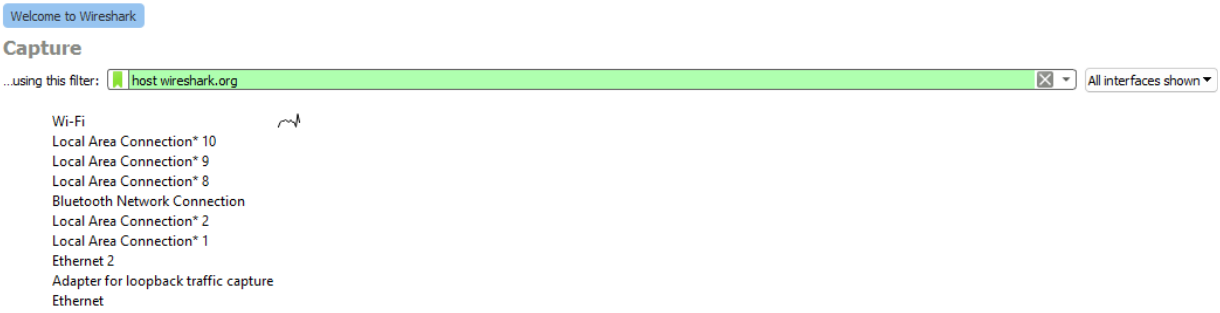
Observing tls cipher suites and comparing ecc and rsa digital signatures

Question 1:

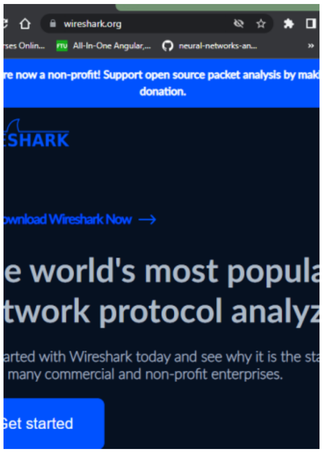


…using this filter: host wireshark.org

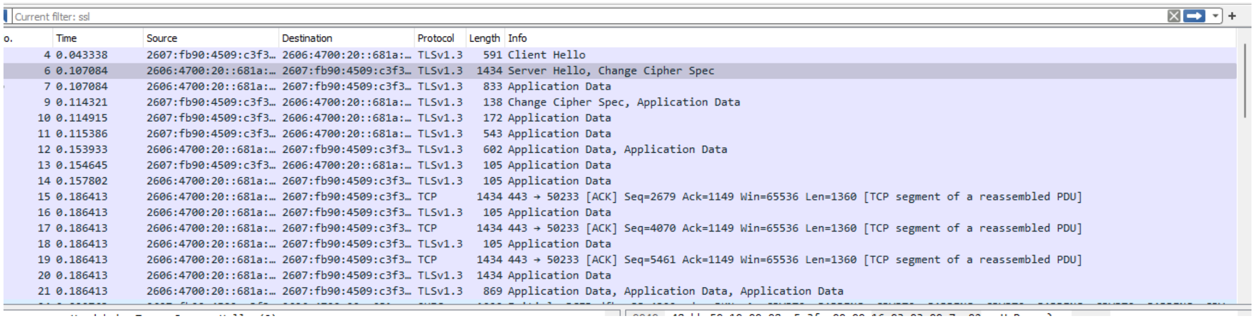
Wi-Fi
Local Area Connection* 10
Local Area Connection* 9
Local Area Connection* 8
Bluetooth Network Connection
Local Area Connection* 2
Local Area Connection* 1
Ethernet 2
Adapter for loopback traffic capture
Ethernet

a)

b)

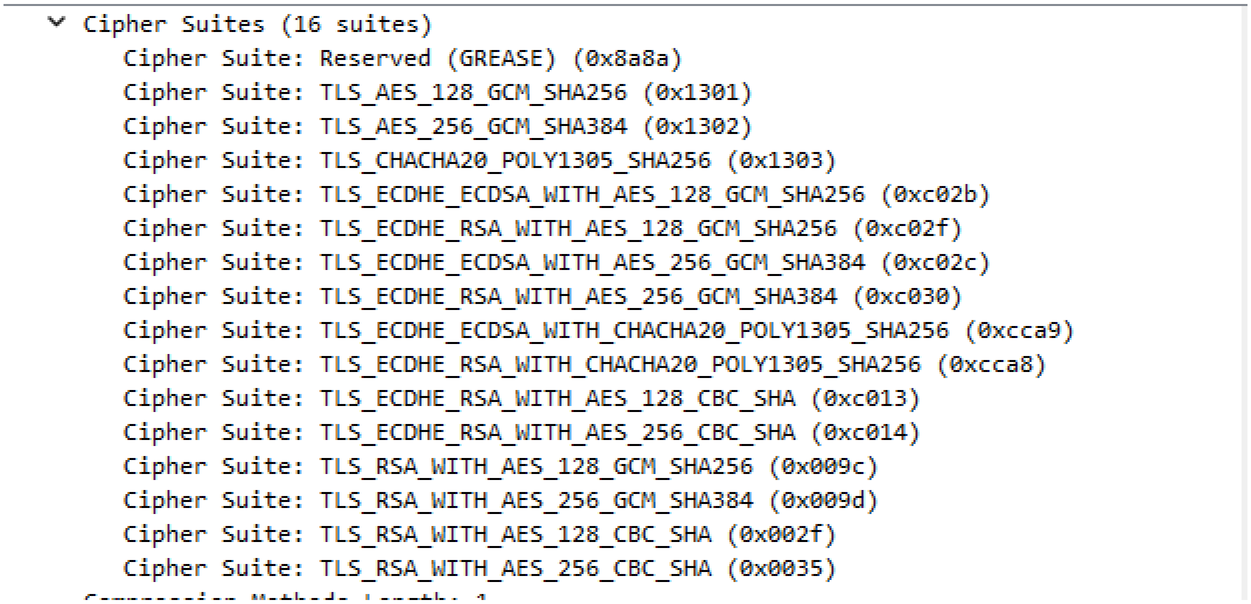

wireshark.org
ses Onlin...
All-In-One Angular,...
neural-networks-an...
re now a non-profit! Support open source packet analysis by mak
donation.
SHARK
wnload Wireshark Now →
e world's most popula
twork protocol analyz
arted with Wireshark today and see why it is the sta
many commercial and non-profit enterprises.
Get started

c)



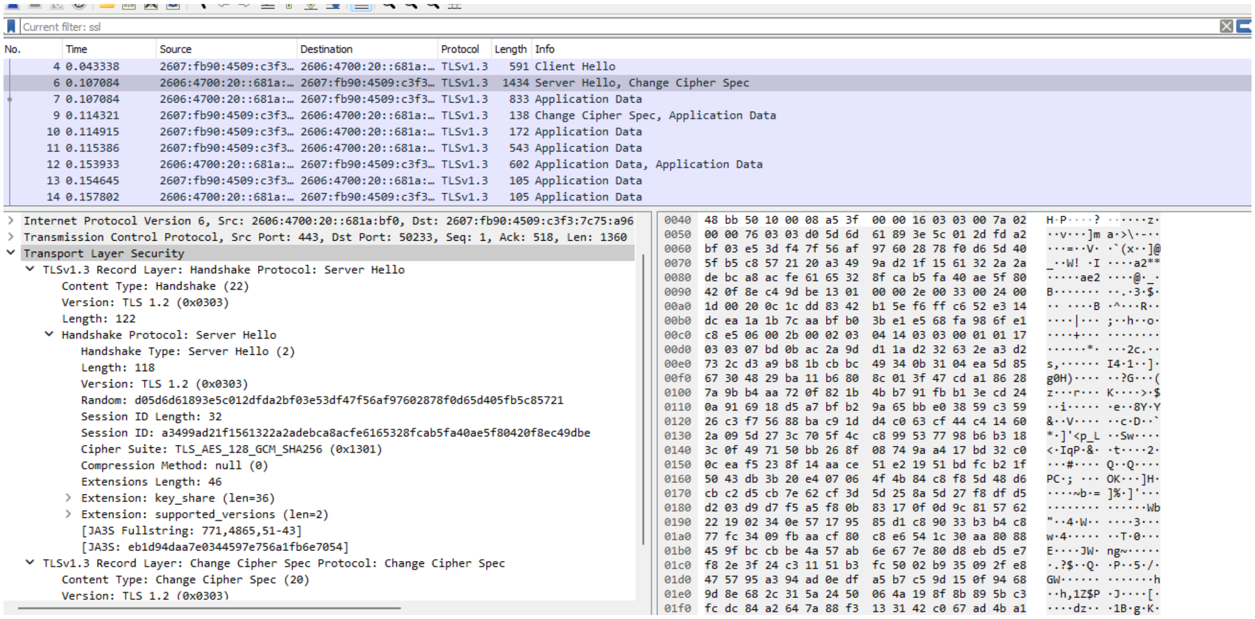
```
Current filter: ssl
o.   Time     Source              Destination         Protocol Length Info
   4 0.043338 2607:fb90:4509:c3f3… 2606:4700:20::681a:… TLSv1.3   591 Client Hello
   6 0.107084 2606:4700:20::681a:… 2607:fb90:4509:c3f3… TLSv1.3  1434 Server Hello, Change Cipher Spec
   7 0.107084 2606:4700:20::681a:… 2607:fb90:4509:c3f3… TLSv1.3   833 Application Data
   9 0.114321 2607:fb90:4509:c3f3… 2606:4700:20::681a:… TLSv1.3   138 Change Cipher Spec, Application Data
  10 0.114915 2607:fb90:4509:c3f3… 2606:4700:20::681a:… TLSv1.3   172 Application Data
  11 0.115386 2607:fb90:4509:c3f3… 2606:4700:20::681a:… TLSv1.3   543 Application Data
  12 0.153933 2606:4700:20::681a:… 2607:fb90:4509:c3f3… TLSv1.3   602 Application Data, Application Data
  13 0.154645 2607:fb90:4509:c3f3… 2606:4700:20::681a:… TLSv1.3   105 Application Data
  14 0.157802 2606:4700:20::681a:… 2607:fb90:4509:c3f3… TLSv1.3   105 Application Data
  15 0.186413 2606:4700:20::681a:… 2607:fb90:4509:c3f3… TCP      1434 443 → 50233 [ACK] Seq=2679 Ack=1149 Win=65536 Len=1360 [TCP segment of a reassembled PDU]
  16 0.186413 2606:4700:20::681a:… 2607:fb90:4509:c3f3… TLSv1.3   105 Application Data
  17 0.186413 2606:4700:20::681a:… 2607:fb90:4509:c3f3… TCP      1434 443 → 50233 [ACK] Seq=4070 Ack=1149 Win=65536 Len=1360 [TCP segment of a reassembled PDU]
  18 0.186413 2606:4700:20::681a:… 2607:fb90:4509:c3f3… TLSv1.3   105 Application Data
  19 0.186413 2606:4700:20::681a:… 2607:fb90:4509:c3f3… TCP      1434 443 → 50233 [ACK] Seq=5461 Ack=1149 Win=65536 Len=1360 [TCP segment of a reassembled PDU]
  20 0.186413 2606:4700:20::681a:… 2607:fb90:4509:c3f3… TLSv1.3  1434 Application Data
  21 0.186413 2606:4700:20::681a:… 2607:fb90:4509:c3f3… TLSv1.3   869 Application Data, Application Data, Application Data
```

d)



```
 ⌄ Cipher Suites (16 suites)
     Cipher Suite: Reserved (GREASE) (0x8a8a)
     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
```

e)



