Change Cipher Spec, Finished
80 0.296146816 192.168.43.22   175.100.160.21  TCP       66 0xb51c (4…   64 43284 → 443 [ACK] Seq=644 Ack=4455 Win=64128 Len=0 TSval=2432453228 TSecr=957174460
81 0.299769499 175.100.160.21  192.168.43.22   TCP       66 0xa9c1 (4…  243 443 → 43286 [ACK] Seq=4404 Ack=644 Win=14240 Len=0 TSval=957174467 TSecr=2432453133
82 0.305976041 175.100.160.21  192.168.43.22   TLSv1.2  117 0xa9c8 (4…  243 Change Cipher Spec, Finished
```

1. **TLSv1.2**
   TLS1.2 is a transport layer security protocol that is built on top of TCP to ensure secure encrypted communication between the communicating parties to maintain the confidentiality of the exchanged message, the integrity of the message from outside/middle intruders, and the authenticity of the communicating parties.
2. **TLS v1.3**
   TLS 1.3 is an improved version of 1.2 that requires less handshake time, provides a more secure cryptographic encryption, reduced roundtrip time, streamlined key exchange, and overall more security.
3. **TCP**
   TCP is one of the principal internet transport protocols that ensure reliable and in-order delivery of packets from source to destination with proper congestion control and a dedicated connection setup.
4. **UDP**
   UDP is the best effort internet transfer protocol that doesn't ensure any reliability, in-order delivery, or any flow control mechanisms. It simply is the best effort protocol that is used for applications that require high-speed delivery of data with less constraint incomplete delivery like in VoIP, real-time video streaming, etc.
5. **HTTP2**
   HTTP2 is an application layer protocol, an improved version of HTTP1.1, that is used to fetch, change or delete resources, information, or any data from a server. An improved version of HTTP1.1 ensures increased flexibility at the server and mitigates HOL Blocking (decreased delay in multi-object HTTP requests).

3. Each of the TLS records begins with the same three fields (with possibly different values). One of these fields is “content-type” and has a length of one byte. List all three fields and their lengths for the first 10 records in the trace.

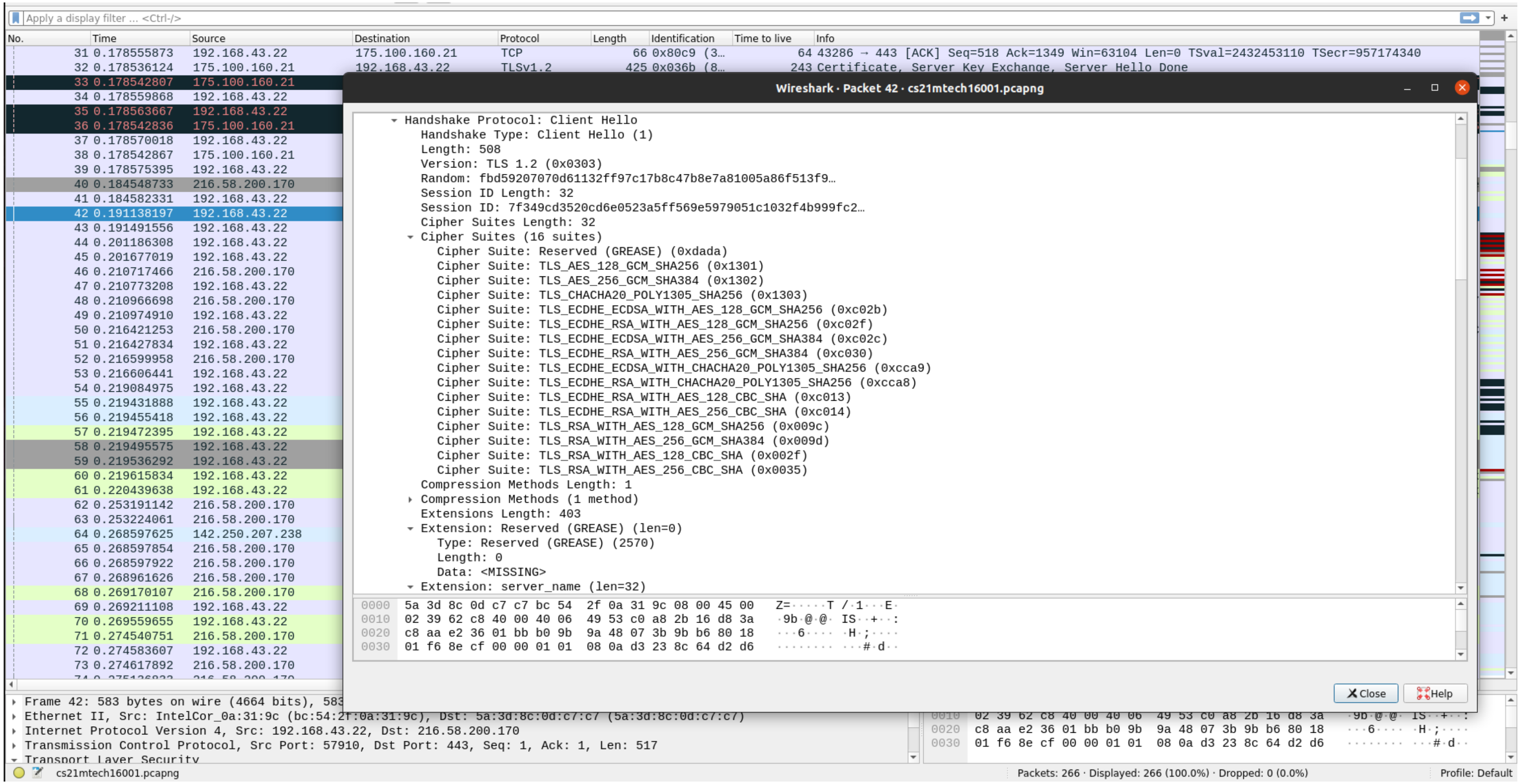
Records	**Fields**		**Lengths**
1	Content-Type	Handshake (22)	1 Byte
	Version	TLS 1.0	2 Bytes
	Length	512	2 Bytes

2	Content-Type	Handshake (22)	1 Byte
	Version	TLS 1.2	2 Bytes
	Length	508	2 Bytes
3	Opaque-Type	Change Cipher Spec (20)	1 Byte
	Version	TLS 1.2	2 Bytes
	Length	1	2 Bytes
4	Content-Type	Handshake (22)	1 Byte
	Version	TLS 1.2	2 Bytes
	Length	70	2 Bytes
5	Content-Type	Handshake (22)	1 Byte
	Version	TLS 1.2	2 Bytes
	Length	122	2 Bytes
6	Opaque-Type	Application Data (23)	1 Byte
	Version	TLS 1.2	2 Bytes
	Length	4543	2 Bytes
7	Opaque-Type	Application Data (23)	1 Byte
	Version	TLS 1.2	2 Bytes
	Length	53	2 Bytes

8	Opaque-Type	Application Data (23)	1 Byte
	Version	TLS 1.2	2 Bytes
	Length	87	2 Bytes
9	Opaque-Type	Application Data (23)	1 Byte
	Version	TLS 1.2	2 Bytes
	Length	820	2 Bytes
10	Opaque-Type	Application Data (23)	1 Byte
	Version	TLS 1.2	2 Bytes
	Length	634	2 Bytes

4. Cipher Suites in ClientHello Record: Look at the first two and the last cipher suites offered by the client and compare them. What cipher suite does the server select?

   Looking at the cipher suites advertised by the client, the following are the first two and the last two cipher suites offered:



```
Apply a display filter ... <Ctrl-/>
No.  Time          Source           Destination      Protocol  Length  Identification  Time to live  Info
31 0.178555873   192.168.43.22    175.100.160.21   TCP           66 0x80c9 (3…          64 43286 → 443 [ACK] Seq=518 Ack=1349 Win=63104 Len=0 TSval=2432453110 TSecr=957174340
32 0.178536124   175.100.160.21   192.168.43.22    TLSv1.2      425 0x036b (8…         243 Certificate, Server Key Exchange, Server Hello Done
33 0.178542807   175.100.160.21
34 0.178559868   192.168.43.22
35 0.178563667   192.168.43.22
36 0.178542836   175.100.160.21
37 0.178570018   192.168.43.22
38 0.178542867   175.100.160.21
39 0.178575395   192.168.43.22
40 0.184548733   216.58.200.170
41 0.184582331   192.168.43.22
42 0.191138197   192.168.43.22
43 0.191491556   192.168.43.22
44 0.201186308   192.168.43.22
45 0.201677019   192.168.43.22
46 0.210717466   216.58.200.170
47 0.210773208   192.168.43.22
48 0.210966698   216.58.200.170
49 0.210974910   192.168.43.22
50 0.216421253   216.58.200.170
51 0.216427834   192.168.43.22
52 0.216599958   216.58.200.170
53 0.216606441   192.168.43.22
54 0.219084975   192.168.43.22
55 0.219431888   192.168.43.22
56 0.219455418   192.168.43.22
57 0.219472395   192.168.43.22
58 0.219495575   192.168.43.22
59 0.219536292   192.168.43.22
60 0.219615834   192.168.43.22
61 0.220439638   192.168.43.22
62 0.253191142   216.58.200.170
63 0.253224061   216.58.200.170
64 0.268597625   142.250.207.238
65 0.268597854   216.58.200.170
66 0.268597922   216.58.200.170
67 0.268961626   216.58.200.170
68 0.269170107   216.58.200.170
69 0.269211108   192.168.43.22
70 0.269559655   192.168.43.22
71 0.274540751   216.58.200.170
72 0.274583607   192.168.43.22
73 0.274617892   216.58.200.170
```

Wireshark · Packet 42 · cs21mtech16001.pcapng

```
▾ Handshake Protocol: Client Hello
    Handshake Type: Client Hello (1)
    Length: 508
    Version: TLS 1.2 (0x0303)
    Random: fbd59207070d61132ff97c17b8c47b8e7a81005a86f513f9…
    Session ID Length: 32
    Session ID: 7f349cd3520cd6e0523a5ff569e5979051c1032f4b999fc2…
    Cipher Suites Length: 32
  ▾ Cipher Suites (16 suites)
      Cipher Suite: Reserved (GREASE) (0xdada)
      Cipher Suite: TLS_AES_128_GCM_SHA256 (0x1301)
      Cipher Suite: TLS_AES_256_GCM_SHA384 (0x1302)
      Cipher Suite: TLS_CHACHA20_POLY1305_SHA256 (0x1303)
      Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 (0xc02b)
      Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)
      Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 (0xc02c)
      Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030)
      Cipher Suite: TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256 (0xcca9)
      Cipher Suite: TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256 (0xcca8)
      Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xc013)
      Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014)
      Cipher Suite: TLS_RSA_WITH_AES_128_GCM_SHA256 (0x009c)
      Cipher Suite: TLS_RSA_WITH_AES_256_GCM_SHA384 (0x009d)
      Cipher Suite: TLS_RSA_WITH_AES_128_CBC_SHA (0x002f)
      Cipher Suite: TLS_RSA_WITH_AES_256_CBC_SHA (0x0035)
    Compression Methods Length: 1
  ▸ Compression Methods (1 method)
    Extensions Length: 403
  ▾ Extension: Reserved (GREASE) (len=0)
      Type: Reserved (GREASE) (2570)
      Length: 0
      Data: <MISSING>
  ▾ Extension: server_name (len=32)

0000  5a 3d 8c 0d c7 c7 bc 54  2f 0a 31 9c 08 00 45 00   Z=·····T /·1···E·
0010  02 39 62 c8 40 00 40 06  49 53 c0 a8 2b 16 d8 3a   ·9b·@·@· IS··+··:
0020  c8 aa e2 36 01 bb b0 9b  9a 48 07 3b 9b b6 80 18   ···6···· ·H·;····
0030  01 f6 8e cf 00 00 01 01  08 0a d3 23 8c 64 d2 d6   ········ ···#·d··
```

```
▸ Frame 42: 583 bytes on wire (4664 bits), 583
▸ Ethernet II, Src: IntelCor_0a:31:9c (bc:54:2f:0a:31:9c), Dst: 5a:3d:8c:0d:c7:c7 (5a:3d:8c:0d:c7:c7)
▸ Internet Protocol Version 4, Src: 192.168.43.22, Dst: 216.58.200.170
▸ Transmission Control Protocol, Src Port: 57910, Dst Port: 443, Seq: 1, Ack: 1, Len: 517
▾ Transport Layer Security