Current filter: ssl

```
No.  Time      Source              Destination         Protocol Length Info
   4 0.043338  2607:fb90:4509:c3f3… 2606:4700:20::681a:… TLSv1.3   591 Client Hello
   6 0.107084  2606:4700:20::681a:… 2607:fb90:4509:c3f3… TLSv1.3  1434 Server Hello, Change Cipher Spec
   7 0.107084  2606:4700:20::681a:… 2607:fb90:4509:c3f3… TLSv1.3   833 Application Data
   9 0.114321  2607:fb90:4509:c3f3… 2606:4700:20::681a:… TLSv1.3   138 Change Cipher Spec, Application Data
  10 0.114915  2607:fb90:4509:c3f3… 2606:4700:20::681a:… TLSv1.3   172 Application Data
  11 0.115386  2607:fb90:4509:c3f3… 2606:4700:20::681a:… TLSv1.3   543 Application Data
  12 0.153933  2606:4700:20::681a:… 2607:fb90:4509:c3f3… TLSv1.3   602 Application Data, Application Data
  13 0.154645  2607:fb90:4509:c3f3… 2606:4700:20::681a:… TLSv1.3   105 Application Data
  14 0.157802  2606:4700:20::681a:… 2607:fb90:4509:c3f3… TLSv1.3   105 Application Data
```

```
> Internet Protocol Version 6, Src: 2606:4700:20::681a:bf0, Dst: 2607:fb90:4509:c3f3:7c75:a96
> Transmission Control Protocol, Src Port: 443, Dst Port: 50233, Seq: 1, Ack: 518, Len: 1360
⌄ 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: d05d6d61893e5c012dfda2bf03e53df47f56af97602878f0d65d405fb5c85721
        Session ID Length: 32
        Session ID: a3499ad21f1561322a2adebca8acfe6165328fcab5fa40ae5f80420f8ec49dbe
        Cipher Suite: TLS_AES_128_GCM_SHA256 (0x1301)
        Compression Method: null (0)
        Extensions Length: 46
      > Extension: key_share (len=36)
      > Extension: supported_versions (len=2)
        [JA3S Fullstring: 771,4865,51-43]
        [JA3S: eb1d94daa7e0344597e756a1fb6e7054]
  ⌄ TLSv1.3 Record Layer: Change Cipher Spec Protocol: Change Cipher Spec
      Content Type: Change Cipher Spec (20)
      Version: TLS 1.2 (0x0303)
```

```
0040  48 bb 50 10 00 08 a5 3f  00 00 16 03 03 00 7a 02   H·P····? ······z·
0050  00 00 76 03 03 d0 5d 6d  61 89 3e 5c 01 2d fd a2   ··v···]m a·>\·-··
0060  bf 03 e5 3d f4 7f 56 af  97 60 28 78 f0 d6 5d 40   ···=··V· ·`(x··]@
0070  5f b5 c8 57 21 20 a3 49  9a d2 1f 15 61 32 2a 2a   _··W! ·I ····a2**
0080  de bc a8 ac fe 61 65 32  8f ca b5 fa 40 ae 5f 80   ·····ae2 ····@·_·
0090  42 0f 8e c4 9d be 13 01  00 00 2e 00 33 00 24 00   B······· ··.·3·$·
00a0  1d 00 20 0c 1c dd 83 42  b1 5e f6 ff c6 52 e3 14   ·· ····B ·^···R··
00b0  dc ea 1a 1b 7c aa bf b0  3b e1 e5 68 fa 98 6f e1   ····|··· ;··h··o·
00c0  c8 e5 06 00 2b 00 02 03  04 14 03 03 00 01 01 17   ····+··· ········
00d0  03 03 07 bd 0b ac 2a 9d  d1 1a d2 32 63 2e a3 d2   ······*· ···2c.··
00e0  73 2c d3 a9 b8 1b cb bc  49 34 0b 31 04 ea 5d 85   s,······ I4·1··]·
00f0  67 30 48 29 ba 11 b6 80  8c 01 3f 47 cd a1 86 28   g0H)···· ··?G···(
0100  7a 9b b4 aa 72 0f 82 1b  4b b7 91 fb b1 3e cd 24   z···r··· K····>·$
0110  0a 91 69 18 d5 a7 bf b2  9a 65 bb e0 38 59 c3 59   ··i····· ·e··8Y·Y
0120  26 c3 f7 56 88 ba c9 1d  d4 c0 63 cf 44 c4 14 60   &··V···· ··c·D··`
0130  2a 09 5d 27 3c 70 5f 4c  c8 99 53 77 98 b6 b3 18   *·]'<p_L ··Sw····
0140  3c 0f 49 71 50 bb 26 8f  08 74 9a a4 17 bd 32 c0   <·IqP·&· ·t····2·
0150  0c ea f5 23 8f 14 aa ce  51 e2 19 51 bd fc b2 1f   ···#···· Q··Q····
0160  50 43 db 3b 20 e4 07 06  4f 4b 84 c8 f8 5d 48 d6   PC·; ··· OK···]H·
0170  cb c2 d5 cb 7e 62 cf 3d  5d 25 8a 5d 27 f8 df d5   ····~b·= ]%·]'···
0180  d2 03 d9 d7 f5 a5 f8 0b  83 17 0f 0d 9c 81 57 62   ········ ······Wb
0190  22 19 02 34 0e 57 17 95  85 d1 c8 90 33 b3 b4 c8   "··4·W·· ····3···
01a0  77 fc 34 09 fb aa cf 80  c8 e6 54 1c 30 aa 80 88   w·4····· ··T·0···
01b0  45 9f bc cb be 4a 57 ab  6e 67 7e 80 d8 eb d5 e7   E····JW· ng~·····
01c0  f8 2e 3f 24 c3 11 51 b3  fc 50 02 b9 35 09 2f e8   ·.?$··Q· ·P··5·/·
01d0  47 57 95 a3 94 ad 0e df  a5 b7 c5 9d 15 0f 94 68   GW······ ·······h
01e0  9d 8e 68 2c 31 5a 24 50  06 4a 19 8f 8b 89 5b c3   ··h,1Z$P ·J····[·
01f0  fc dc 84 a2 64 7a 88 f3  13 31 42 c0 67 ad 4b a1   ····dz·· ·1B·g·K·
```

The Cipher Suite TLS_AES_128_GCM_SHA256 (0x1301) is used in the Transport Layer Security (TLS) protocol to provide encryption, message authentication, and integrity protection for data transmitted over the internet.

This suite consists of three algorithms:

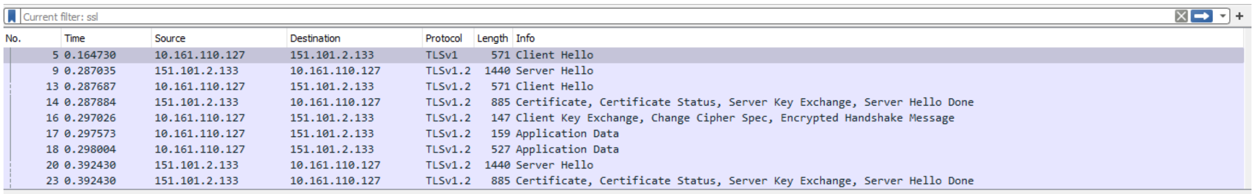
1. AES-128 in Galois/Counter Mode (GCM) for symmetric encryption. AES is a widely used symmetric-key encryption algorithm that provides strong encryption and is considered secure. GCM is a mode of operation for block ciphers that provides both confidentiality and authentication.

2. SHA-256 for message authentication. SHA-256 is a hashing algorithm that generates a fixed-size output of 256 bits, which is used to verify the integrity of the data.

3. The TLS protocol also provides a key exchange algorithm to establish a shared secret key between the client and the server. The key exchange algorithm used in this cipher suite is not specified, and it could be any of the algorithms supported by TLS, such as Diffie-Hellman or Elliptic Curve Cryptography.

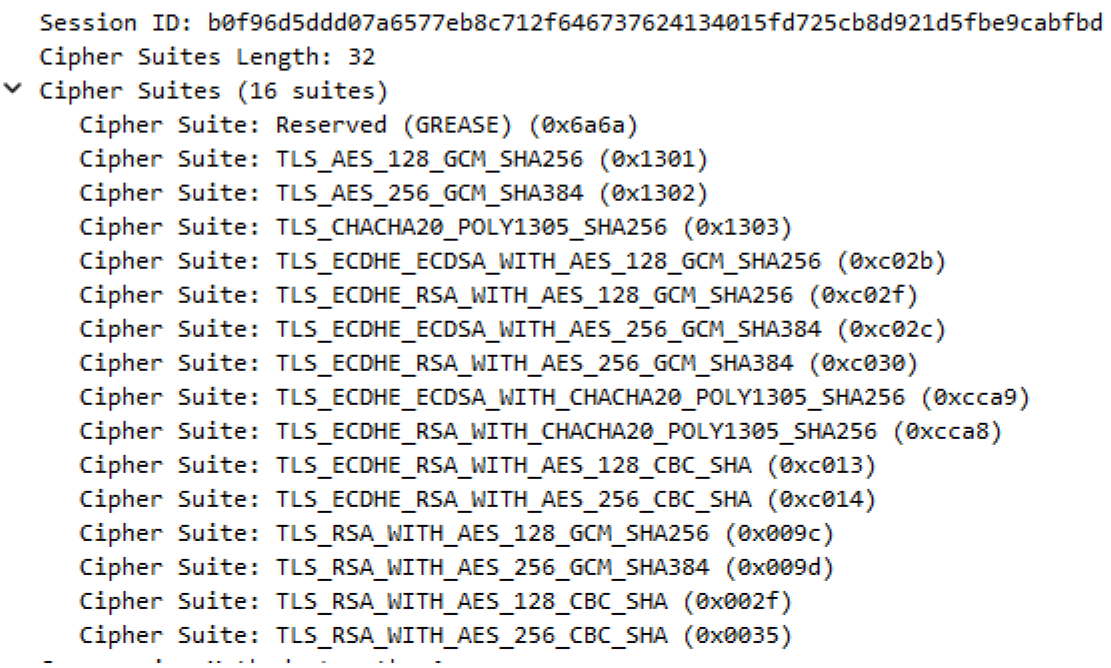
In summary, this cipher suite provides strong encryption and authentication for data transmitted over the internet, making it a popular choice for securing web traffic. The use of GCM mode of operation with AES-128 provides both confidentiality and authentication in a single operation, which makes it efficient and reduces overhead. The use of SHA-256 for message authentication ensures that the data has not been tampered with during transmission.