0010  02 39 62 c8 40 00 40 06  49 53 c0 a8 2b 16 d8 3a   ·9b·@·@· IS··+··:
0020  c8 aa e2 36 01 bb b0 9b  9a 48 07 3b 9b b6 80 18   ···6···· ·H·;····
0030  01 f6 8e cf 00 00 01 01  08 0a d3 23 8c 64 d2 d6   ········ ···#·d··

cs21mtech16001.pcapng        Packets: 266 · Displayed: 266 (100.0%) · Dropped: 0 (0.0%)        Profile: Default
```

Reserved(GREASE) = Any proper implementation of TLS protocol, should also process these GREASE cipher suites which are basically a random collection of a number of unknown cipher suites (not valid). In case of a new or unknown/unidentified cipher suites are advertised by either the client to an old server, then such cipher suites are processed as GREASE suites or reserved suites and ignored so that the compatibility of that particular new suite remains doesn't raise any compatibility issues at the server end and handshake fails.

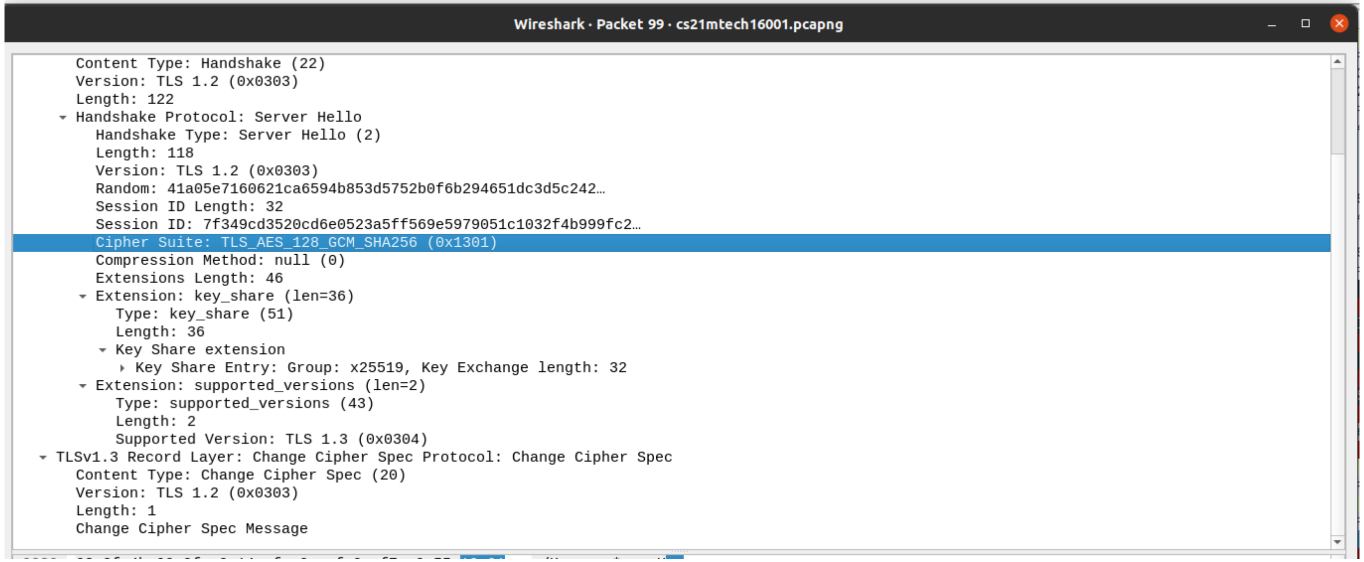
TLS_AES_128_GCM_SHA256 = This cipher suite is used in TLS1.3 that uses Diffie-Hellman Key exchange protocol with AES128 symmetric cryptography in GCM (Galois/Counter Mode) to encrypt the data and SHA256 to generate a digest and maintain the message integrity.

TLS_AES_256_GCM_SHA384 = This cipher suite is similar to the one above but it uses increased symmetric/session key length (increased from 128 to 256) and also increased digest length (increased from 256 to 384). Apart from that, this cipher suite is also an example of a TLS1.3 cipher suite that uses DH key exchange protocol to generate session key encrypt the data using AES in GCM mode followed SHA to maintain message integrity.

TLS_RSA_WITH_AES_128_CBC_SHA = This cipher suite is different from the ones discussed above because it uses a non-ephemeral key exchange protocol to exchange session keys. This suite uses RSA key pairs to authenticate the server and client (endpoints) and also to exchange the agreed-upon session keys between them. This suite also uses symmetric cryptography to encrypt data having a key length of 128 bits in Cipher Block Chaining Mode (CBC Mode). Unlike the above cipher suites, it uses SHA 1 hashing algorithm to generate a digest of 160 bits length (20 bytes) that get signed to preserve the message integrity.

TLS_RSA_WITH_AES_256_CBC_SHA = Only the key length for AES symmetric encryption has changed between the immediate previous cipher suite.

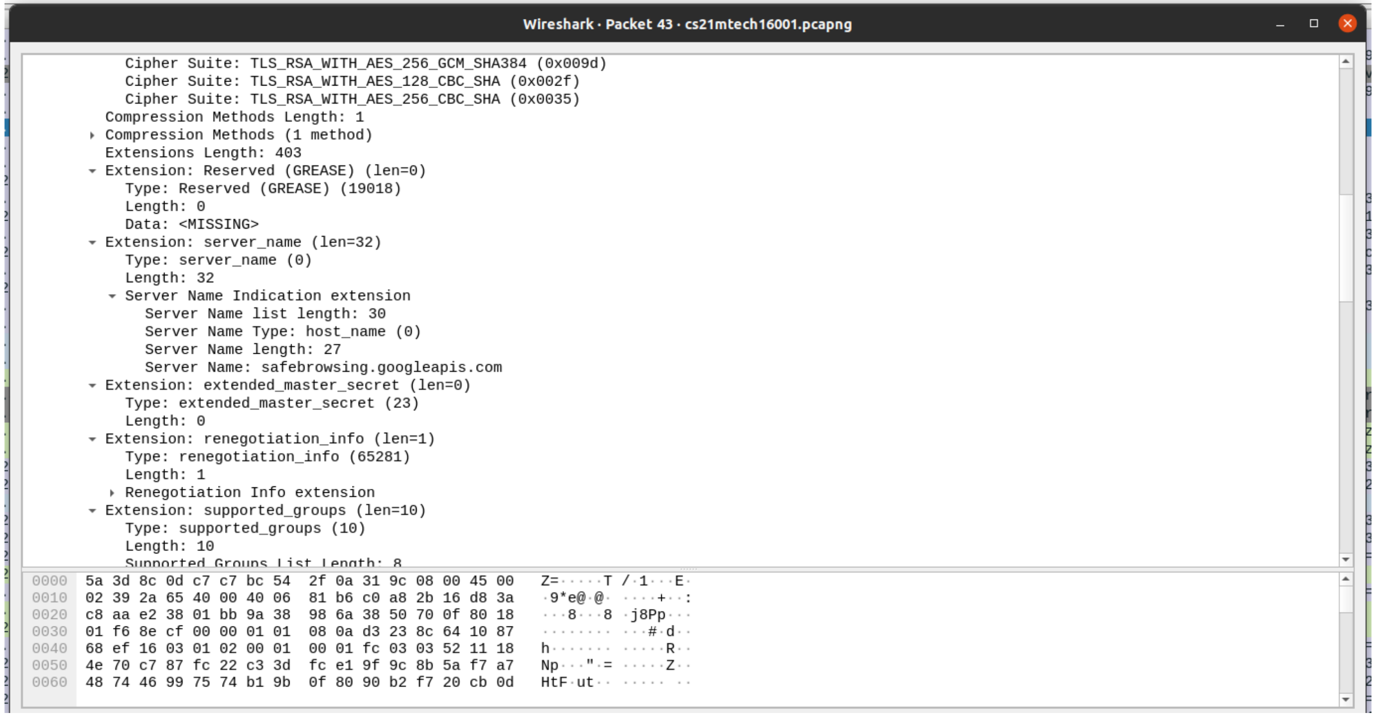
The cipher suite selected by the server is *TLS_AES_128_GCM_SHA256* , as shown in the screenshot below:



```
      Content Type: Handshake (22)
      Version: TLS 1.2 (0x0303)
      Length: 122
    ▾ Handshake Protocol: Server Hello
        Handshake Type: Server Hello (2)
        Length: 118
        Version: TLS 1.2 (0x0303)
        Random: 41a05e7160621ca6594b853d5752b0f6b294651dc3d5c242…
        Session ID Length: 32
        Session ID: 7f349cd3520cd6e0523a5ff569e5979051c1032f4b999fc2…
        Cipher Suite: TLS_AES_128_GCM_SHA256 (0x1301)
        Compression Method: null (0)
        Extensions Length: 46
      ▾ Extension: key_share (len=36)
          Type: key_share (51)
          Length: 36
        ▾ Key Share extension
          ▸ Key Share Entry: Group: x25519, Key Exchange length: 32
      ▾ Extension: supported_versions (len=2)
          Type: supported_versions (43)
          Length: 2
          Supported Version: TLS 1.3 (0x0304)
  ▾ TLSv1.3 Record Layer: Change Cipher Spec Protocol: Change Cipher Spec
      Content Type: Change Cipher Spec (20)
      Version: TLS 1.2 (0x0303)
      Length: 1
      Change Cipher Spec Message
```

5. What is the SNI value in ClientHello Record? What's its purpose? In other words, why is the client advertising it to the server?

It is possible for a webserver to host multiple websites with different domains. Being different websites with different domains, each of them might have different digital certificates that need to be sent when a client hello for that website is received from a client. Since all these websites are hosted in the same webserver, each of them will be redirected to the same IP address (IP of the webserver) which is not enough to identify the website and DC to be sent to the client. This problem occurs because in TLS, the handshake protocol requires verification of certificate of a specific website but there is no indication of which website is being verified. The same problem is solved using SNI value or Server Name indication value which indicates the website that the client is trying to access from a web server that is hosting multiple websites.



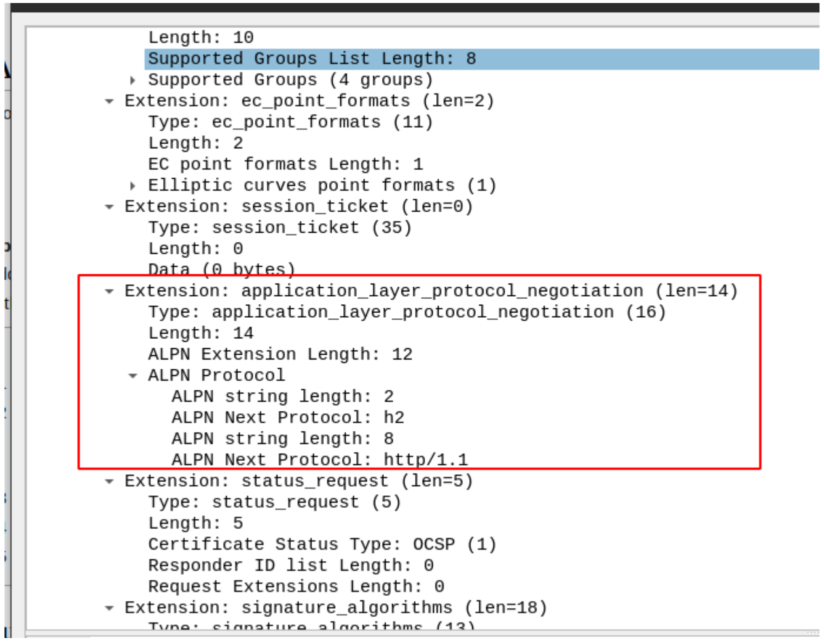
Wireshark · Packet 43 · cs21mtech16001.pcapng

```
          Cipher Suite: TLS_RSA_WITH_AES_256_GCM_SHA384 (0x009d)
          Cipher Suite: TLS_RSA_WITH_AES_128_CBC_SHA (0x002f)
          Cipher Suite: TLS_RSA_WITH_AES_256_CBC_SHA (0x0035)
        Compression Methods Length: 1
      ▸ Compression Methods (1 method)
        Extensions Length: 403
      ▾ Extension: Reserved (GREASE) (len=0)
          Type: Reserved (GREASE) (19018)
          Length: 0
          Data: <MISSING>
      ▾ Extension: server_name (len=32)
          Type: server_name (0)
          Length: 32
        ▾ Server Name Indication extension
            Server Name list length: 30
            Server Name Type: host_name (0)
            Server Name length: 27
            Server Name: safebrowsing.googleapis.com
      ▾ Extension: extended_master_secret (len=0)
          Type: extended_master_secret (23)
          Length: 0
      ▾ Extension: renegotiation_info (len=1)
          Type: renegotiation_info (65281)
          Length: 1
        ▸ Renegotiation Info extension
      ▾ Extension: supported_groups (len=10)
          Type: supported_groups (10)
          Length: 10
          Supported Groups List Length: 8
```

```
0000  5a 3d 8c 0d c7 c7 bc 54  2f 0a 31 9c 08 00 45 00   Z=·····T /·1···E·
0010  02 39 2a 65 40 00 40 06  81 b6 c0 a8 2b 16 d8 3a   ·9*e@·@· ····+··:
0020  c8 aa e2 38 01 bb 9a 38  98 6a 38 50 70 0f 80 18   ···8···8 ·j8Pp···
0030  01 f6 8e cf 00 00 01 01  08 0a d3 23 8c 64 10 87   ········ ···#·d··
0040  68 ef 16 03 01 02 00 01  00 01 fc 03 03 52 11 18   h······· ·····R··
0050  4e 70 c7 87 fc 22 c3 3d  fc e1 9f 9c 8b 5a f7 a7   Np···"·= ·····Z··
0060  48 74 46 99 75 74 b1 9b  0f 80 90 b2 f7 20 cb 0d   HtF·ut·· ····· ··
```

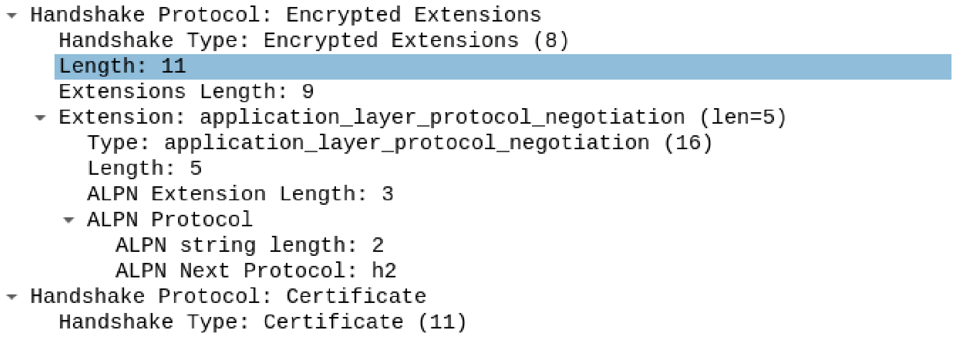
As we can see in the figure above, there is a clear indication of the website that the client is trying to access. Now, if even the webserver contains multiple websites hosted, the server will know which website is trying to get accessed to and which digital certificate to send.

6. What is the ALPN value(s) in ClientHello Record? What's its purpose? Which one did the server select?
Application Layer Protocol Negotiation extension in Client Hello record is used to negotiate (in priority order) which application layer protocol (HTTP protocol) to use over a TLS connection. This is added in the Client Hello record itself as it can be used to prevent additional RTT to decide which protocol to use later on. The figure below shows that the preferred protocol to use is HTTP/2 and HTTP/1.1 after that in that order.



```
          Length: 10
          Supported Groups List Length: 8
        ▸ Supported Groups (4 groups)
      ▾ Extension: ec_point_formats (len=2)
          Type: ec_point_formats (11)
          Length: 2
          EC point formats Length: 1
        ▸ Elliptic curves point formats (1)
      ▾ Extension: session_ticket (len=0)
          Type: session_ticket (35)
          Length: 0
          Data (0 bytes)
      ▾ Extension: application_layer_protocol_negotiation (len=14)
          Type: application_layer_protocol_negotiation (16)
          Length: 14
          ALPN Extension Length: 12
        ▾ ALPN Protocol
            ALPN string length: 2
            ALPN Next Protocol: h2
            ALPN string length: 8
            ALPN Next Protocol: http/1.1
      ▾ Extension: status_request (len=5)
          Type: status_request (5)
          Length: 5
          Certificate Status Type: OCSP (1)
          Responder ID list Length: 0
          Request Extensions Length: 0
      ▾ Extension: signature_algorithms (len=18)
          Type: signature_algorithms (13)
```

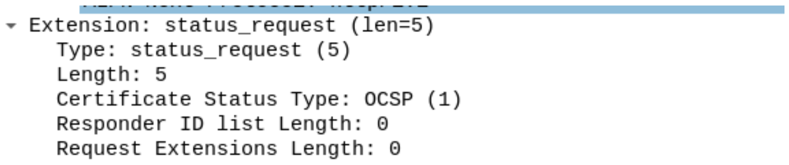
Similarly, we can see from the figure below that the server selected HTTP/2 from the provided ALPN protocol list.



```
▾ Handshake Protocol: Encrypted Extensions
    Handshake Type: Encrypted Extensions (8)
    Length: 11
    Extensions Length: 9
  ▾ Extension: application_layer_protocol_negotiation (len=5)
      Type: application_layer_protocol_negotiation (16)
      Length: 5
      ALPN Extension Length: 3
    ▾ ALPN Protocol
        ALPN string length: 2
        ALPN Next Protocol: h2
▾ Handshake Protocol: Certificate
    Handshake Type: Certificate (11)
```

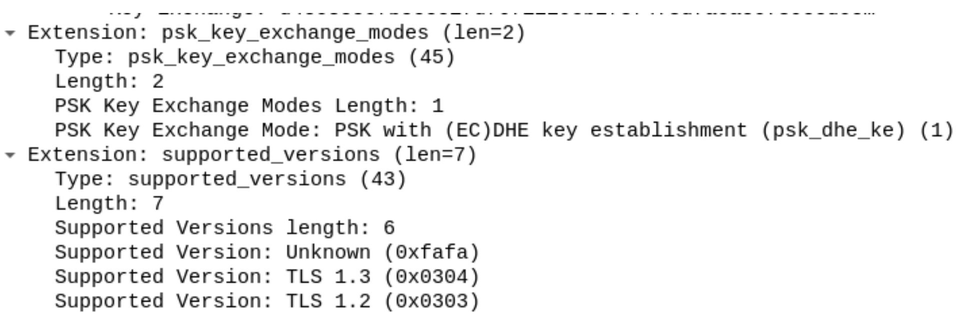
7. Does the ClientHello contain status_request, supported_versions, psk_key_exchange_modes extensions? If so, what do they convey to the server?

Yes the ClientHello contains status_request, supported_versions, psk_key_exchange_modes extensions.



```
▾ Extension: status_request (len=5)
    Type: status_request (5)
    Length: 5
    Certificate Status Type: OCSP (1)
    Responder ID list Length: 0
    Request Extensions Length: 0
```

Status request: The status request extension in the ClientHello message indicates the status of the certificate or the mechanism to check it like OCSP or CRL.

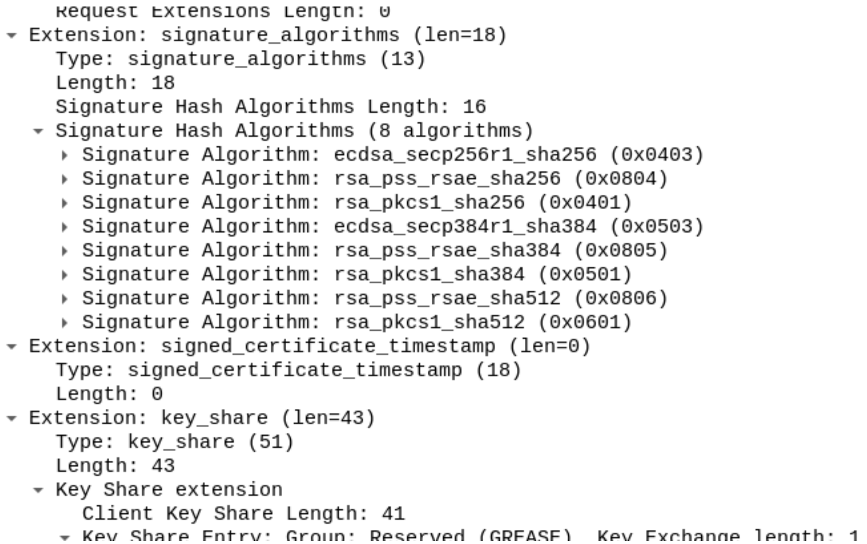


```
▾ Extension: psk_key_exchange_modes (len=2)
    Type: psk_key_exchange_modes (45)
    Length: 2
    PSK Key Exchange Modes Length: 1
    PSK Key Exchange Mode: PSK with (EC)DHE key establishment (psk_dhe_ke) (1)
▾ Extension: supported_versions (len=7)
    Type: supported_versions (43)
    Length: 7
    Supported Versions length: 6
    Supported Version: Unknown (0xfafa)
    Supported Version: TLS 1.3 (0x0304)
    Supported Version: TLS 1.2 (0x0303)
```

Supported versions = This extension indicates which TLS versions are supported by the client browser to establish the secure connection.
psk_key_exchange_modes extensions = This extension indicates to the server which key exchange modes like RSA, (EC)DHE are supported with the

pre-shared key. This extension comes along with a pre-shared key extension which will further be used to generate the Handshake Secret based on the mode indicated in the extension.

8. Does ClientHello Record contain the Signature_algorithms extension? What’s its purpose?

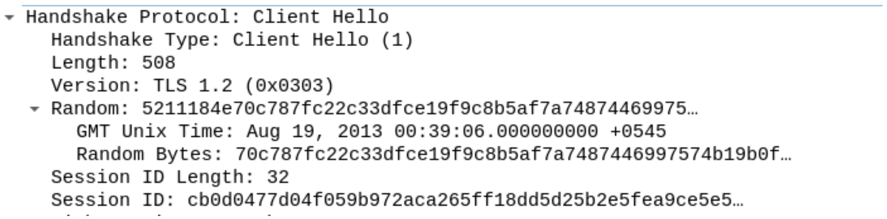


```
        Request Extensions Length: 0
    ▾ Extension: signature_algorithms (len=18)
        Type: signature_algorithms (13)
        Length: 18
        Signature Hash Algorithms Length: 16
      ▾ Signature Hash Algorithms (8 algorithms)
        ▸ Signature Algorithm: ecdsa_secp256r1_sha256 (0x0403)
        ▸ Signature Algorithm: rsa_pss_rsae_sha256 (0x0804)
        ▸ Signature Algorithm: rsa_pkcs1_sha256 (0x0401)
        ▸ Signature Algorithm: ecdsa_secp384r1_sha384 (0x0503)
        ▸ Signature Algorithm: rsa_pss_rsae_sha384 (0x0805)
        ▸ Signature Algorithm: rsa_pkcs1_sha384 (0x0501)
        ▸ Signature Algorithm: rsa_pss_rsae_sha512 (0x0806)
        ▸ Signature Algorithm: rsa_pkcs1_sha512 (0x0601)
    ▾ Extension: signed_certificate_timestamp (len=0)
        Type: signed_certificate_timestamp (18)
        Length: 0
    ▾ Extension: key_share (len=43)
        Type: key_share (51)
        Length: 43
      ▾ Key Share extension
          Client Key Share Length: 41
        ▾ Key Share Entry: Group: Reserved (GREASE), Key Exchange length: 1
```

Yes, the ClientHello Record contains a signature algorithms extension that contains a number of signature algorithms (including the algorithms to generate the digest and the one used to sign it) that can be used to sign a certificate or generate the digest or simply for digital signatures.

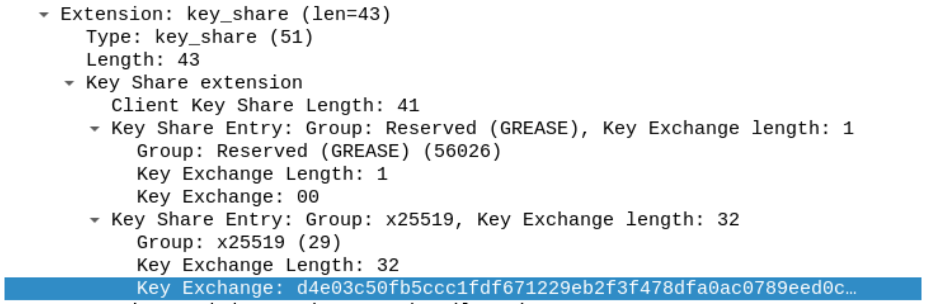
9. Does the client offer any Random number, key share, Supported Groups, and PSK in ClientHello Record? How will be these used by the Server?

The client random number shared by the client will be used to generate the master secret which in turn will be used to generate the key material. The same random number or noonces will be used to prevent any sort of replay attacks.

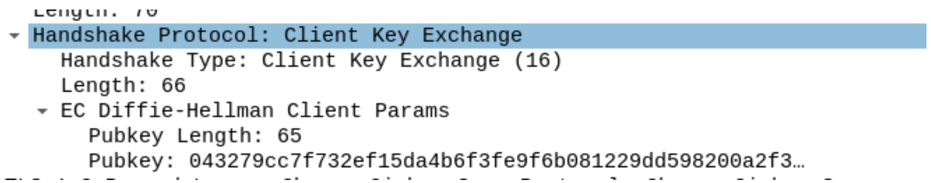


```
▾ Handshake Protocol: Client Hello
    Handshake Type: Client Hello (1)
    Length: 508
    Version: TLS 1.2 (0x0303)
  ▾ Random: 5211184e70c787fc22c33dfce19f9c8b5af7a74874469975…
      GMT Unix Time: Aug 19, 2013 00:39:06.000000000 +0545
      Random Bytes: 70c787fc22c33dfce19f9c8b5af7a7487446997574b19b0f…
    Session ID Length: 32
    Session ID: cb0d0477d04f059b972aca265ff18dd5d25b2e5fea9ce5e5…
```

The Key shares shared by the client will be used to generate the PMS during the Key exchange protocol following the key exchange protocol selected by the server. If there are global parameters that can be used to generate the PMS, the client will advertise it along with key share to the server so that it can be used to get the PMS. According to RFC 8446, key share contains the endpoint's cryptographic parameters.

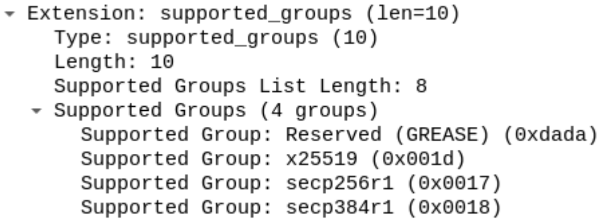


```
▾ Extension: key_share (len=43)
    Type: key_share (51)
    Length: 43
  ▾ Key Share extension
      Client Key Share Length: 41
    ▾ Key Share Entry: Group: Reserved (GREASE), Key Exchange length: 1
        Group: Reserved (GREASE) (56026)
        Key Exchange Length: 1
        Key Exchange: 00
    ▾ Key Share Entry: Group: x25519, Key Exchange length: 32
        Group: x25519 (29)
        Key Exchange Length: 32
        Key Exchange: d4e03c50fb5ccc1fdf671229eb2f3f478dfa0ac0789eed0c…
```



```
  Length: 70
▾ Handshake Protocol: Client Key Exchange
    Handshake Type: Client Key Exchange (16)
    Length: 66
  ▾ EC Diffie-Hellman Client Params
      Pubkey Length: 65
      Pubkey: 043279cc7f732ef15da4b6f3fe9f6b081229dd598200a2f3…
```

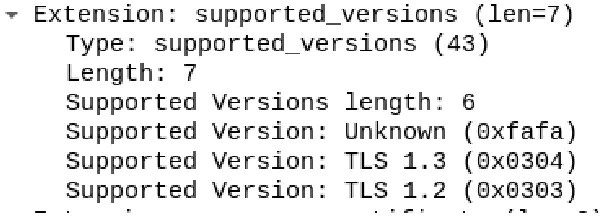
The supported groups' extension in the client hello message indicates the name of the groups which the client supports for key exchange in preferential order.