f)

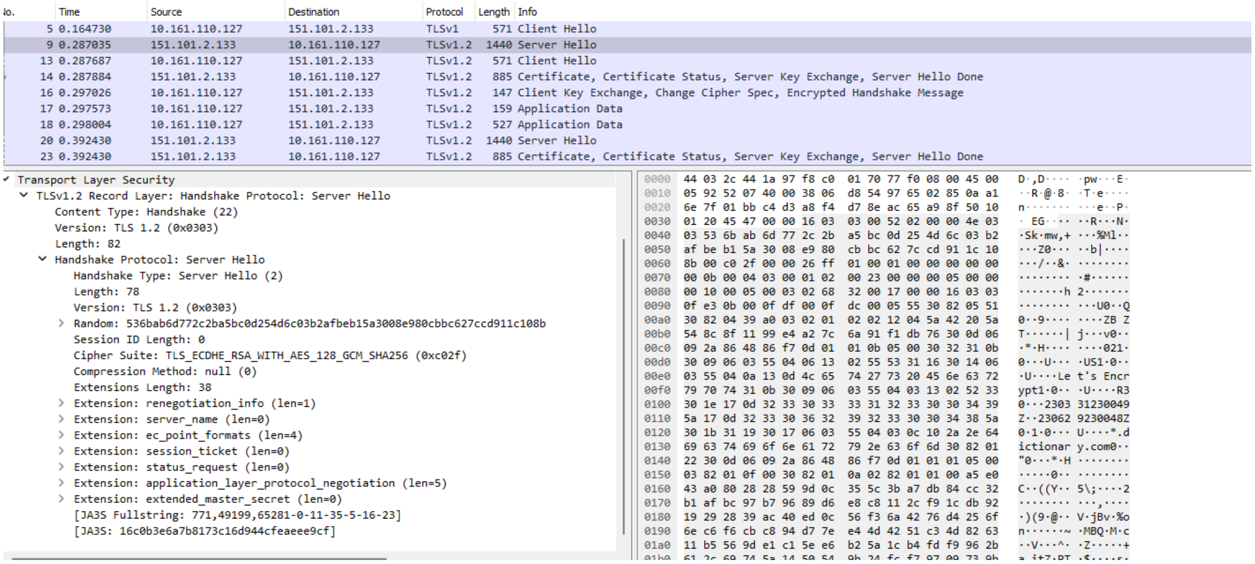


Current filter: ssl

No.	Time	Source	Destination	Protocol	Length	Info
5	0.164730	10.161.110.127	151.101.2.133	TLSv1	571	Client Hello
9	0.287035	151.101.2.133	10.161.110.127	TLSv1.2	1440	Server Hello
13	0.287687	10.161.110.127	151.101.2.133	TLSv1.2	571	Client Hello
14	0.287884	151.101.2.133	10.161.110.127	TLSv1.2	885	Certificate, Certificate Status, Server Key Exchange, Server Hello Done
16	0.297026	10.161.110.127	151.101.2.133	TLSv1.2	147	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
17	0.297573	10.161.110.127	151.101.2.133	TLSv1.2	159	Application Data
18	0.298004	10.161.110.127	151.101.2.133	TLSv1.2	527	Application Data
20	0.392430	151.101.2.133	10.161.110.127	TLSv1.2	1440	Server Hello
23	0.392430	151.101.2.133	10.161.110.127	TLSv1.2	885	Certificate, Certificate Status, Server Key Exchange, Server Hello Done



```
Session ID: b0f96d5ddd07a6577eb8c712f646737624134015fd725cb8d921d5fbe9cabfbd
Cipher Suites Length: 32
⌄ Cipher Suites (16 suites)
    Cipher Suite: Reserved (GREASE) (0x6a6a)
    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)
```



No.	Time	Source	Destination	Protocol	Length	Info
5	0.164730	10.161.110.127	151.101.2.133	TLSv1	571	Client Hello
9	0.287035	151.101.2.133	10.161.110.127	TLSv1.2	1440	Server Hello
13	0.287687	10.161.110.127	151.101.2.133	TLSv1.2	571	Client Hello
14	0.287884	151.101.2.133	10.161.110.127	TLSv1.2	885	Certificate, Certificate Status, Server Key Exchange, Server Hello Done
16	0.297026	10.161.110.127	151.101.2.133	TLSv1.2	147	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
17	0.297573	10.161.110.127	151.101.2.133	TLSv1.2	159	Application Data
18	0.298004	10.161.110.127	151.101.2.133	TLSv1.2	527	Application Data
20	0.392430	151.101.2.133	10.161.110.127	TLSv1.2	1440	Server Hello
23	0.392430	151.101.2.133	10.161.110.127	TLSv1.2	885	Certificate, Certificate Status, Server Key Exchange, Server Hello Done

```
✓ Transport Layer Security
  ⌄ TLSv1.2 Record Layer: Handshake Protocol: Server Hello
      Content Type: Handshake (22)
      Version: TLS 1.2 (0x0303)
      Length: 82
    ⌄ Handshake Protocol: Server Hello
        Handshake Type: Server Hello (2)
        Length: 78
        Version: TLS 1.2 (0x0303)
      > Random: 536bab6d772c2ba5bc0d254d6c03b2afbeb15a3008e980cbbc627ccd911c108b
        Session ID Length: 0
        Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)
        Compression Method: null (0)
        Extensions Length: 38
      > Extension: renegotiation_info (len=1)
      > Extension: server_name (len=0)
      > Extension: ec_point_formats (len=4)
      > Extension: session_ticket (len=0)
      > Extension: status_request (len=0)
      > Extension: application_layer_protocol_negotiation (len=5)
      > Extension: extended_master_secret (len=0)
        [JA3S Fullstring: 771,49199,65281-0-11-35-5-16-23]
        [JA3S: 16c0b3e6a7b8173c16d944cfeaeee9cf]
```

```
0000  44 03 2c 44 1a 97 f8 c0  01 70 77 f0 08 00 45 00   D·,D····  ·pw···E·
0010  05 92 52 07 40 00 38 06  d8 54 97 65 02 85 0a a1   ··R·@·8·  ·T·e····
0020  6e 7f 01 bb c4 d3 a8 f4  d7 8e ac 65 a9 8f 50 10   n·······  ···e··P·
0030  01 20 45 47 00 00 16 03  03 00 52 02 00 00 4e 03   · EG····  ··R···N·
0040  03 53 6b ab 6d 77 2c 2b  a5 bc 0d 25 4d 6c 03 b2   ·Sk·mw,+  ···%Ml··
0050  af be b1 5a 30 08 e9 80  cb bc 62 7c cd 91 1c 10   ···Z0···  ··b|····
0060  8b 00 c0 2f 00 00 26 ff  01 00 01 00 00 00 00 00   ···/··&·  ········
0070  00 0b 00 04 03 00 01 02  00 23 00 00 00 05 00 00   ········  ·#······
0080  00 10 00 05 00 03 02 68  32 00 17 00 00 16 03 03   ·······h  2·······
0090  0f e3 0b 00 0f df 00 0f  dc 00 05 55 30 82 05 51   ········  ···U0··Q
00a0  30 82 04 39 a0 03 02 01  02 02 12 04 5a 42 20 5a   0··9····  ····ZB Z
00b0  54 8c 8f 11 99 e4 a2 7c  6a 91 f1 db 76 30 0d 06   T······|  j···v0··
00c0  09 2a 86 48 86 f7 0d 01  01 0b 05 00 30 32 31 0b   ·*·H····  ····021·
00d0  30 09 06 03 55 04 06 13  02 55 53 31 16 30 14 06   0···U···  ·US1·0··
00e0  03 55 04 0a 13 0d 4c 65  74 27 73 20 45 6e 63 72   ·U····Le  t's Encr
00f0  79 70 74 31 0b 30 09 06  03 55 04 03 13 02 52 33   ypt1·0··  ·U····R3
0100  30 1e 17 0d 32 33 30 33  33 31 32 33 30 30 34 39   0···2303  31230049
0110  5a 17 0d 32 33 30 36 32  39 32 33 30 30 34 38 5a   Z··23062  9230048Z
0120  30 1b 31 19 30 17 06 03  55 04 03 0c 10 2a 2e 64   0·1·0···  U····*.d
0130  69 63 74 69 6f 6e 61 72  79 2e 63 6f 6d 30 82 01   ictionar  y.com0··
0140  22 30 0d 06 09 2a 86 48  86 f7 0d 01 01 01 05 00   "0···*·H  ········
0150  03 82 01 0f 00 30 82 01  0a 02 82 01 01 00 a5 e0   ·····0··  ········
0160  43 a0 80 28 28 59 9d 0c  35 5c 3b a7 db 84 cc 32   C··((Y··  5\;····2
0170  b1 af bc 97 b7 96 89 d6  e8 c8 11 2c f9 1c db 92   ········  ···,····
0180  19 29 28 39 ac 40 ed 0c  56 f3 6a 42 76 d4 25 6f   ·)(9·@··  V·jBv·%o
0190  6e c6 f6 cb c8 94 d7 7e  e4 4d 42 51 c3 4d 82 63   n······~  ·MBQ·M·c
01a0  11 b5 56 9d e1 c1 5e e6  b2 5a 1c b4 fd f9 96 2b   ··V···^·  ·Z·····+
```

The Cipher Suite TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f) is used in the Transport Layer Security (TLS) protocol to provide secure communication over the internet.

This cipher suite consists of four algorithms:

1. ECDHE (Elliptic Curve Diffie-Hellman Ephemeral) for key exchange. ECDHE is a key exchange algorithm that provides forward secrecy by generating a new key for each session, making it more secure than other key exchange algorithms.