```
▾ Extension: supported_groups (len=10)
    Type: supported_groups (10)
    Length: 10
    Supported Groups List Length: 8
  ▾ Supported Groups (4 groups)
      Supported Group: Reserved (GREASE) (0xdada)
      Supported Group: x25519 (0x001d)
      Supported Group: secp256r1 (0x0017)
      Supported Group: secp384r1 (0x0018)
```

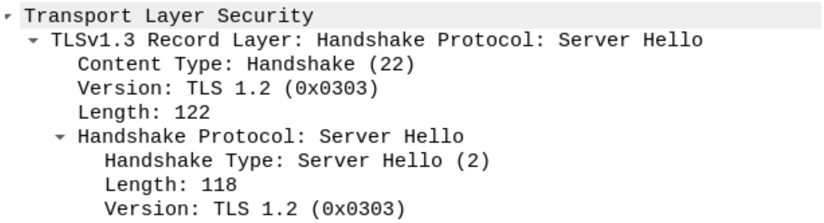
The client doesn't offer any out-of-bounds PSK to the server. PSK will be used by the server to generate the Early secret, handshake secret as well as Master Secret using multiple additional parameters. The PSK can either be a session ticket corresponding to a previous conversation or any key that is agreed between the communication parties prior to the communication by other agreement forms.

10. What TLS versions your browser/client is supporting? Which one did the server select?

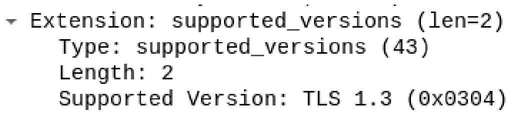


```
▾ Extension: supported_versions (len=7)
    Type: supported_versions (43)
    Length: 7
    Supported Versions length: 6
    Supported Version: Unknown (0xfafa)
    Supported Version: TLS 1.3 (0x0304)
    Supported Version: TLS 1.2 (0x0303)
```

As the screenshot indicates, the client/my browser is supporting TLS1.2, TLS1.3, and an unknown (for backward compatibility and extensibility). The server selected TLS 1.2 for establishing a secure connection although it supported TLS1.3.

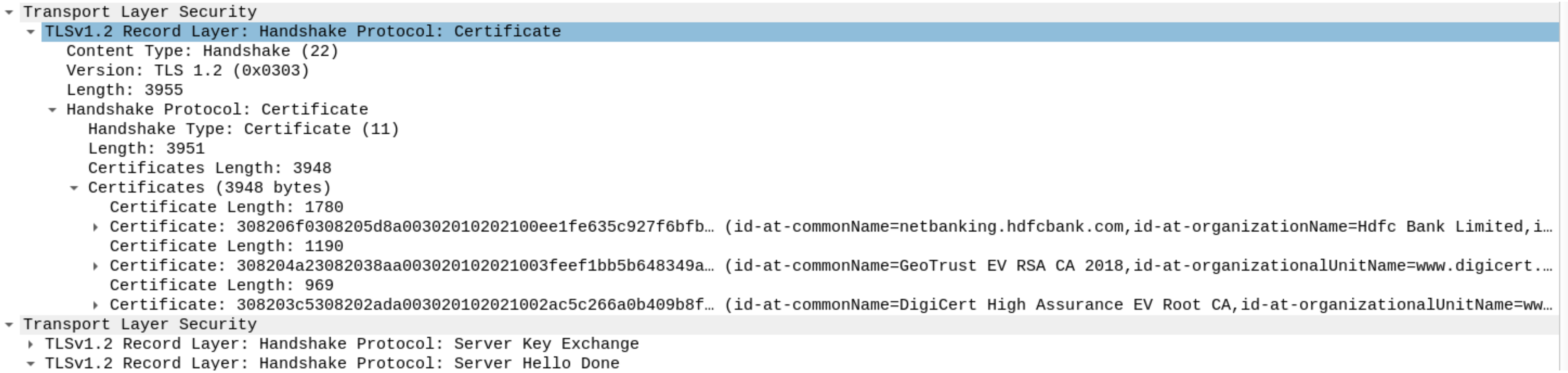


```
▾ Transport Layer Security
  ▾ TLSv1.3 Record Layer: Handshake Protocol: Server Hello
      Content Type: Handshake (22)
      Version: TLS 1.2 (0x0303)
      Length: 122
    ▾ Handshake Protocol: Server Hello
        Handshake Type: Server Hello (2)
        Length: 118
        Version: TLS 1.2 (0x0303)
```



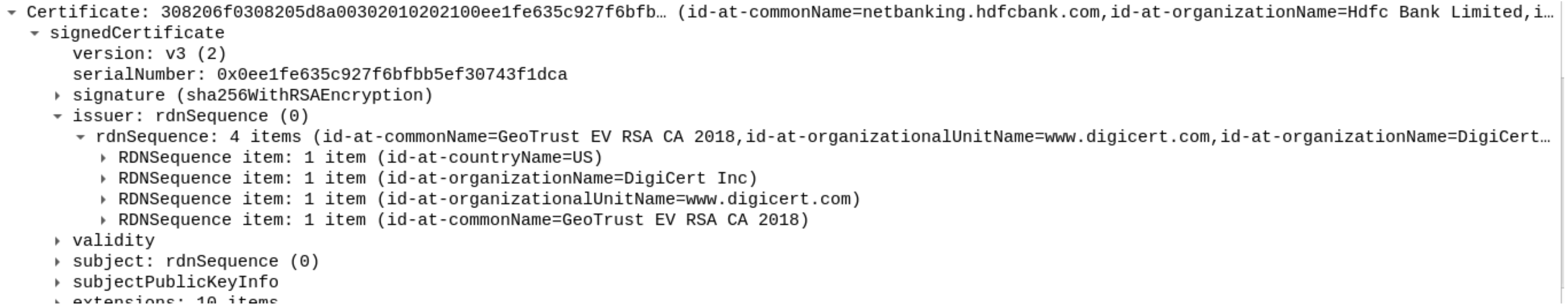
```
▾ Extension: supported_versions (len=2)
    Type: supported_versions (43)
    Length: 2
    Supported Version: TLS 1.3 (0x0304)
```

11. Look at Certificate Record from the server to the client: How many certificates did the server return and how are they related? Who is the issuer of the Bank’s certificate? What type of public key the bank is using?



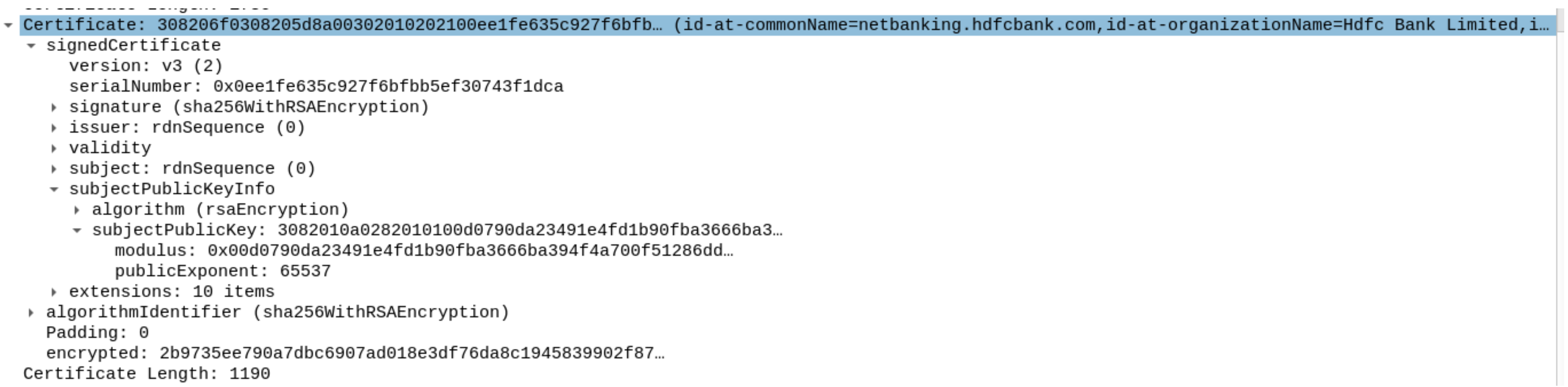
```
▾ Transport Layer Security
  ▾ TLSv1.2 Record Layer: Handshake Protocol: Certificate
      Content Type: Handshake (22)
      Version: TLS 1.2 (0x0303)
      Length: 3955
    ▾ Handshake Protocol: Certificate
        Handshake Type: Certificate (11)
        Length: 3951
        Certificates Length: 3948
      ▾ Certificates (3948 bytes)
          Certificate Length: 1780
        ▸ Certificate: 308206f0308205d8a00302010202100ee1fe635c927f6bfb… (id-at-commonName=netbanking.hdfcbank.com,id-at-organizationName=Hdfc Bank Limited,i…
          Certificate Length: 1190
        ▸ Certificate: 308204a23082038aa003020102021003feef1bb5b648349a… (id-at-commonName=GeoTrust EV RSA CA 2018,id-at-organizationalUnitName=www.digicert.…
          Certificate Length: 969
        ▸ Certificate: 308203c5308202ada003020102021002ac5c266a0b409b8f… (id-at-commonName=DigiCert High Assurance EV Root CA,id-at-organizationalUnitName=ww…
▾ Transport Layer Security
  ▸ TLSv1.2 Record Layer: Handshake Protocol: Server Key Exchange
  ▾ TLSv1.2 Record Layer: Handshake Protocol: Server Hello Done
```

As we can see from the screenshot above, there are three certificates being sent from the server to the client. The subject's name in each of these certificates indicates the party for which the certificate is issued. The topmost certificate is the certificate of the website that is being accessed i.e. netbanking.hdfcbank.com. The certificate below that is the certificate of the intermediate CA that has signed the certificate of the website is accessed (HDFC Net Banking). Here, the intermediate CA is GeoTrust EV RSA CA 2018. Similarly, the final certificate is the certificate of the root CA that is self-signed. Here, the root CA is Digicert High Assurance EV Root CA. So these certificates are in a chain of signed certificates.



```
▾ Certificate: 308206f0308205d8a00302010202100ee1fe635c927f6bfb… (id-at-commonName=netbanking.hdfcbank.com,id-at-organizationName=Hdfc Bank Limited,i…
  ▾ signedCertificate
      version: v3 (2)
      serialNumber: 0x0ee1fe635c927f6bfbb5ef30743f1dca
    ▸ signature (sha256WithRSAEncryption)
    ▾ issuer: rdnSequence (0)
      ▾ rdnSequence: 4 items (id-at-commonName=GeoTrust EV RSA CA 2018,id-at-organizationalUnitName=www.digicert.com,id-at-organizationName=DigiCert…
        ▸ RDNSequence item: 1 item (id-at-countryName=US)
        ▸ RDNSequence item: 1 item (id-at-organizationName=DigiCert Inc)
        ▸ RDNSequence item: 1 item (id-at-organizationalUnitName=www.digicert.com)
        ▸ RDNSequence item: 1 item (id-at-commonName=GeoTrust EV RSA CA 2018)
    ▸ validity
    ▸ subject: rdnSequence (0)
    ▸ subjectPublicKeyInfo
    ▸ extensions: 10 items
```

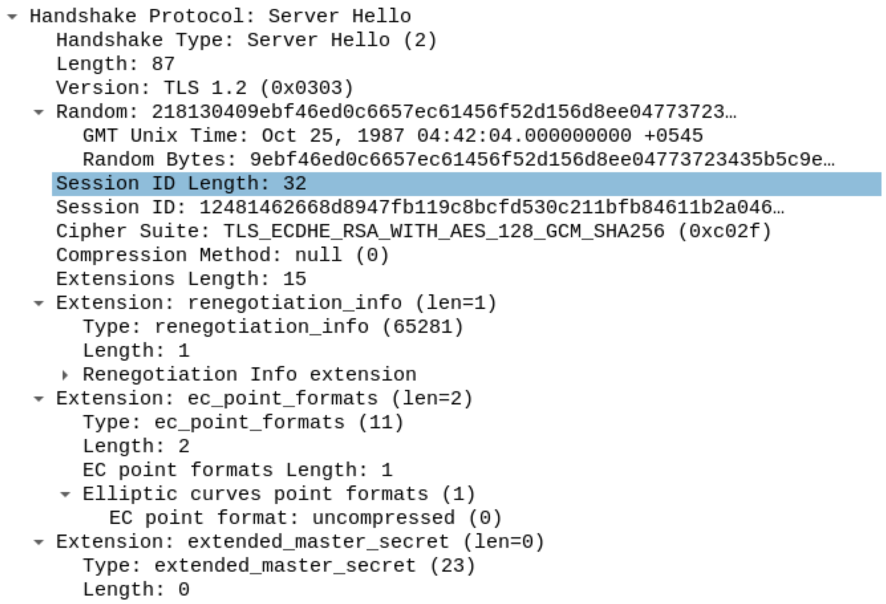
Geo Trust EV RSA CA 2018 is the issuer of the bank's certificate.



```
▾ Certificate: 308206f0308205d8a00302010202100ee1fe635c927f6bfb… (id-at-commonName=netbanking.hdfcbank.com,id-at-organizationName=Hdfc Bank Limited,i…
  ▾ signedCertificate
      version: v3 (2)
      serialNumber: 0x0ee1fe635c927f6bfbb5ef30743f1dca
    ▸ signature (sha256WithRSAEncryption)
    ▸ issuer: rdnSequence (0)
    ▸ validity
    ▸ subject: rdnSequence (0)
    ▾ subjectPublicKeyInfo
      ▸ algorithm (rsaEncryption)
      ▾ subjectPublicKey: 3082010a0282010100d0790da23491e4fd1b90fba3666ba3…
          modulus: 0x00d0790da23491e4fd1b90fba3666ba394f4a700f51286dd…
          publicExponent: 65537
    ▸ extensions: 10 items
  ▸ algorithmIdentifier (sha256WithRSAEncryption)
    Padding: 0
    encrypted: 2b9735ee790a7dbc6907ad018e3df76da8c1945839902f87…
  Certificate Length: 1190
```

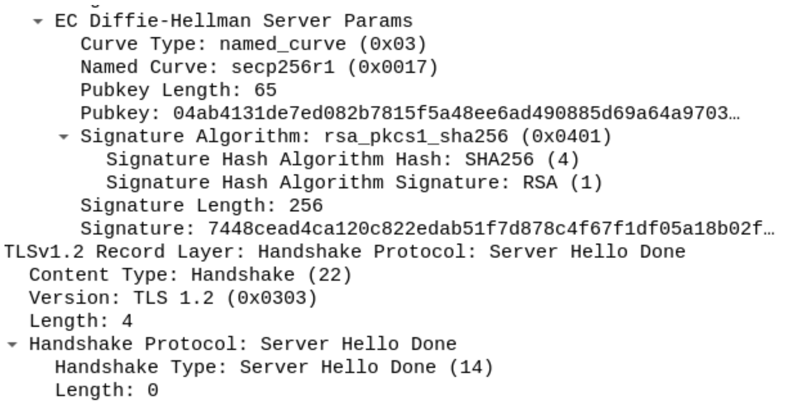
According to the screenshot above, the bank is using the RSA Public key of 2048 bits.

12. Comment on the key exchange algorithm agreed upon, what are the parameters that got exchanged between client and server to derive the session keys.



```
▾ Handshake Protocol: Server Hello
    Handshake Type: Server Hello (2)
    Length: 87
    Version: TLS 1.2 (0x0303)
  ▾ Random: 218130409ebf46ed0c6657ec61456f52d156d8ee04773723…
      GMT Unix Time: Oct 25, 1987 04:42:04.000000000 +0545
      Random Bytes: 9ebf46ed0c6657ec61456f52d156d8ee04773723435b5c9e…
    Session ID Length: 32
    Session ID: 12481462668d8947fb119c8bcfd530c211bfb84611b2a046…
    Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)
    Compression Method: null (0)
    Extensions Length: 15
  ▾ Extension: renegotiation_info (len=1)
      Type: renegotiation_info (65281)
      Length: 1
    ▸ Renegotiation Info extension
  ▾ Extension: ec_point_formats (len=2)
      Type: ec_point_formats (11)
      Length: 2
      EC point formats Length: 1
    ▾ Elliptic curves point formats (1)
        EC point format: uncompressed (0)
  ▾ Extension: extended_master_secret (len=0)
      Type: extended_master_secret (23)
      Length: 0
```

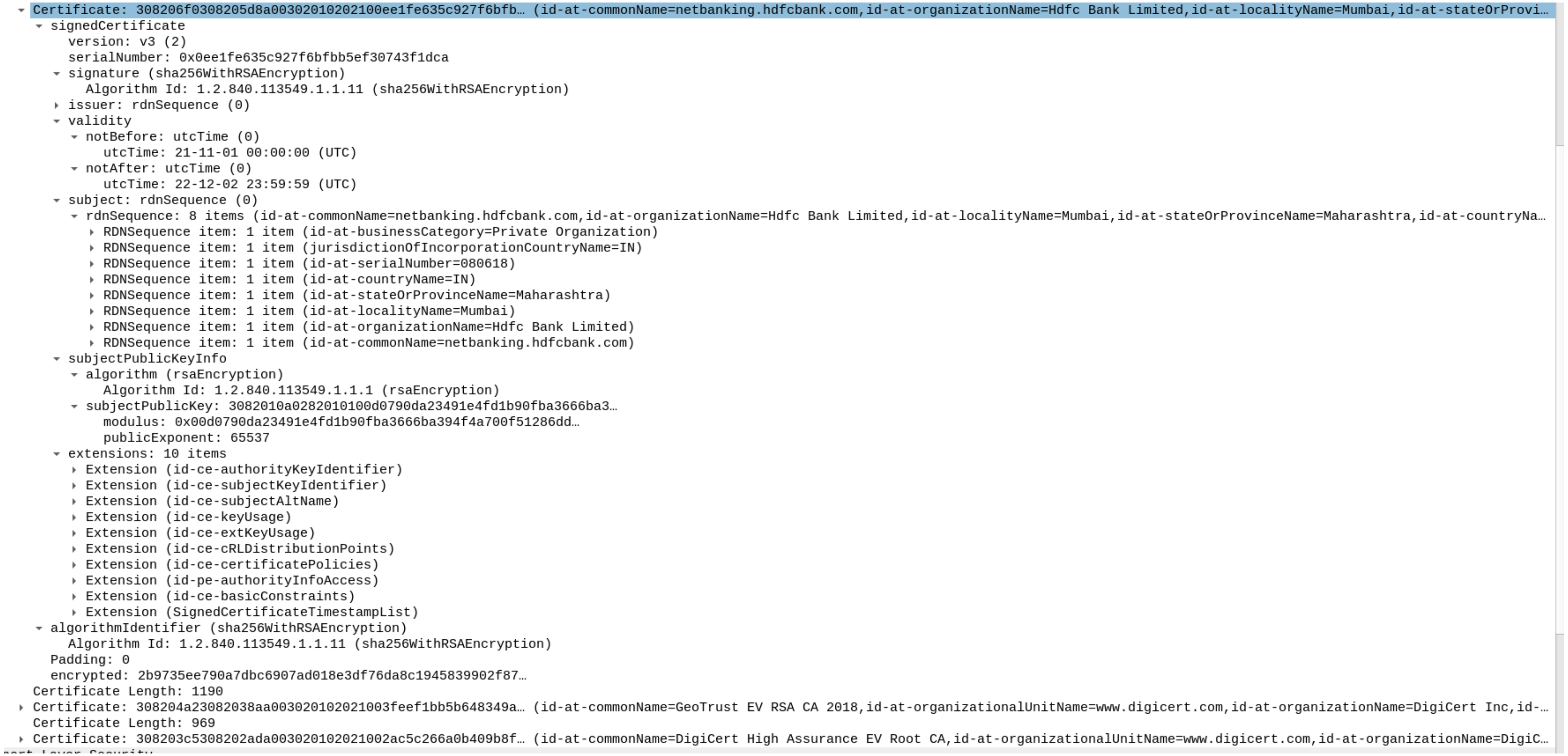
Looking at server hello ECDHE (Elliptic Curve Diffie-Hellman Key Exchange) protocol is used to exchange keys between them. Similarly, the above screenshot also shows the elliptic curve constraints to follow to generate the keys.



```
  ▾ EC Diffie-Hellman Server Params
      Curve Type: named_curve (0x03)
      Named Curve: secp256r1 (0x0017)
      Pubkey Length: 65
      Pubkey: 04ab4131de7ed082b7815f5a48ee6ad490885d69a64a9703…
    ▾ Signature Algorithm: rsa_pkcs1_sha256 (0x0401)
        Signature Hash Algorithm Hash: SHA256 (4)
        Signature Hash Algorithm Signature: RSA (1)
      Signature Length: 256
      Signature: 7448cead4ca120c822edab51f7d878c4f67f1df05a18b02f…
TLSv1.2 Record Layer: Handshake Protocol: Server Hello Done
  Content Type: Handshake (22)
  Version: TLS 1.2 (0x0303)
  Length: 4
▾ Handshake Protocol: Server Hello Done
    Handshake Type: Server Hello Done (14)
    Length: 0
```

Similarly, there are EC Diffie-Hellman parameters also being sent from server to client indicating the groups, length of the key, and more so that the client can use it to generate the session keys using these parameters.

13. Which certificate type (DV/OV/EV) the bank is using?

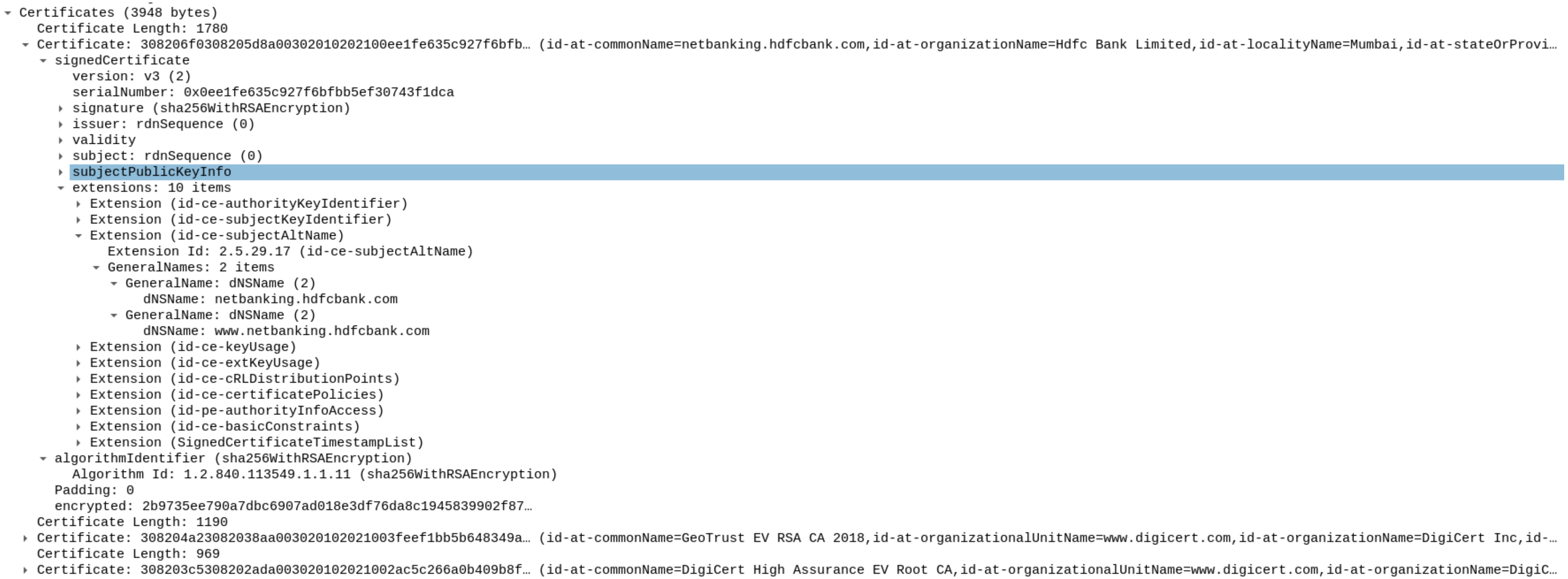