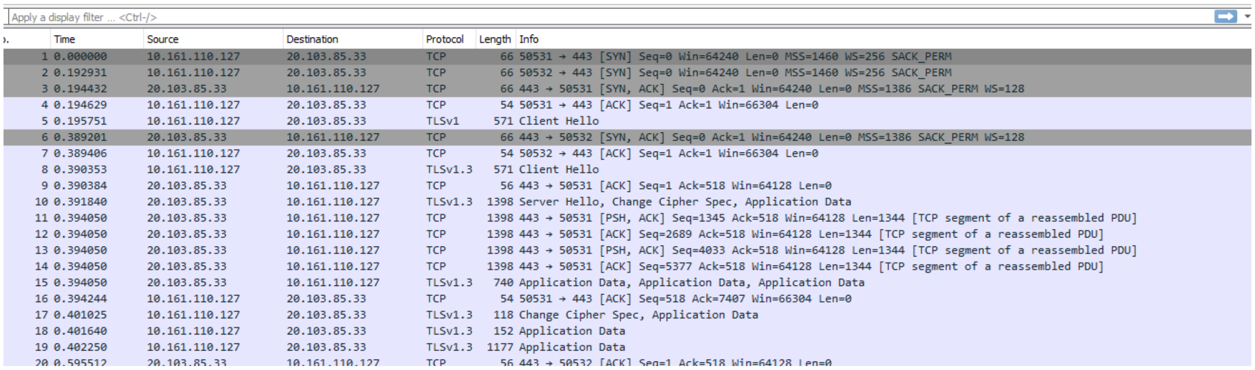
2. RSA for server authentication. RSA is a widely used public-key encryption algorithm that provides a secure method for the server to authenticate itself to the client.

3. AES-128 in Galois/Counter Mode (GCM) for symmetric encryption. AES is a widely used symmetric-key encryption algorithm that provides strong encryption and is considered secure. GCM is a mode of operation for block ciphers that provides both confidentiality and authentication.

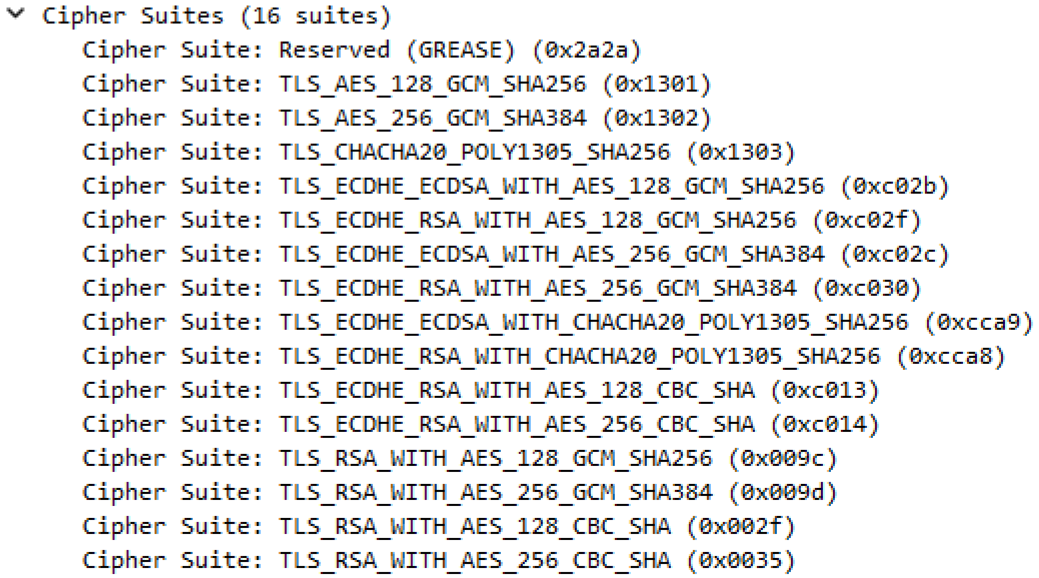
4. SHA-256 for message authentication. SHA-256 is a hashing algorithm that generates a fixed-size output of 256 bits, which is used to verify the integrity of the data.

In summary, this cipher suite provides strong encryption, authentication, and key exchange for data transmitted over the internet. The use of ECDHE for key exchange provides forward secrecy, which ensures that even if the long-term private key of the server is compromised, the confidentiality of past sessions is still maintained. The use of RSA for server authentication ensures that the client is communicating with the intended server. The use of AES-128 in GCM mode provides both confidentiality and authentication in a single operation, which reduces overhead and improves performance. Finally, the use of SHA-256 for message authentication ensures that the data has not been tampered with during transmission.

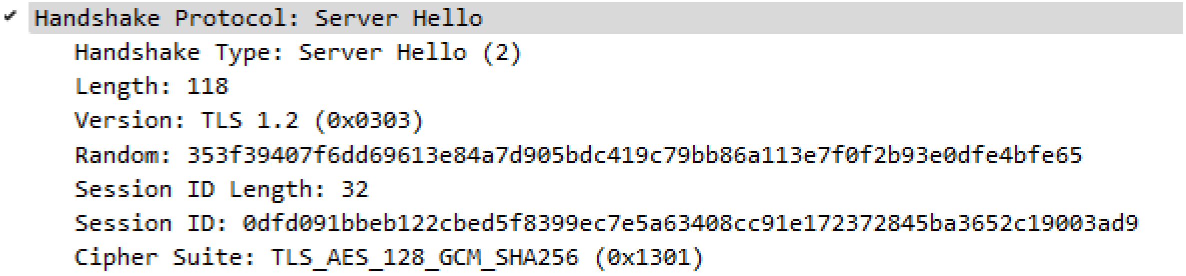
2nd example:



```
.    Time     Source          Destination     Protocol Length Info
   1 0.000000 10.161.110.127  20.103.85.33    TCP        66 50531 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
   2 0.192931 10.161.110.127  20.103.85.33    TCP        66 50532 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
   3 0.194432 20.103.85.33    10.161.110.127  TCP        66 443 → 50531 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1386 SACK_PERM WS=128
   4 0.194629 10.161.110.127  20.103.85.33    TCP        54 50531 → 443 [ACK] Seq=1 Ack=1 Win=66304 Len=0
   5 0.195751 10.161.110.127  20.103.85.33    TLSv1     571 Client Hello
   6 0.389201 20.103.85.33    10.161.110.127  TCP        66 443 → 50532 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1386 SACK_PERM WS=128
   7 0.389406 10.161.110.127  20.103.85.33    TCP        54 50532 → 443 [ACK] Seq=1 Ack=1 Win=66304 Len=0
   8 0.390353 10.161.110.127  20.103.85.33    TLSv1.3   571 Client Hello
   9 0.390384 20.103.85.33    10.161.110.127  TCP        56 443 → 50531 [ACK] Seq=1 Ack=518 Win=64128 Len=0
  10 0.391840 20.103.85.33    10.161.110.127  TLSv1.3  1398 Server Hello, Change Cipher Spec, Application Data
  11 0.394050 20.103.85.33    10.161.110.127  TCP      1398 443 → 50531 [PSH, ACK] Seq=1345 Ack=518 Win=64128 Len=1344 [TCP segment of a reassembled PDU]
  12 0.394050 20.103.85.33    10.161.110.127  TCP      1398 443 → 50531 [ACK] Seq=2689 Ack=518 Win=64128 Len=1344 [TCP segment of a reassembled PDU]
  13 0.394050 20.103.85.33    10.161.110.127  TCP      1398 443 → 50531 [PSH, ACK] Seq=4033 Ack=518 Win=64128 Len=1344 [TCP segment of a reassembled PDU]
  14 0.394050 20.103.85.33    10.161.110.127  TCP      1398 443 → 50531 [ACK] Seq=5377 Ack=518 Win=64128 Len=1344 [TCP segment of a reassembled PDU]
  15 0.394050 20.103.85.33    10.161.110.127  TLSv1.3   740 Application Data, Application Data, Application Data
  16 0.394244 10.161.110.127  20.103.85.33    TCP        54 50531 → 443 [ACK] Seq=518 Ack=7407 Win=66304 Len=0
  17 0.401025 10.161.110.127  20.103.85.33    TLSv1.3   118 Change Cipher Spec, Application Data
  18 0.401640 10.161.110.127  20.103.85.33    TLSv1.3   152 Application Data
  19 0.402250 10.161.110.127  20.103.85.33    TLSv1.3  1177 Application Data
  20 0.595512 20.103.85.33    10.161.110.127  TCP        56 443 → 50532 [ACK] Seq=1 Ack=518 Win=64128 Len=0
```



```
˅ Cipher Suites (16 suites)
    Cipher Suite: Reserved (GREASE) (0x2a2a)
    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)
```



```
✓ Handshake Protocol: Server Hello
    Handshake Type: Server Hello (2)
    Length: 118
    Version: TLS 1.2 (0x0303)
    Random: 353f39407f6dd69613e84a7d905bdc419c79bb86a113e7f0f2b93e0dfe4bfe65
    Session ID Length: 32
    Session ID: 0dfd091bbeb122cbed5f8399ec7e5a63408cc91e172372845ba3652c19003ad9
    Cipher Suite: TLS_AES_128_GCM_SHA256 (0x1301)
```

The Cipher Suite TLS_AES_128_GCM_SHA256 (0x1301) is a widely used cipher suite in the Transport Layer Security (TLS) protocol. It provides encryption, message authentication, and integrity protection for data transmitted over the internet.

This cipher suite consists of three algorithms:

1. AES-128 in Galois/Counter Mode (GCM) for symmetric encryption. AES is a widely used symmetric-key encryption algorithm that provides strong encryption and is considered secure. GCM is a mode of operation for block ciphers that provides both confidentiality and authentication.

2. SHA-256 for message authentication. SHA-256 is a hashing algorithm that generates a fixed-size output of 256 bits, which is used to verify the integrity of the data.

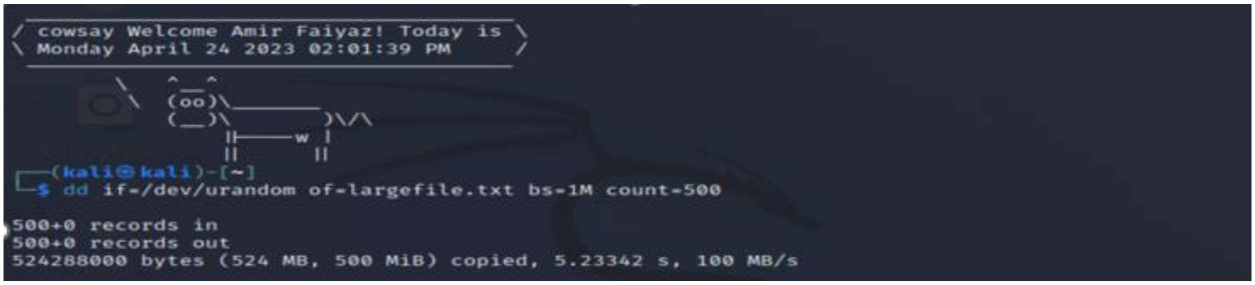
3. The TLS protocol also provides a key exchange algorithm to establish a shared secret key between the client and the server. The key exchange algorithm used in this cipher suite is not specified, and it could be any of the algorithms supported by TLS, such as Diffie-Hellman or Elliptic Curve Cryptography.