```
▾ Certificate: 308206f0308205d8a00302010202100ee1fe635c927f6bfb… (id-at-commonName=netbanking.hdfcbank.com,id-at-organizationName=Hdfc Bank Limited,id-at-localityName=Mumbai,id-at-stateOrProvi…
  ▾ signedCertificate
      version: v3 (2)
      serialNumber: 0x0ee1fe635c927f6bfbb5ef30743f1dca
    ▾ signature (sha256WithRSAEncryption)
        Algorithm Id: 1.2.840.113549.1.1.11 (sha256WithRSAEncryption)
    ▸ issuer: rdnSequence (0)
    ▾ validity
      ▾ notBefore: utcTime (0)
          utcTime: 21-11-01 00:00:00 (UTC)
      ▾ notAfter: utcTime (0)
          utcTime: 22-12-02 23:59:59 (UTC)
    ▾ subject: rdnSequence (0)
      ▾ rdnSequence: 8 items (id-at-commonName=netbanking.hdfcbank.com,id-at-organizationName=Hdfc Bank Limited,id-at-localityName=Mumbai,id-at-stateOrProvinceName=Maharashtra,id-at-countryNa…
        ▸ RDNSequence item: 1 item (id-at-businessCategory=Private Organization)
        ▸ RDNSequence item: 1 item (jurisdictionOfIncorporationCountryName=IN)
        ▸ RDNSequence item: 1 item (id-at-serialNumber=080618)
        ▸ RDNSequence item: 1 item (id-at-countryName=IN)
        ▸ RDNSequence item: 1 item (id-at-stateOrProvinceName=Maharashtra)
        ▸ RDNSequence item: 1 item (id-at-localityName=Mumbai)
        ▸ RDNSequence item: 1 item (id-at-organizationName=Hdfc Bank Limited)
        ▸ RDNSequence item: 1 item (id-at-commonName=netbanking.hdfcbank.com)
    ▾ subjectPublicKeyInfo
      ▾ algorithm (rsaEncryption)
          Algorithm Id: 1.2.840.113549.1.1.1 (rsaEncryption)
      ▾ subjectPublicKey: 3082010a0282010100d0790da23491e4fd1b90fba3666ba3…
          modulus: 0x00d0790da23491e4fd1b90fba3666ba394f4a700f51286dd…
          publicExponent: 65537
    ▾ extensions: 10 items
      ▸ Extension (id-ce-authorityKeyIdentifier)
      ▸ Extension (id-ce-subjectKeyIdentifier)
      ▸ Extension (id-ce-subjectAltName)
      ▸ Extension (id-ce-keyUsage)
      ▸ Extension (id-ce-extKeyUsage)
      ▸ Extension (id-ce-cRLDistributionPoints)
      ▸ Extension (id-ce-certificatePolicies)
      ▸ Extension (id-pe-authorityInfoAccess)
      ▸ Extension (id-ce-basicConstraints)
      ▸ Extension (SignedCertificateTimestampList)
  ▾ algorithmIdentifier (sha256WithRSAEncryption)
      Algorithm Id: 1.2.840.113549.1.1.11 (sha256WithRSAEncryption)
    Padding: 0
    encrypted: 2b9735ee790a7dbc6907ad018e3df76da8c1945839902f87…
  Certificate Length: 1190
▸ Certificate: 308204a23082038aa003020102021003feef1bb5b648349a… (id-at-commonName=GeoTrust EV RSA CA 2018,id-at-organizationalUnitName=www.digicert.com,id-at-organizationName=DigiCert Inc,id-…
  Certificate Length: 969
▸ Certificate: 308203c5308202ada003020102021002ac5c266a0b409b8f… (id-at-commonName=DigiCert High Assurance EV Root CA,id-at-organizationalUnitName=www.digicert.com,id-at-organizationName=DigiC…
```

As we can see from the screenshot above, the certificate contains detailed information like business category, locality, the jurisdiction of incorporation country name, states, and everything. To conduct and validate such information, the only validation method is to carry out Extended Validation (EV). Any other certificate without EV won't contain named parameters like jurisdictions, business categories, and other detailed information.

Reference

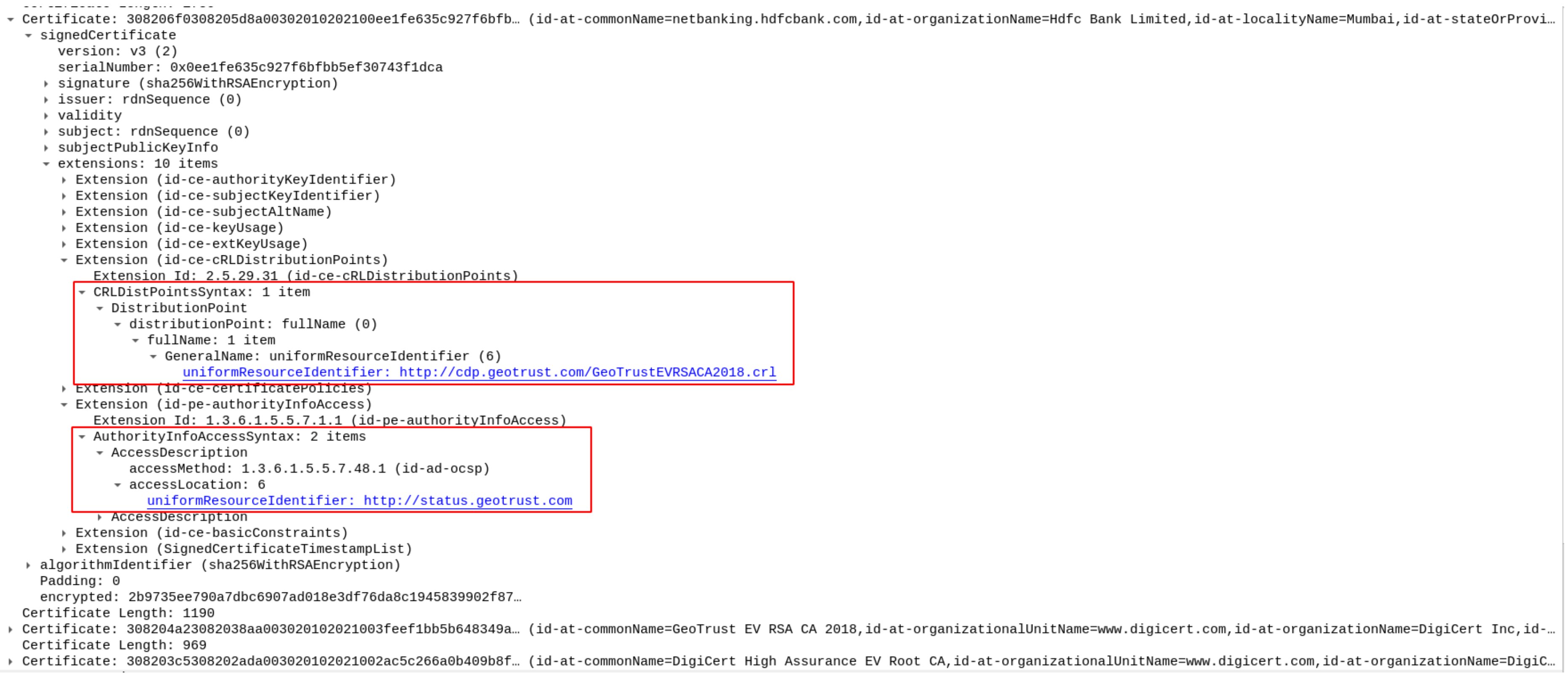
14. Which certificate type (single or multi-domain or wild-card) the bank is using?



```
▾ Certificates (3948 bytes)
    Certificate Length: 1780
  ▾ Certificate: 308206f0308205d8a00302010202100ee1fe635c927f6bfb… (id-at-commonName=netbanking.hdfcbank.com,id-at-organizationName=Hdfc Bank Limited,id-at-localityName=Mumbai,id-at-stateOrProvi…
    ▾ signedCertificate
        version: v3 (2)
        serialNumber: 0x0ee1fe635c927f6bfbb5ef30743f1dca
      ▸ signature (sha256WithRSAEncryption)
      ▸ issuer: rdnSequence (0)
      ▸ validity
      ▸ subject: rdnSequence (0)
      ▸ subjectPublicKeyInfo
      ▾ extensions: 10 items
        ▸ Extension (id-ce-authorityKeyIdentifier)
        ▸ Extension (id-ce-subjectKeyIdentifier)
        ▾ Extension (id-ce-subjectAltName)
            Extension Id: 2.5.29.17 (id-ce-subjectAltName)
          ▾ GeneralNames: 2 items
            ▾ GeneralName: dNSName (2)
                dNSName: netbanking.hdfcbank.com
            ▾ GeneralName: dNSName (2)
                dNSName: www.netbanking.hdfcbank.com
        ▸ Extension (id-ce-keyUsage)
        ▸ Extension (id-ce-extKeyUsage)
        ▸ Extension (id-ce-cRLDistributionPoints)
        ▸ Extension (id-ce-certificatePolicies)
        ▸ Extension (id-pe-authorityInfoAccess)
        ▸ Extension (id-ce-basicConstraints)
        ▸ Extension (SignedCertificateTimestampList)
    ▾ algorithmIdentifier (sha256WithRSAEncryption)
        Algorithm Id: 1.2.840.113549.1.1.11 (sha256WithRSAEncryption)
      Padding: 0
      encrypted: 2b9735ee790a7dbc6907ad018e3df76da8c1945839902f87…
    Certificate Length: 1190
  ▸ Certificate: 308204a23082038aa003020102021003feef1bb5b648349a… (id-at-commonName=GeoTrust EV RSA CA 2018,id-at-organizationalUnitName=www.digicert.com,id-at-organizationName=DigiCert Inc,id-…
    Certificate Length: 969
  ▸ Certificate: 308203c5308202ada003020102021002ac5c266a0b409b8f… (id-at-commonName=DigiCert High Assurance EV Root CA,id-at-organizationalUnitName=www.digicert.com,id-at-organizationName=DigiC…
```

The certificate contains multiple subject alternative names, SANs, without any asterisk (*) meaning the certificate is not a wild card certificate and since it contains multiple SANs, the certificate must be a multi-domain certificate.

15. How can the client check whether the certificate is revoked or not: OCSP/CRL? Does the server support OCSP stapling?

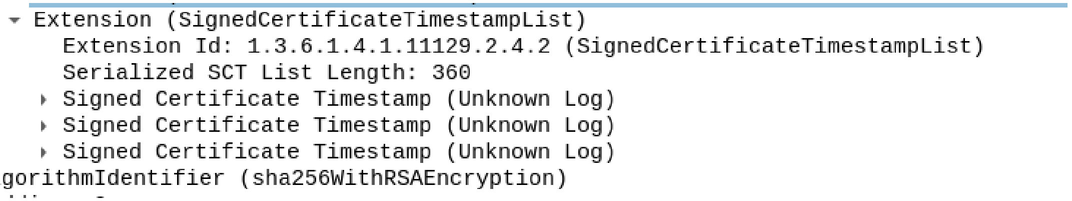


```
▾ Certificate: 308206f0308205d8a00302010202100ee1fe635c927f6bfb… (id-at-commonName=netbanking.hdfcbank.com,id-at-organizationName=Hdfc Bank Limited,id-at-localityName=Mumbai,id-at-stateOrProvi…
  ▾ signedCertificate
      version: v3 (2)
      serialNumber: 0x0ee1fe635c927f6bfbb5ef30743f1dca
    ▸ signature (sha256WithRSAEncryption)
    ▸ issuer: rdnSequence (0)
    ▸ validity
    ▸ subject: rdnSequence (0)
    ▸ subjectPublicKeyInfo
    ▾ extensions: 10 items
      ▸ Extension (id-ce-authorityKeyIdentifier)
      ▸ Extension (id-ce-subjectKeyIdentifier)
      ▸ Extension (id-ce-subjectAltName)
      ▸ Extension (id-ce-keyUsage)
      ▸ Extension (id-ce-extKeyUsage)
      ▾ Extension (id-ce-cRLDistributionPoints)
          Extension Id: 2.5.29.31 (id-ce-cRLDistributionPoints)
        ▾ CRLDistPointsSyntax: 1 item
          ▾ DistributionPoint
            ▾ distributionPoint: fullName (0)
              ▾ fullName: 1 item
                ▾ GeneralName: uniformResourceIdentifier (6)
                    uniformResourceIdentifier: http://cdp.geotrust.com/GeoTrustEVRSACA2018.crl
      ▸ Extension (id-ce-certificatePolicies)
      ▾ Extension (id-pe-authorityInfoAccess)
          Extension Id: 1.3.6.1.5.5.7.1.1 (id-pe-authorityInfoAccess)
        ▾ AuthorityInfoAccessSyntax: 2 items
          ▾ AccessDescription
              accessMethod: 1.3.6.1.5.5.7.48.1 (id-ad-ocsp)
            ▾ accessLocation: 6
                uniformResourceIdentifier: http://status.geotrust.com
          ▸ AccessDescription
      ▸ Extension (id-ce-basicConstraints)
      ▸ Extension (SignedCertificateTimestampList)
  ▸ algorithmIdentifier (sha256WithRSAEncryption)
    Padding: 0
    encrypted: 2b9735ee790a7dbc6907ad018e3df76da8c1945839902f87…
  Certificate Length: 1190
▸ Certificate: 308204a23082038aa003020102021003feef1bb5b648349a… (id-at-commonName=GeoTrust EV RSA CA 2018,id-at-organizationalUnitName=www.digicert.com,id-at-organizationName=DigiCert Inc,id-…
  Certificate Length: 969
▸ Certificate: 308203c5308202ada003020102021002ac5c266a0b409b8f… (id-at-commonName=DigiCert High Assurance EV Root CA,id-at-organizationalUnitName=www.digicert.com,id-at-organizationName=DigiC…
```

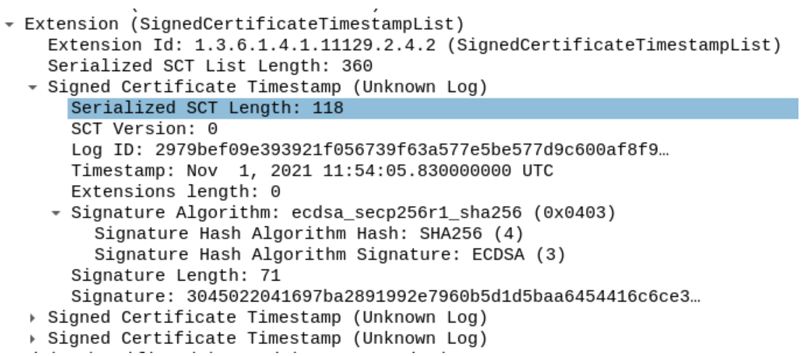
The client can check the status of the certificate using the CRL distribution points or the OCSP status URL. According to the screenshot above, we can see the links to the CRL distribution and OCSP server. The client can either index the CRL list or query the OCSP server to check the status of the digital certificate of the server.

No, the server doesn't support OCSP stapling. Couldn't find the evidence for not supporting the OCSP stapling in the trace but the

16. How many log servers logged the certificate of the bank? What role does the log server play in the Web PKI ecosystem? Refer: SCT extension.