This cipher suite offers strong encryption and authentication for web traffic using AES-128 in GCM mode for confidentiality and authentication, and SHA-256 for message authentication. It provides flexibility by supporting various key exchange algorithms.

g)

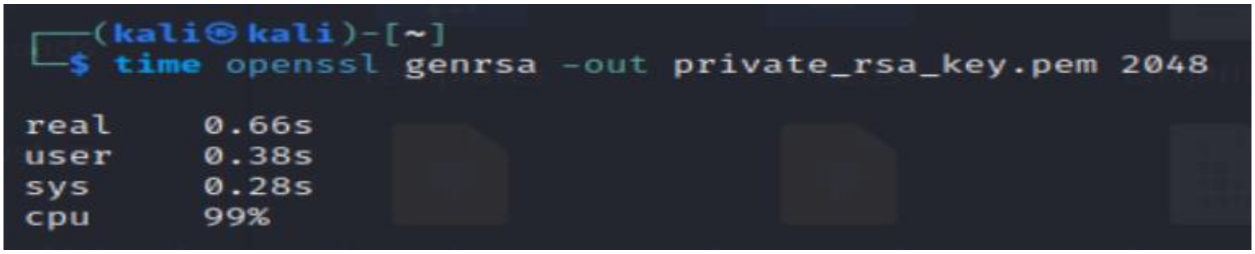
When selecting a cipher suite for a server, the considerations typically include ensuring that the selected suite provides strong encryption, authentication, and key exchange, while also being compatible with the clients that will be connecting to the server. Other factors may include performance, the level of security required for the data being transmitted, and any regulatory or industry standards that must be met. It's important to choose a cipher suite that balances these considerations appropriately and stays up-to-date with the latest security recommendations.

Question 2)



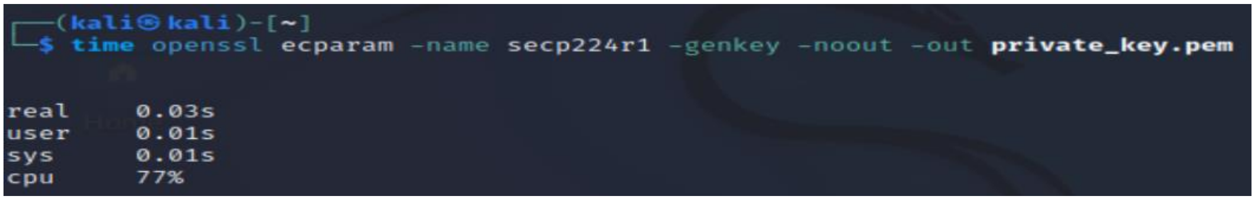
```
 ____________________________________
/ cowsay Welcome Amir Faiyaz! Today is \
\ Monday April 24 2023 02:01:39 PM     /
 ------------------------------------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||
┌──(kali㉿kali)-[~]
└─$ dd if=/dev/urandom of=largefile.txt bs=1M count=500

500+0 records in
500+0 records out
524288000 bytes (524 MB, 500 MiB) copied, 5.23342 s, 100 MB/s
```

c) generate rsa key pair 2048



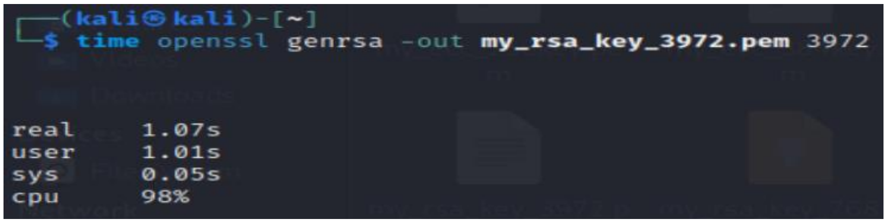
```
┌──(kali㉿kali)-[~]
└─$ time openssl genrsa -out private_rsa_key.pem 2048

real    0.66s
user    0.38s
sys     0.28s
cpu     99%
```

d) ecc key 224



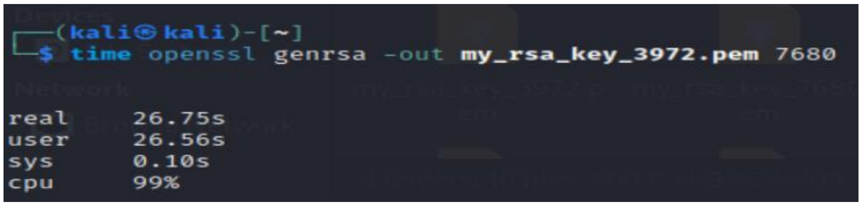
```
┌──(kali㉿kali)-[~]
└─$ time openssl ecparam -name secp224r1 -genkey -noout -out private_key.pem

real    0.03s
user    0.01s
sys     0.01s
cpu     77%
```

Rsa 3972 key



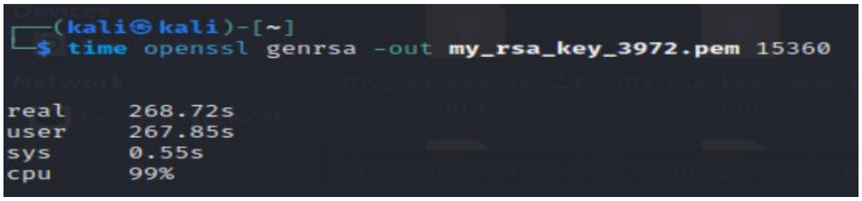
```
┌──(kali㉿kali)-[~]
└─$ time openssl genrsa -out my_rsa_key_3972.pem 3972

real    1.07s
user    1.01s
sys     0.05s
cpu     98%
```

Rsa 7680 key



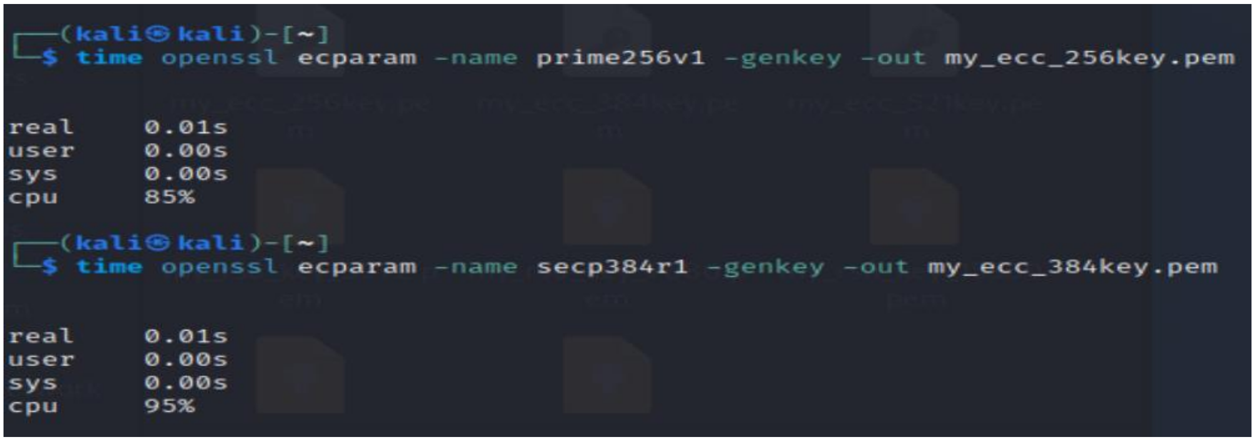
```
┌──(kali㉿kali)-[~]
└─$ time openssl genrsa -out my_rsa_key_3972.pem 7680

real    26.75s
user    26.56s
sys     0.10s
cpu     99%
```

Rsa 15360 key

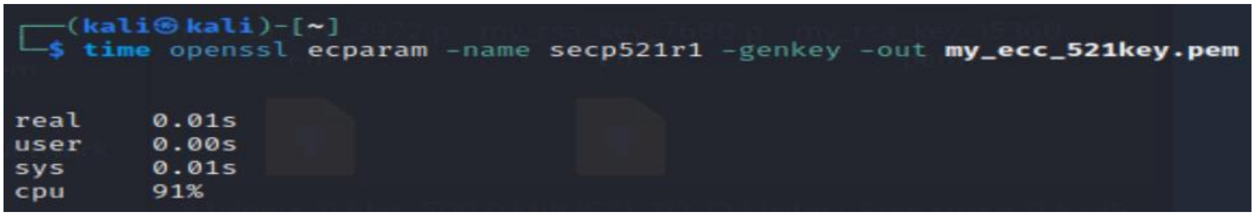


```
┌──(kali㉿kali)-[~]
└─$ time openssl genrsa -out my_rsa_key_3972.pem 15360

real    268.72s
user    267.85s
sys     0.55s
cpu     99%
```

Ecc 256 , 384 and 521 key:

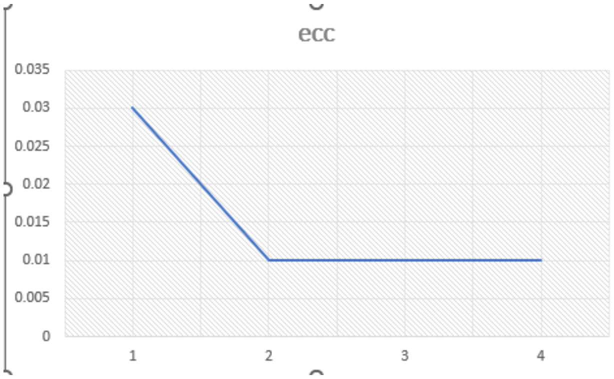


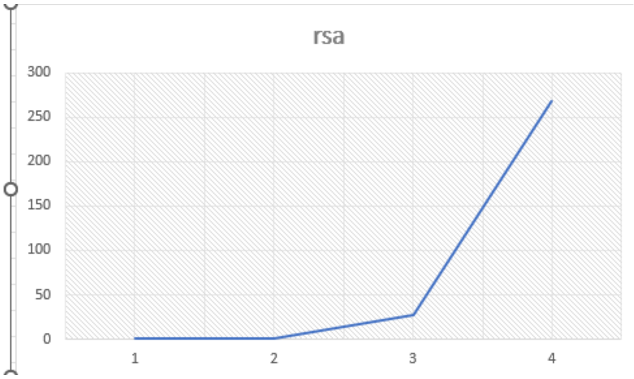
```
┌──(kali㉿kali)-[~]
└─$ time openssl ecparam -name prime256v1 -genkey -out my_ecc_256key.pem

real    0.01s
user    0.00s
sys     0.00s
cpu     85%

┌──(kali㉿kali)-[~]
└─$ time openssl ecparam -name secp384r1 -genkey -out my_ecc_384key.pem

real    0.01s
user    0.00s
sys     0.00s
cpu     95%
```



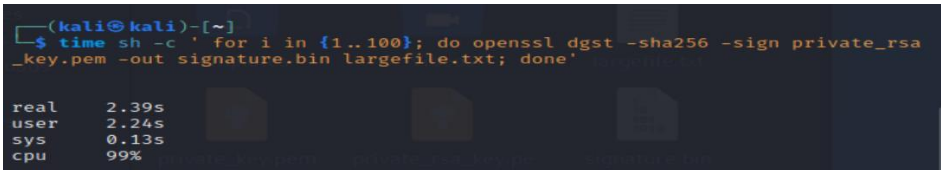
```
┌──(kali㉿kali)-[~]
└─$ time openssl ecparam -name secp521r1 -genkey -out my_ecc_521key.pem

real    0.01s
user    0.00s
sys     0.01s
cpu     91%
```

e)



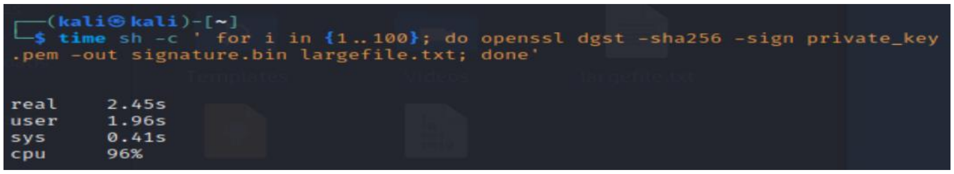


f)



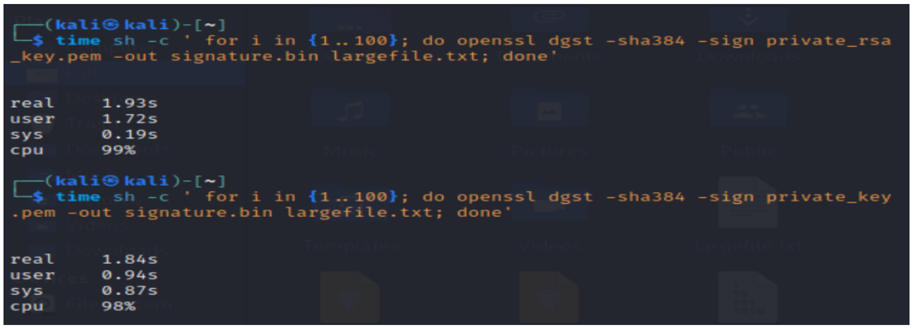
```
┌──(kali㉿kali)-[~]
└─$ time sh -c ' for i in {1..100}; do openssl dgst -sha256 -sign private_rsa
_key.pem -out signature.bin largefile.txt; done'

real     2.39s
user     2.24s
sys      0.13s
cpu      99%
```

g)

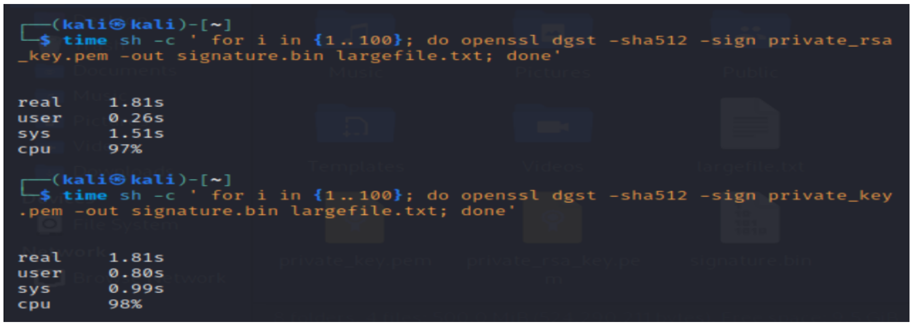


```
┌──(kali㉿kali)-[~]
└─$ time sh -c ' for i in {1..100}; do openssl dgst -sha256 -sign private_key
.pem -out signature.bin largefile.txt; done'

real     2.45s
user     1.96s
sys      0.41s
cpu      96%
```

h)

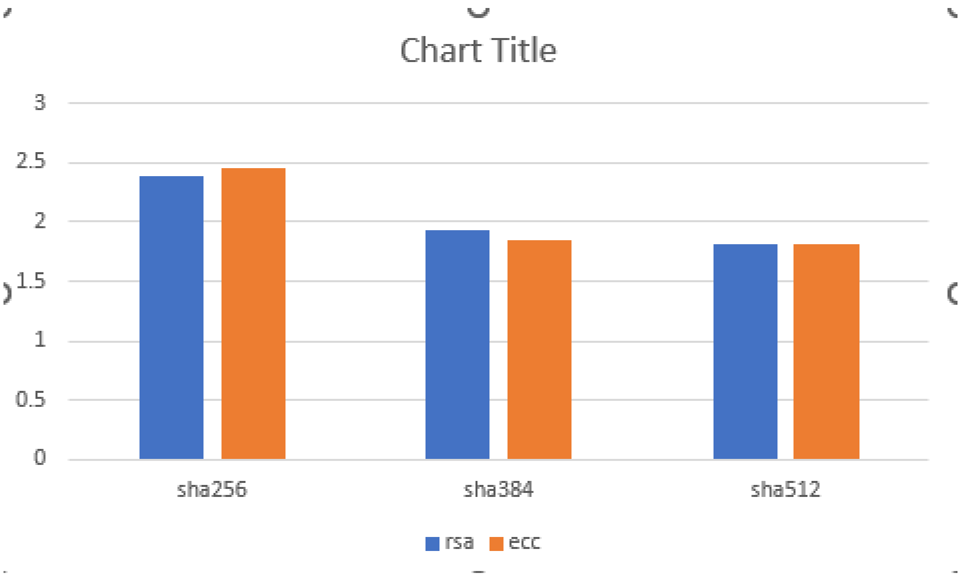


```
┌──(kali㉿kali)-[~]
└─$ time sh -c ' for i in {1..100}; do openssl dgst -sha384 -sign private_rsa
_key.pem -out signature.bin largefile.txt; done'

real     1.93s
user     1.72s
sys      0.19s
cpu      99%

┌──(kali㉿kali)-[~]
└─$ time sh -c ' for i in {1..100}; do openssl dgst -sha384 -sign private_key
.pem -out signature.bin largefile.txt; done'

real     1.84s
user     0.94s
sys      0.87s
cpu      98%
```

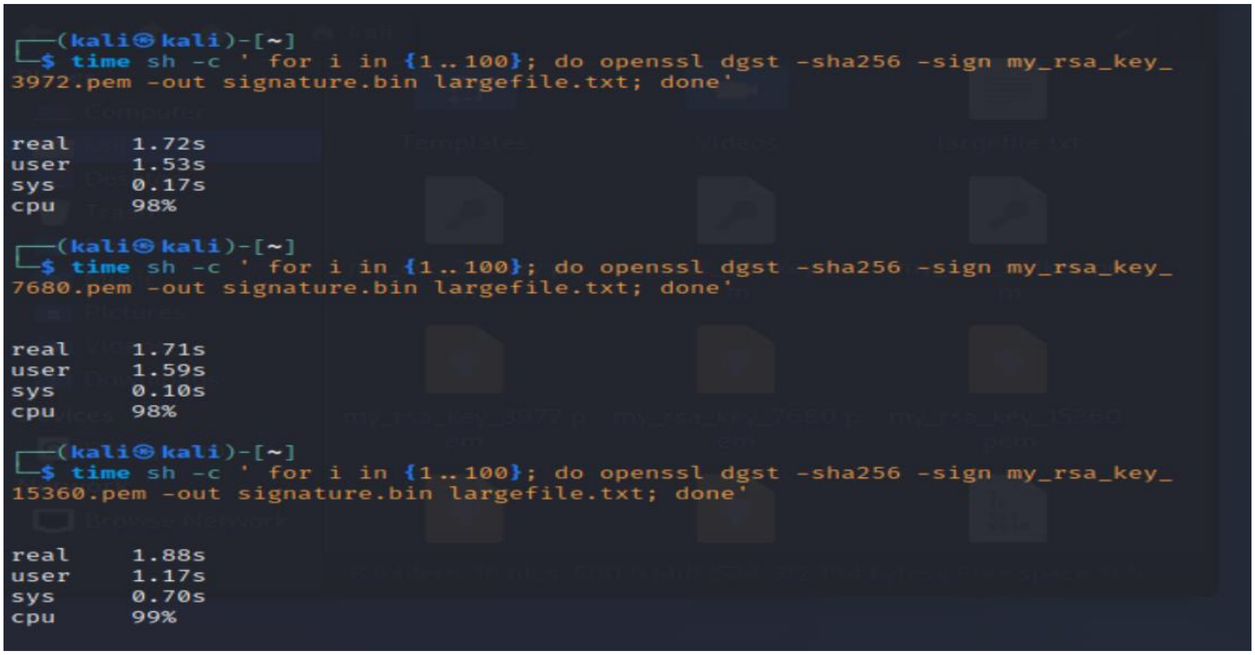


```
┌──(kali㉿kali)-[~]
└─$ time sh -c ' for i in {1..100}; do openssl dgst -sha512 -sign private_rsa
_key.pem -out signature.bin largefile.txt; done'

real    1.81s
user    0.26s
sys     1.51s
cpu     97%

┌──(kali㉿kali)-[~]
└─$ time sh -c ' for i in {1..100}; do openssl dgst -sha512 -sign private_key
.pem -out signature.bin largefile.txt; done'

real    1.81s
user    0.80s
sys     0.99s
cpu     98%
```

i) graph of secp224r1:

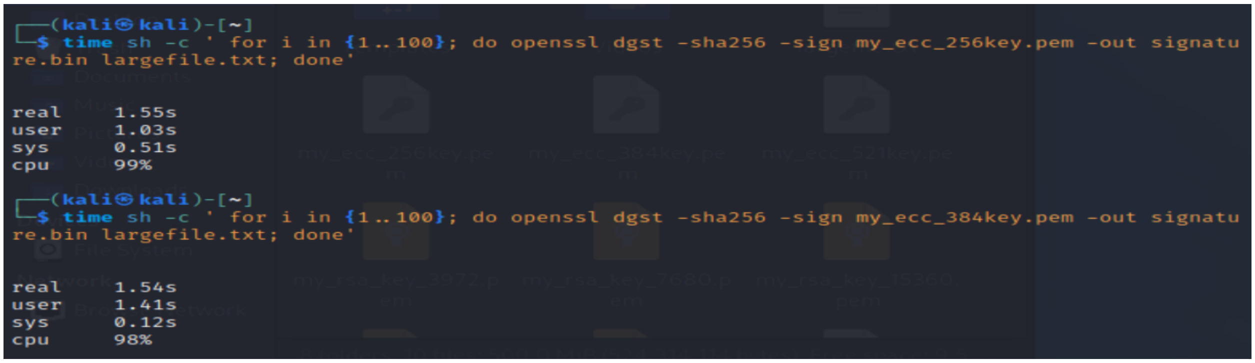


j) sha256 rsa

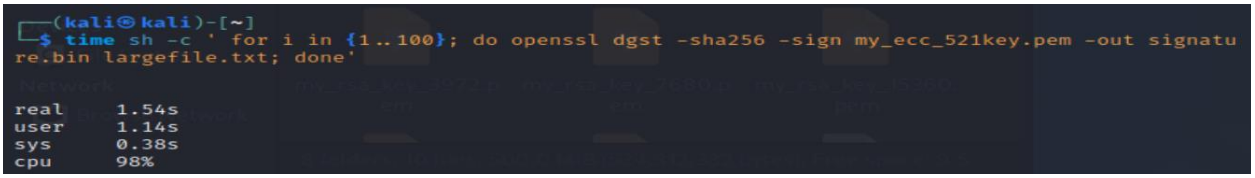


```
┌──(kali㉿kali)-[~]
└─$ time sh -c ' for i in {1..100}; do openssl dgst -sha256 -sign my_rsa_key_
3972.pem -out signature.bin largefile.txt; done'

real    1.72s
user    1.53s
sys     0.17s
cpu     98%

┌──(kali㉿kali)-[~]
└─$ time sh -c ' for i in {1..100}; do openssl dgst -sha256 -sign my_rsa_key_
7680.pem -out signature.bin largefile.txt; done'

real    1.71s
user    1.59s
sys     0.10s
cpu     98%

┌──(kali㉿kali)-[~]
└─$ time sh -c ' for i in {1..100}; do openssl dgst -sha256 -sign my_rsa_key_
15360.pem -out signature.bin largefile.txt; done'

real    1.88s
user    1.17s
sys     0.70s
cpu     99%
```

Sha 256 ecc

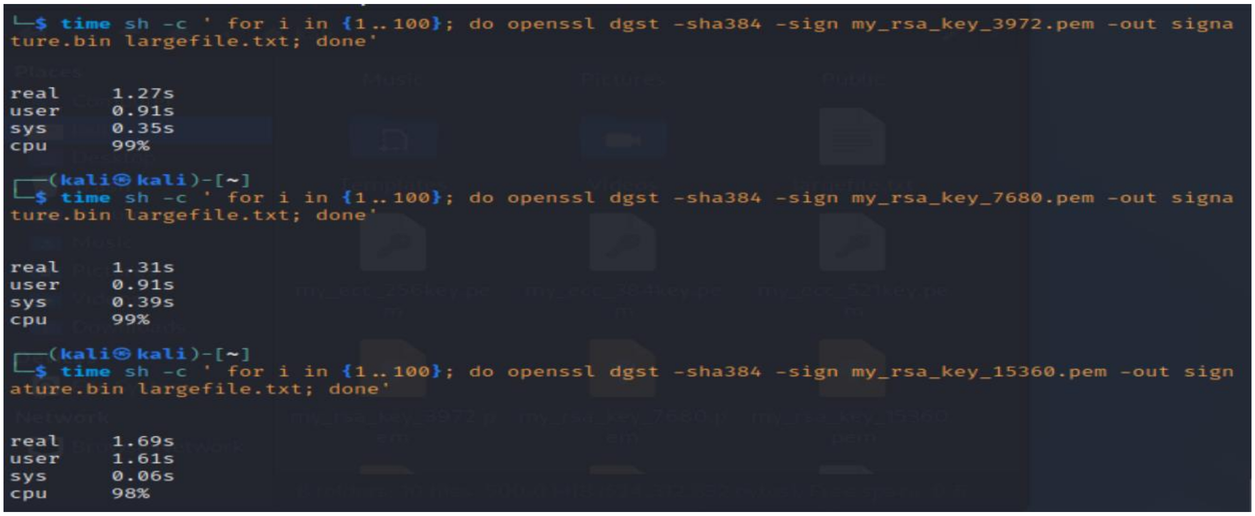


```
┌──(kali㉿kali)-[~]
└─$ time sh -c ' for i in {1..100}; do openssl dgst -sha256 -sign my_ecc_256key.pem -out signatu
re.bin largefile.txt; done'

real    1.55s
user    1.03s
sys     0.51s
cpu     99%

┌──(kali㉿kali)-[~]
└─$ time sh -c ' for i in {1..100}; do openssl dgst -sha256 -sign my_ecc_384key.pem -out signatu
re.bin largefile.txt; done'

real    1.54s
user    1.41s
sys     0.12s
cpu     98%
```



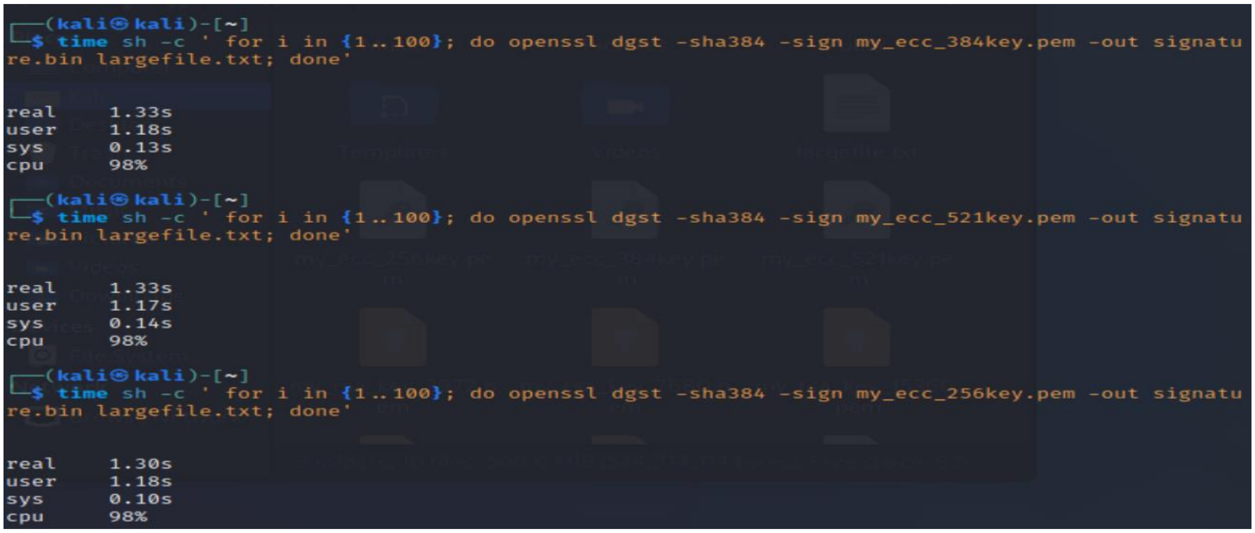
```
┌──(kali㉿kali)-[~]
└─$ time sh -c ' for i in {1..100}; do openssl dgst -sha256 -sign my_ecc_521key.pem -out signatu
re.bin largefile.txt; done'

real    1.54s
user    1.14s
sys     0.38s
cpu     98%
```

Rsa Sha 384



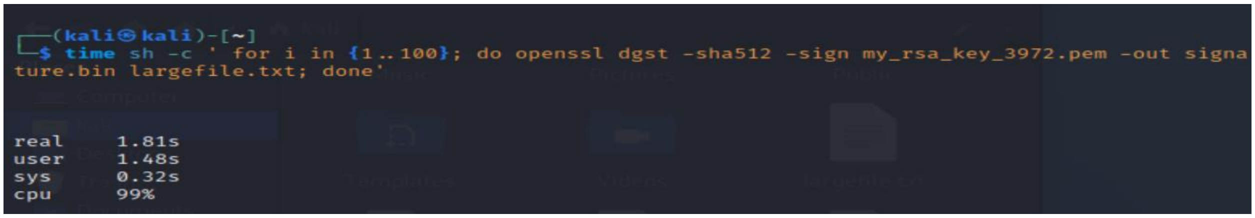
```
└─$ time sh -c ' for i in {1..100}; do openssl dgst -sha384 -sign my_rsa_key_3972.pem -out signa
ture.bin largefile.txt; done'

real    1.27s
user    0.91s
sys     0.35s
cpu     99%

┌──(kali㉿kali)-[~]
└─$ time sh -c ' for i in {1..100}; do openssl dgst -sha384 -sign my_rsa_key_7680.pem -out signa
ture.bin largefile.txt; done'

real    1.31s
user    0.91s
sys     0.39s
cpu     99%

┌──(kali㉿kali)-[~]
└─$ time sh -c ' for i in {1..100}; do openssl dgst -sha384 -sign my_rsa_key_15360.pem -out sign
ature.bin largefile.txt; done'

real    1.69s
user    1.61s
sys     0.06s
cpu     98%
```

Ecc sha 384

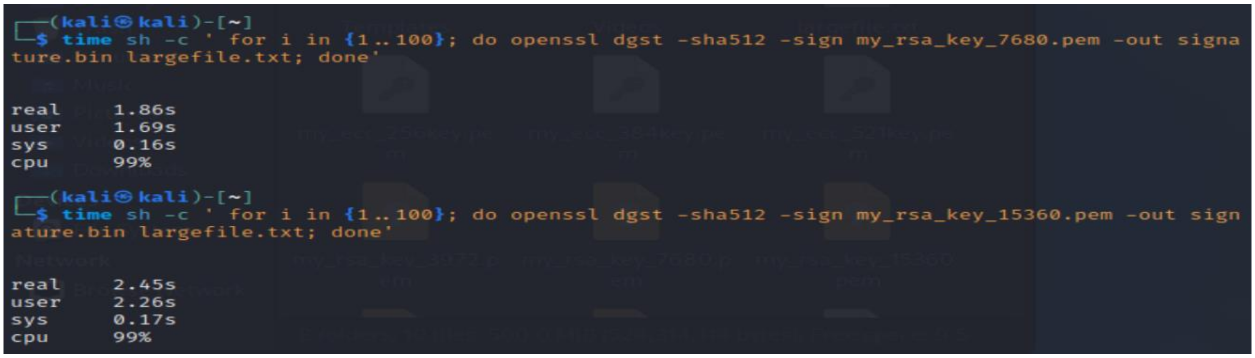


```
┌──(kali㉿kali)-[~]
└─$ time sh -c ' for i in {1..100}; do openssl dgst -sha384 -sign my_ecc_384key.pem -out signatu
re.bin largefile.txt; done'

real    1.33s
user    1.18s
sys     0.13s
cpu     98%
┌──(kali㉿kali)-[~]
└─$ time sh -c ' for i in {1..100}; do openssl dgst -sha384 -sign my_ecc_521key.pem -out signatu
re.bin largefile.txt; done'

real    1.33s
user    1.17s
sys     0.14s
cpu     98%
┌──(kali㉿kali)-[~]
└─$ time sh -c ' for i in {1..100}; do openssl dgst -sha384 -sign my_ecc_256key.pem -out signatu
re.bin largefile.txt; done'

real    1.30s
user    1.18s
sys     0.10s
cpu     98%
```