```
▾ Extension (SignedCertificateTimestampList)
    Extension Id: 1.3.6.1.4.1.11129.2.4.2 (SignedCertificateTimestampList)
    Serialized SCT List Length: 360
  ▸ Signed Certificate Timestamp (Unknown Log)
  ▸ Signed Certificate Timestamp (Unknown Log)
  ▸ Signed Certificate Timestamp (Unknown Log)
▸ algorithmIdentifier (sha256WithRSAEncryption)
```



```
▾ Extension (SignedCertificateTimestampList)
    Extension Id: 1.3.6.1.4.1.11129.2.4.2 (SignedCertificateTimestampList)
    Serialized SCT List Length: 360
  ▾ Signed Certificate Timestamp (Unknown Log)
      Serialized SCT Length: 118
      SCT Version: 0
      Log ID: 2979bef09e393921f056739f63a577e5be577d9c600af8f9…
      Timestamp: Nov  1, 2021 11:54:05.830000000 UTC
      Extensions length: 0
    ▾ Signature Algorithm: ecdsa_secp256r1_sha256 (0x0403)
        Signature Hash Algorithm Hash: SHA256 (4)
        Signature Hash Algorithm Signature: ECDSA (3)
      Signature Length: 71
      Signature: 3045022041697ba2891992e7960b5d1d5baa6454416c6ce3…
  ▸ Signed Certificate Timestamp (Unknown Log)
  ▸ Signed Certificate Timestamp (Unknown Log)
```

As we can see from the above screenshot, there are three unknown Logs that have logged the issuance of the certificate of the bank. The Certificate Issuance logs are generated/recorded at multiple Logs over the internet when the CA issues the certificate. The purpose of such logs of the certificate is to verify the authenticity of the received certificate. The client can verify the certificate received from the servers with the certificates logged in multiple (three in our case) Logs so that we know whether the certificate is valid or not. The logs contain the timestamp of issuance, log ID, SCT version for verification.

17. How is the application data being encrypted? Do the records containing application data include a separate MAC? Does Wireshark distinguish between the encrypted application data and the MAC?

Application data is encrypted using the key material derived using the Master Secret which is in inturn from PMS using Diffie-Hellman Key Exchange Protocol in an ephemeral fashion. (ECDHE was the agreed-upon key exchange protocol in the handshake protocol). As we saw earlier, the agreed-upon version for the TLS was TLS 1.3 which generates the encrypted data along with the MAC in a single process unlike in TLS 1.2 where the generation of MAC, using keys, and encryption of data was a different process (one after another). Here in TLS1.3, we have AEAD for encryption of the application data so it doesn't differentiate the encrypted application data with a separate MAC. So, the records containing the encrypted data don’t contain a separate MAC.

No, Wireshark doesn't differentiate between the encrypted application data and the MAC.s

18. Look at various keys logged in the file pointed to by the SSLKEYLOGFILE environment variable in your host OS and describe their usage. Also, comment on how they are derived from nonces and other parameters using HKDF. Which entity in your system does this job on the fly?

    There are multiple keys logged in the SSLKEYLOGFILE, as shown below:

    a. CLIENT_EARLY_TRAFFIC_SECRET: This key is used to encrypt the data even before the handshake protocol has finished negotiating any keys. This key is used in the 0-RTT protocol to send HTTP/ or any application data over to the server before the completion of the handshake protocol and any fixed keys/key materials are generated. So, This secret is used to derive the early traffic secret key that is used to send application data without waiting for the server to negotiate the session keys.

    b. CLIENT_RANDOM: This value is used as an input to PRF along with the MS to generate the key materials in (EC)DHE key exchange protocols. The ephemeral nature of this value prevents any sort of replay attacks.

    c. CLIENT_HANDSHAKE_TRAFFIC_SECRET: This secret is used to derive the handshake traffic secret key that is used to encrypt the handshake messages that are being sent to the server.
    d. SERVER_HANDSHAKE_TRAFFIC_SECRET: This secret is used to derive keys that are used to encrypt the handshake messages that are being sent from the server to the client.

    e. EXPORTER_SECRET: This secret is used to derive a key that is used by the application layer to encrypt data in the application layer itself for more security.

    f. CLIENT_TRAFFIC_SECRET_0: This secret is used to derive the key that the client uses to encrypt the application data sent from client to server.
    g. SERVER_TRAFFIC_SECRET_0: This secret is used to derive the key that the server uses to encrypt the application data sent from server to client.

    There are multiple steps in generating the actual keys in TLS 1.3. Initially with a pre-shared key; PSK, out of bound key, or a key that was shared earlier is used along with the SALT to generate the Early secret (of fixed length) using the HKDF-Extract Function.
    This early secret is used with label and message for the function Derive Secret to generate different secrets like binder keys, early traffic secrets,

exported master secrets.

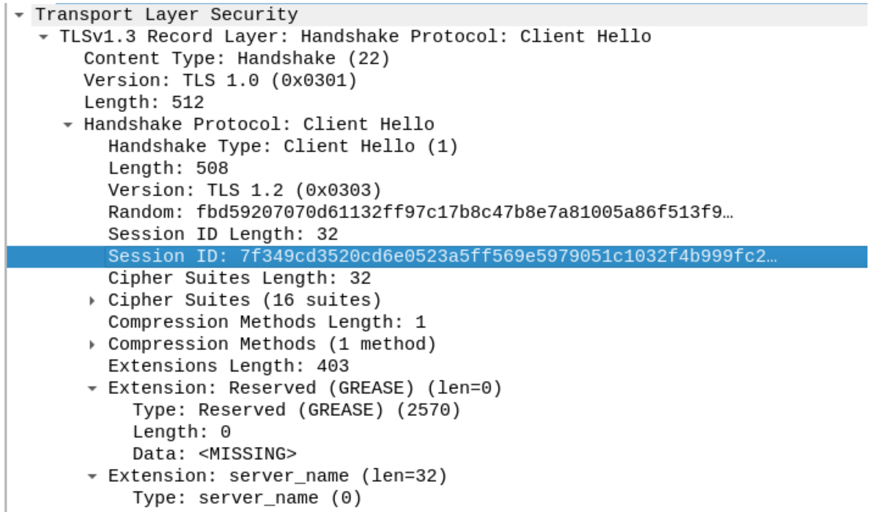
Now, this early secret is passed to the function Derive Secret with a "derived" message along with the parameters of EC-DHE to generate the Handshake Secret using the function HKDF-Extract. This handshake secret is passed with the derive secret function to generate multiple secrets like client handshake traffic secret or server handshake traffic secret.

So, Derive secret function here is internally calling the HKDF-Expand Label which is internally calling the HKDF-Expand Function to generate the specified keys. Now such keys can be used to encrypt various types of data.

The browser or the client entity on the communication along with the server both calculate the key materials using TLS1.3. Key scheduling. In the client-side, the browser querying the website does all of these operations.

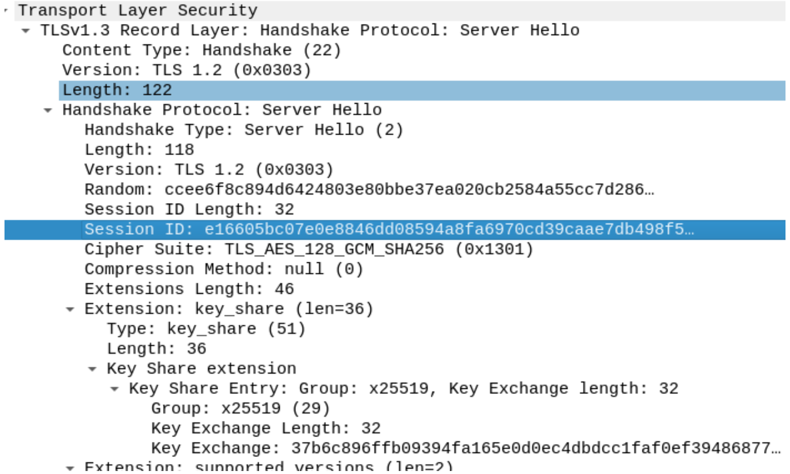
19. Do you see any support for session resumption in the trace? What do you find inside the session ticket, if it is used? Is it based on Session ID/Session ticket or PSK-based Session ticket? What role do the session IDs play in TLS 1.3?

Yes, In the client hello message, the client is providing session ID to resume the earlier sessions as shown in the figure below.



```
▾ Transport Layer Security
  ▾ TLSv1.3 Record Layer: Handshake Protocol: Client Hello
      Content Type: Handshake (22)
      Version: TLS 1.0 (0x0301)
      Length: 512
    ▾ Handshake Protocol: Client Hello
        Handshake Type: Client Hello (1)
        Length: 508
        Version: TLS 1.2 (0x0303)
        Random: fbd59207070d61132ff97c17b8c47b8e7a81005a86f513f9…
        Session ID Length: 32
        Session ID: 7f349cd3520cd6e0523a5ff569e5979051c1032f4b999fc2…
        Cipher Suites Length: 32
      ▸ Cipher Suites (16 suites)
        Compression Methods Length: 1
      ▸ Compression Methods (1 method)
        Extensions Length: 403
      ▾ Extension: Reserved (GREASE) (len=0)
          Type: Reserved (GREASE) (2570)
          Length: 0
          Data: <MISSING>
      ▾ Extension: server_name (len=32)
          Type: server_name (0)
```

To that particular Client Hello, the server is responding with a completely different session ID which indicates that the server is opting for a full handshake rather than session resumption as shown in the figure below:



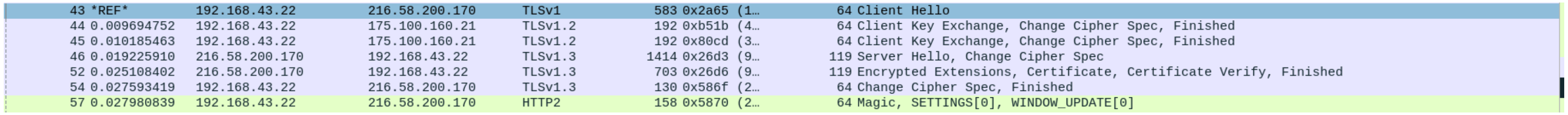
```
· Transport Layer Security
  ▾ TLSv1.3 Record Layer: Handshake Protocol: Server Hello
      Content Type: Handshake (22)
      Version: TLS 1.2 (0x0303)
      Length: 122
    ▾ Handshake Protocol: Server Hello
        Handshake Type: Server Hello (2)
        Length: 118
        Version: TLS 1.2 (0x0303)
        Random: ccee6f8c894d6424803e80bbe37ea020cb2584a55cc7d286…
        Session ID Length: 32
        Session ID: e16605bc07e0e8846dd08594a8fa6970cd39caae7db498f5…
        Cipher Suite: TLS_AES_128_GCM_SHA256 (0x1301)
        Compression Method: null (0)
        Extensions Length: 46
      ▾ Extension: key_share (len=36)
          Type: key_share (51)
          Length: 36
        ▾ Key Share extension
          ▾ Key Share Entry: Group: x25519, Key Exchange length: 32
              Group: x25519 (29)
              Key Exchange Length: 32
              Key Exchange: 37b6c896ffb09394fa165e0d0ec4dbdcc1faf0ef39486877…
      ▾ Extension: supported_versions (len=2)
```

This indicates that the session ID sent by the client is no longer valid to resume the earlier session.

Apart from the session ID, there are no packets indicating the availability or exchange of session tickets for session resumptions. The session resumptions in TLS1.3 are based on session tickets that are generated using the PSK or generated using resumption master secret. These tickets have their own lifetime hint indicating the availability of session tickets for usage. So in TLS1.3 session IDs are not used to resume the session, in fact IDs are not used at all.

20. How long does it take for TLS to establish a secure pipe? How much of it could be reduced when session resumption is used?

Since the client and server are using TLS1.2 to establish a secure pipe, TLS1.2 will take 2 RTTs to establish the connection and starts with HTTP/application layer encrypted requests. Also, the client can send multiple Client Hello Messages to ensure that the connection initiation has successfully reached the server. When Session resumption is used in TLS1.2 the RTT is reduced to 1 RTT before HTTP requests can be sent. Similarly in TLS1.3, we can have 0 RTT with session resumption meaning we can send HTTP requests along with the Client Hello messages.