Rsa sha512

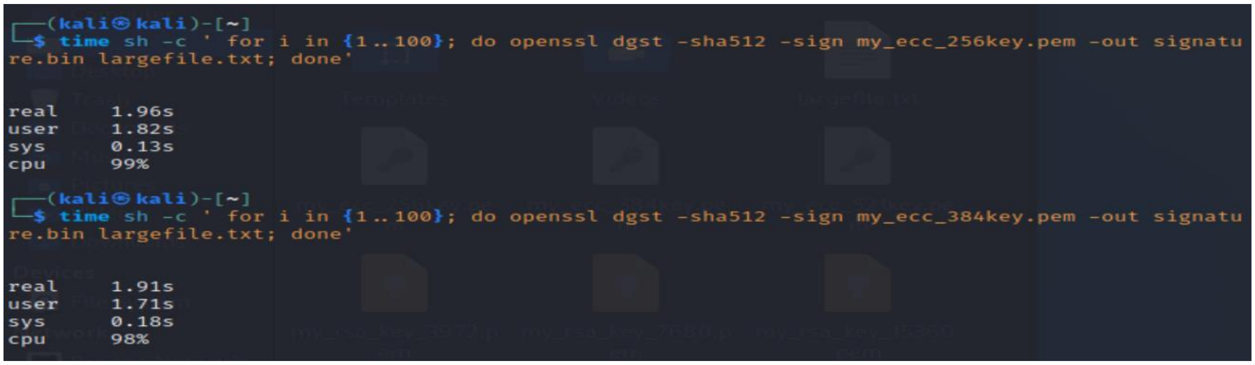


```
┌──(kali㉿kali)-[~]
└─$ time sh -c ' for i in {1..100}; do openssl dgst -sha512 -sign my_rsa_key_3972.pem -out signa
ture.bin largefile.txt; done'

real    1.81s
user    1.48s
sys     0.32s
cpu     99%
```

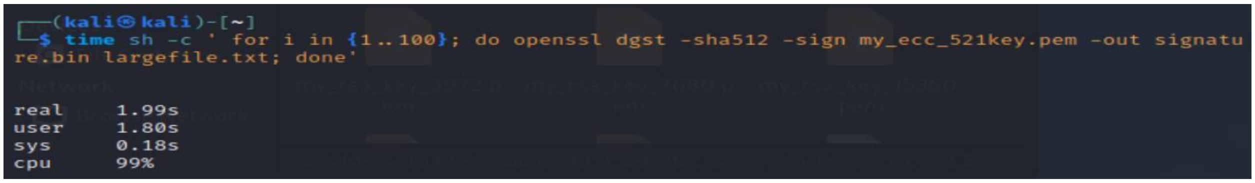


```
┌──(kali㉿kali)-[~]
└─$ time sh -c ' for i in {1..100}; do openssl dgst -sha512 -sign my_rsa_key_7680.pem -out signa
ture.bin largefile.txt; done'

real    1.86s
user    1.69s
sys     0.16s
cpu     99%
┌──(kali㉿kali)-[~]
└─$ time sh -c ' for i in {1..100}; do openssl dgst -sha512 -sign my_rsa_key_15360.pem -out sign
ature.bin largefile.txt; done'

real    2.45s
user    2.26s
sys     0.17s
cpu     99%
```

Ecc sha512

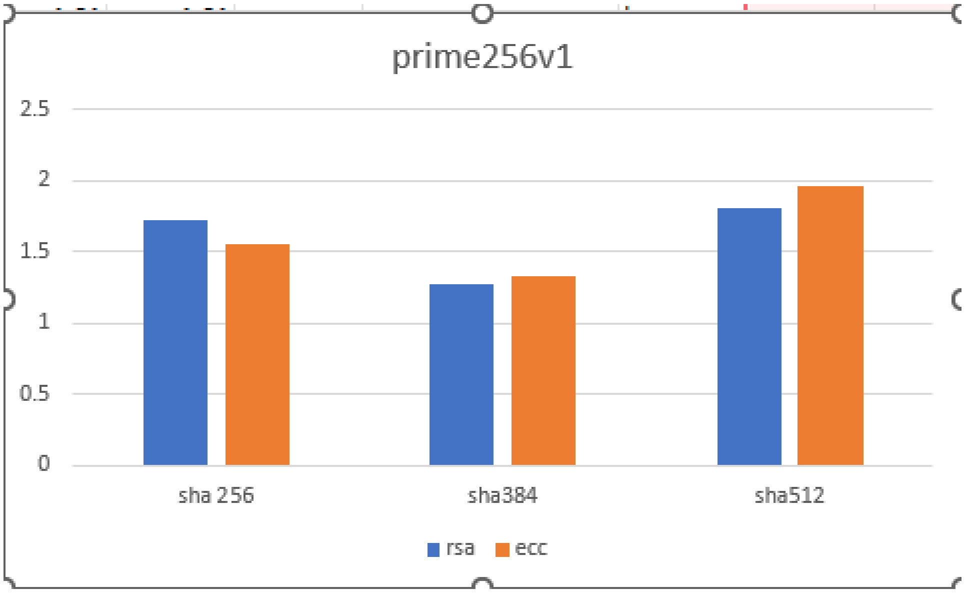


```
┌──(kali㉿kali)-[~]
└─$ time sh -c ' for i in {1..100}; do openssl dgst -sha512 -sign my_ecc_256key.pem -out signatu
re.bin largefile.txt; done'

real    1.96s
user    1.82s
sys     0.13s
cpu     99%
┌──(kali㉿kali)-[~]
└─$ time sh -c ' for i in {1..100}; do openssl dgst -sha512 -sign my_ecc_384key.pem -out signatu
re.bin largefile.txt; done'

real    1.91s
user    1.71s
sys     0.18s
cpu     98%
```

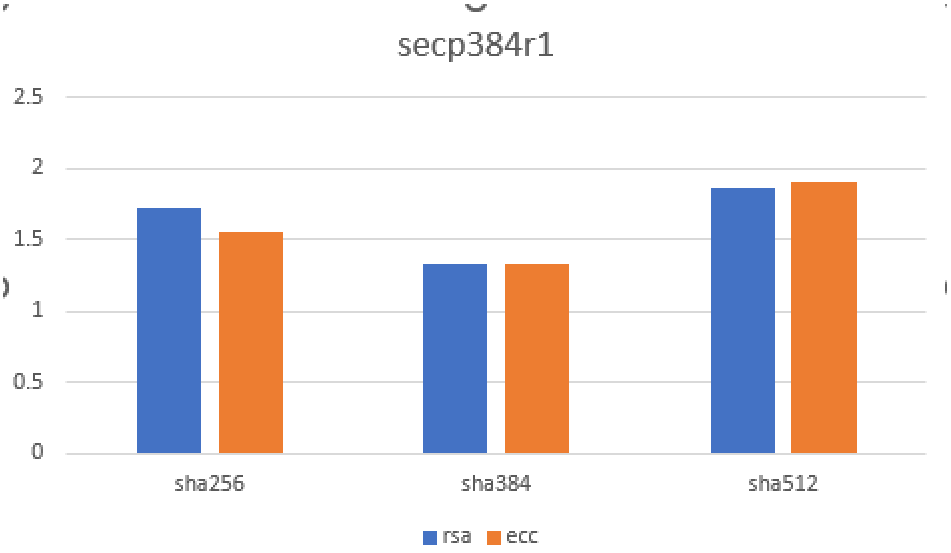


```
┌──(kali㉿kali)-[~]
└─$ time sh -c ' for i in {1..100}; do openssl dgst -sha512 -sign my_ecc_521key.pem -out signatu
re.bin largefile.txt; done'

real    1.99s
user    1.80s
sys     0.18s
cpu     99%
```

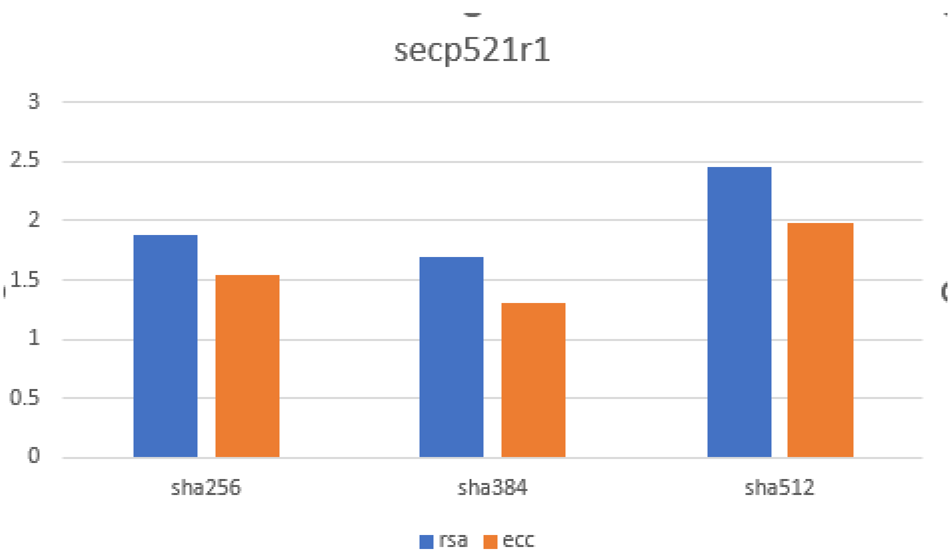
j) graph of prime256v1



Graph of secp384r1



Graph of 521r1



k)

The plots show that the key generation time for ECC is generally faster than for RSA across all four sets of algorithm settings. This trend is in line with the theoretical ideas we have discussed in class, which suggest that ECC is generally faster than RSA for cryptographic operations of the same security level. The difference in performance between the two algorithms is most pronounced for the larger key sizes, where ECC key generation is often several orders of magnitude faster than RSA. This can be explained by the fact that ECC relies on smaller key sizes than RSA to provide equivalent security, which reduces the computational complexity of key generation and other cryptographic operations. Overall, the results are as expected and support the use of ECC over RSA for applications where performance is a critical consideration.

--------------------------------the end--------------------------