```
 43 *REF*        192.168.43.22   216.58.200.170  TLSv1     583 0x2a65 (1…   64 Client Hello
 44 0.009694752  192.168.43.22   175.100.160.21  TLSv1.2   192 0xb51b (4…   64 Client Key Exchange, Change Cipher Spec, Finished
 45 0.010185463  192.168.43.22   175.100.160.21  TLSv1.2   192 0x80cd (3…   64 Client Key Exchange, Change Cipher Spec, Finished
 46 0.019225910  216.58.200.170  192.168.43.22   TLSv1.3  1414 0x26d3 (9…  119 Server Hello, Change Cipher Spec
 52 0.025108402  216.58.200.170  192.168.43.22   TLSv1.3   703 0x26d6 (9…  119 Encrypted Extensions, Certificate, Certificate Verify, Finished
 54 0.027593419  192.168.43.22   216.58.200.170  TLSv1.3   130 0x586f (2…   64 Change Cipher Spec, Finished
 57 0.027980839  192.168.43.22   216.58.200.170  HTTP2     158 0x5870 (2…   64 Magic, SETTINGS[0], WINDOW_UPDATE[0]
```

Now looking at the trace, the time taken from client hello to receiving a finished message from the server, the total time taken was 0.02759 seconds which is equivalent to 2 RTTs time, so the use of session resumption in TLS1.2 could reduce the handshake time to 0.02759/2 = 0.013579 seconds.

21. What is the duration of the HTTPS session, how many IP packets are exchanged in the browsing session (starting from the first TCP SYN packet till TCP FIN packet)?

    To calculate the duration of the HTTPS session, we have to subtract the time difference between the initial [SYN] packet received to the [FIN, ACK] packet received in the trace. But unfortunately, the trace did not contain the [FIN, ACK] packet and only contained the [SYN] packet which must be because I stopped capturing the packets before I close the website for the banking website. Since no close website request was sent to the server while the packet capture was still on, the [FIN] packets were not captured.

    Also, the total number of IP packets captured was 266 which is the total number of packets captured during the browsing session, excluding the packets that correspond to TCP FIN. Since the browsing session was only for the banking website, all the packets captured were for the same, so the entire captured packets count is the total number of IP packets captured.

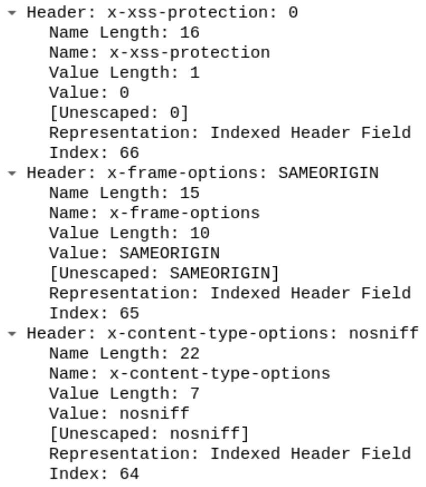
22. How many TLS connections are established?

    Looking at the captured trace and counting the Change Cipher Spec, Finished message from server to client, a total of 5 TLS connections seemed to be established.

23. How many HTTP request/response packets are exchanged in the browsing session? Identify the packet(s) that carried the response that included the Netbanking LOG-IN page of the bank. Do these response messages carry any security-related directives like XSS, same-origin, HSTS?

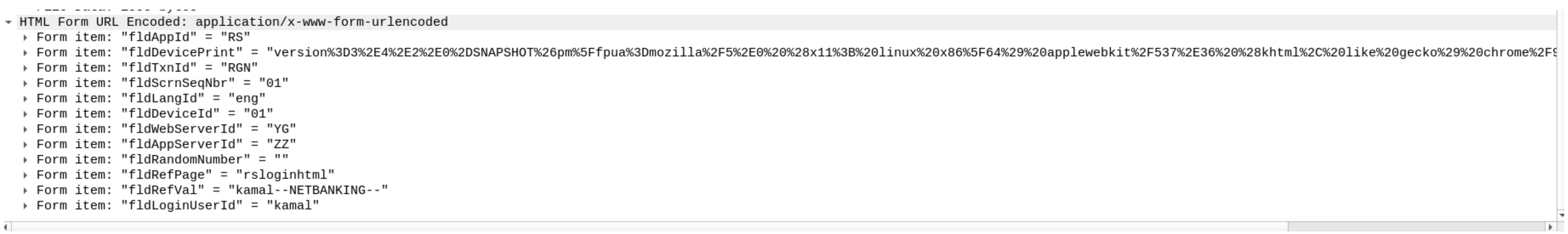
    A total of 6 HTTP request/response packets are exchanged in the browsing session.

    Yes, the response messages carry some security-related directives like XSS protection, x-frame-options: SAME ORIGIN as shown in the figure below:

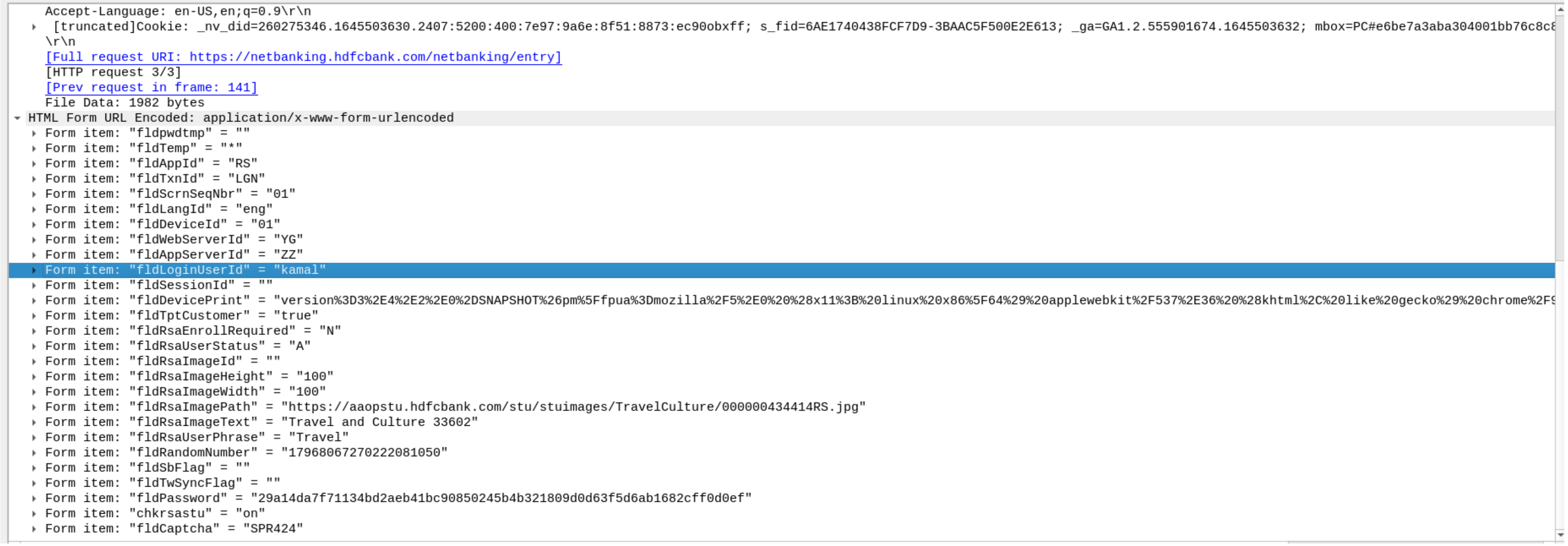


```
▾ Header: x-xss-protection: 0
    Name Length: 16
    Name: x-xss-protection
    Value Length: 1
    Value: 0
    [Unescaped: 0]
    Representation: Indexed Header Field
    Index: 66
▾ Header: x-frame-options: SAMEORIGIN
    Name Length: 15
    Name: x-frame-options
    Value Length: 10
    Value: SAMEORIGIN
    [Unescaped: SAMEORIGIN]
    Representation: Indexed Header Field
    Index: 65
▾ Header: x-content-type-options: nosniff
    Name Length: 22
    Name: x-content-type-options
    Value Length: 7
    Value: nosniff
    [Unescaped: nosniff]
    Representation: Indexed Header Field
    Index: 64
```

24. Identify the HTTP packet(s) that carried LOG-IN credentials supplied by you. Look at the raw bytes displayed in the Wireshark GUI and identify strings that carried your LOG-IN credentials. Are you able to find both user id and password in the raw packet capture?
    a. It’s important that you only keyed in some arbitrary user id and password as part of this assignment for more safety!



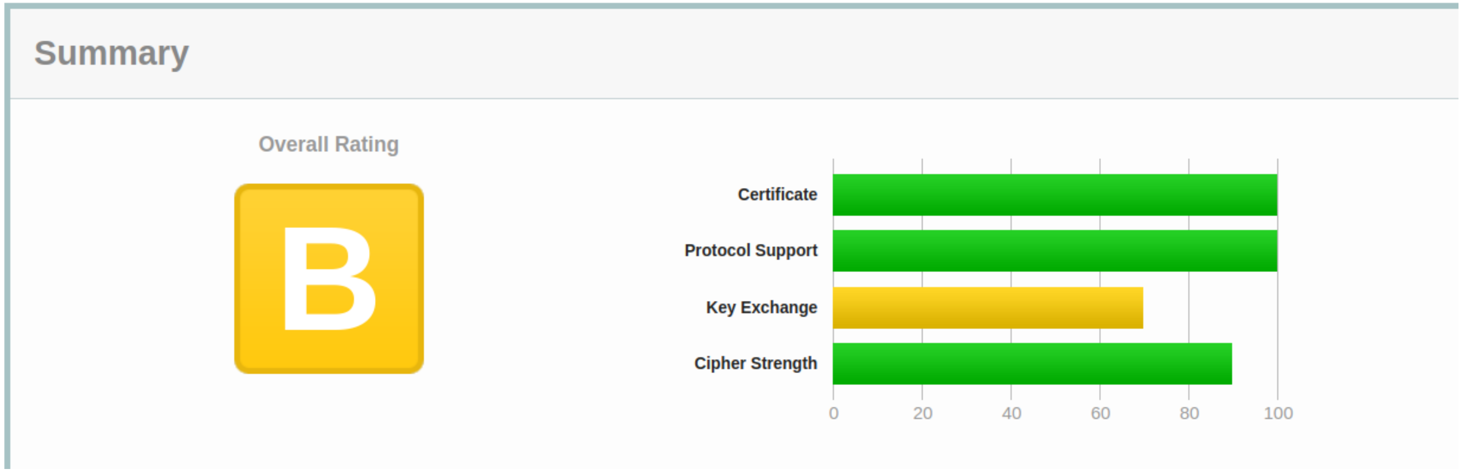
```
▾ HTML Form URL Encoded: application/x-www-form-urlencoded
  ▸ Form item: "fldAppId" = "RS"
  ▸ Form item: "fldDevicePrint" = "version%3D3%2E4%2E2%2E0%2DSNAPSHOT%26pm%5Ffpua%3Dmozilla%2F5%2E0%20%28x11%3B%20linux%20x86%5F64%29%20applewebkit%2F537%2E36%20%28khtml%2C%20like%20gecko%29%20chrome%2F
  ▸ Form item: "fldTxnId" = "RGN"
  ▸ Form item: "fldScrnSeqNbr" = "01"
  ▸ Form item: "fldLangId" = "eng"
  ▸ Form item: "fldDeviceId" = "01"
  ▸ Form item: "fldWebServerId" = "YG"
  ▸ Form item: "fldAppServerId" = "ZZ"
  ▸ Form item: "fldRandomNumber" = ""
  ▸ Form item: "fldRefPage" = "rsloginhtml"
  ▸ Form item: "fldRefVal" = "kamal--NETBANKING--"
  ▸ Form item: "fldLoginUserId" = "kamal"
```



```
    Accept-Language: en-US,en;q=0.9\r\n
  ▸  [truncated]Cookie: _nv_did=260275346.1645503630.2407:5200:400:7e97:9a6e:8f51:8873:ec90obxff; s_fid=6AE1740438FCF7D9-3BAAC5F500E2E613; _ga=GA1.2.555901674.1645503632; mbox=PC#e6be7a3aba304001bb76c8c
    \r\n
    [Full request URI: https://netbanking.hdfcbank.com/netbanking/entry]
    [HTTP request 3/3]
    [Prev request in frame: 141]
    File Data: 1982 bytes
▾ HTML Form URL Encoded: application/x-www-form-urlencoded
  ▸ Form item: "fldpwdtmp" = ""
  ▸ Form item: "fldTemp" = "*"
  ▸ Form item: "fldAppId" = "RS"
  ▸ Form item: "fldTxnId" = "LGN"
  ▸ Form item: "fldScrnSeqNbr" = "01"
  ▸ Form item: "fldLangId" = "eng"
  ▸ Form item: "fldDeviceId" = "01"
  ▸ Form item: "fldWebServerId" = "YG"
  ▸ Form item: "fldAppServerId" = "ZZ"
  ▸ Form item: "fldLoginUserId" = "kamal"
  ▸ Form item: "fldSessionId" = ""
  ▸ Form item: "fldDevicePrint" = "version%3D3%2E4%2E2%2E0%2DSNAPSHOT%26pm%5Ffpua%3Dmozilla%2F5%2E0%20%28x11%3B%20linux%20x86%5F64%29%20applewebkit%2F537%2E36%20%28khtml%2C%20like%20gecko%29%20chrome%2F
  ▸ Form item: "fldTptCustomer" = "true"
  ▸ Form item: "fldRsaEnrollRequired" = "N"
  ▸ Form item: "fldRsaUserStatus" = "A"
  ▸ Form item: "fldRsaImageId" = ""
  ▸ Form item: "fldRsaImageHeight" = "100"
  ▸ Form item: "fldRsaImageWidth" = "100"
  ▸ Form item: "fldRsaImagePath" = "https://aaopstu.hdfcbank.com/stu/stuimages/TravelCulture/000000434414RS.jpg"
  ▸ Form item: "fldRsaImageText" = "Travel and Culture 33602"
  ▸ Form item: "fldRsaUserPhrase" = "Travel"
  ▸ Form item: "fldRandomNumber" = "17968067270222081050"
  ▸ Form item: "fldSbFlag" = ""
  ▸ Form item: "fldTwSyncFlag" = ""
  ▸ Form item: "fldPassword" = "29a14da7f71134bd2aeb41bc90850245b4b321809d0d63f5d6ab1682cff0d0ef"
  ▸ Form item: "chkrsastu" = "on"
  ▸ Form item: "fldCaptcha" = "SPR424"
```

As you can see in the screenshot above, I can clearly identify the username I supplied in the login form as well as the password that seems to be hashed with some derivative function. There were two login forms (one additional with captcha) so there are two requests in the trace as shown above.

25. Generate an SSL report of the bank using SSL Server Test (Powered by Qualys SSL Labs) and summarize what security features are implemented by the bank's web server for improved online banking by its customers. Does the report flag any issues with the security of the bank?


Summary

Overall Rating

B

The signature algorithm that is being used is SHA256withRSA with 2048 bits for RSA. The server is not providing OCSP Staple with Server Hello which means the revocation status check is a little cumbersome. The bank has a properly signed certificate, that is signed by the intermediate CA. The server is only preferring TLS1.2 to establish a secure communication which brings a series of problems with itself, first and foremost being the weak cryptographic encryption algorithms along with no perfect forward secrecy as CBC mode is still being used to encrypt the messages and RSA is still being used to authenticate the server.

The server is also allowing secure client-initiated renegotiation. Any sorts of POODLE attacks, Heartbleed, ticket bleed, ROBOT are prevented.

Yes, the issue of no perfect forward secrecy is flagged along with the issue of the use of TLS1.2 cipher suites that involves CBC mode encryption. That is why the site is receiving only B grades.

26. Comment on and explain anything else that you found interesting in the trace.

Along with the handshake messages and the application data, there were a number of settings and update messages being exchanged between the communicating parties which contained encrypted application data. The exchange of settings and updates even after the completion of the handshake protocol was interesting. The selection of connection preface using MAGIC: PRI indicates the HTTP protocol to be used to query information from the server, was interesting.

A total of 5 TLS connections seemed to be established, which might potentially be because of other browsing sessions even though there were no other forms of browsing at the time of packet capture. So, 5 TLS connections were surprising and interesting,

Similarly, the use of TLS1.2 even though both the client and server support TLS1.3 just because the middleboxes don't seem pretty interesting to me because of the vast difficulty of implementations can be clearly seen.

PLAGIARISM STATEMENT

I certify that this assignment/report is my own work, based on my personal study and/or research and that I have acknowledged all material and sources used in its preparation, whether they be books, articles, reports, lecture notes, and any other kind of document, electronic or personal communication. I also certify that this assignment/report has not previously been submitted for assessment in any other course, except where specific permission has been granted from all course instructors involved, or at any other time in this course, and that I have not copied in part or whole or otherwise plagiarised the work of other students and/or persons. I pledge to uphold the principles of honesty and responsibility at CSE@IITH. In addition, I understand my responsibility to report honor violations by other students if I become aware of them.

Name: Kamal Shrestha
Date: Feb 27, 2022
Signature: K.S.