CS315: Assignment-13

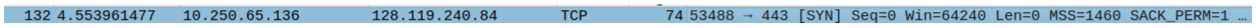
Name: *Shahil Patel*

Roll No.: *200010039*

Part-2: A first look at the captured trace

1. What is the packet number in your trace that contains the initial TCP SYN message? (By "packet number," we meant the number in the "No." column at the left of the Wireshark display, not the sequence number in the TCP segment itself).

Ans: The packet number is trace that contains the initial TCP SYN message is: **132**



```
132 4.553961477    10.250.65.136        128.119.240.84        TCP        74 53488 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 …
```

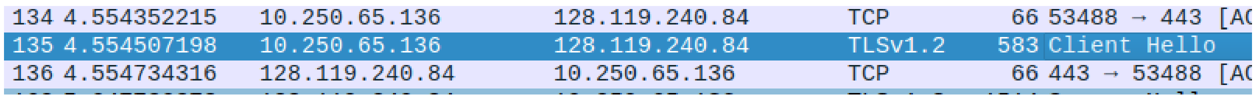
2. Is the TCP connection set up before or after the first TLS message is sent from client to server?

Ans: After a TCP connection has been established through a TCP handshake, The TLS handshake takes place.

Part-3: The TLS Handshake: Client Hello message

1. What is the packet number in your trace that contains the TLS *Client Hello* message?

Ans: The packet number in trace that contains the TLS Client Hello Message is: **135.**



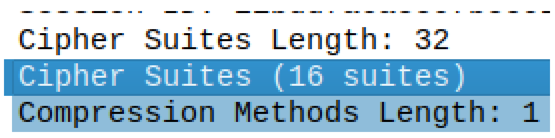
```
134 4.554352215   10.250.65.136     128.119.240.84     TCP        66 53488 → 443 [AC
135 4.554507198   10.250.65.136     128.119.240.84     TLSv1.2   583 Client Hello
136 4.554734316   128.119.240.84    10.250.65.136      TCP        66 443 → 53488 [AC
```

2. What version of TLS is your client running, as declared in the *Client Hello* message?

Ans: The version of TLS as declared in the Client Hello Message: **TLSv1.2**

3. How many cipher suites are supported by your client, as declared in the *Client Hello* message? A cipher suite is a set of related cryptographic algorithms that determine how session keys will be derived, and hoid-at-commonName=www.cs.umass.edua HMAC algorithm.

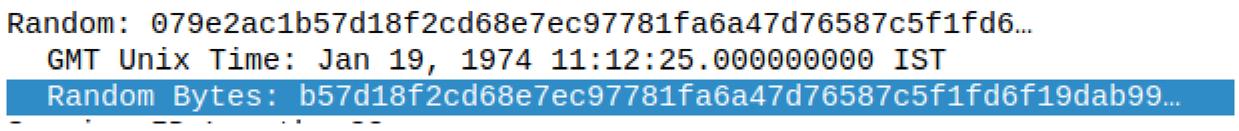
Ans: **16** cipher suites are supported by our client as declared in the client Hello message.



```
Cipher Suites Length: 32
Cipher Suites (16 suites)
Compression Methods Length: 1
```

4. Your client generates and sends a string of “random bytes” to the server in the *Client Hello* message. What are the first two hexadecimal digits in the random bytes field of the *Client Hello* message? Enter the two hexadecimal digits (without spaces between the hex digits and without any leading '0x' , using lowercase letters where needed). *Hint:* be careful to fully dig into the Random field to find the Random Bytes subfield (do not consider the GMT UNIX Time subfield of Random).

Ans: The first two hexadecimal digits in the random bytes field of the Client Hello message are: **b5.**



```
Random: 079e2ac1b57d18f2cd68e7ec97781fa6a47d76587c5f1fd6…
  GMT Unix Time: Jan 19, 1974 11:12:25.000000000 IST
  Random Bytes: b57d18f2cd68e7ec97781fa6a47d76587c5f1fd6f19dab99…
```

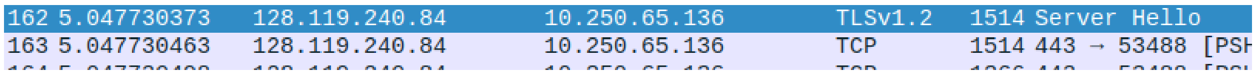
5. What is the purpose(s) of the “random bytes” field in the *Client Hello* message? Note: you’ll have do some searching and reading to get the answer to this question; see section 8.6 and in RFC 5246 (section 8.1 in RFC 5246 in particular).

Ans: The premaster secret is created using the random bytes. That is, a shared secret key known as the pre master key is created by combining the random bytes with additional information sent during the TLS handshake.

Part-4: The TLS Handshake: Server Hello message

1. What is the packet number in your trace that contains the TLS *Server Hello* message?

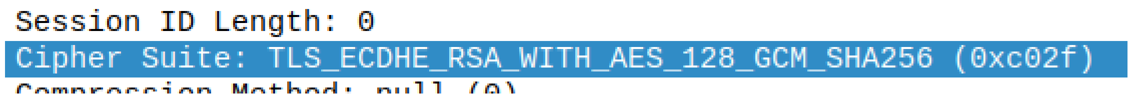
Ans: The packet number in the trace that contains the TLD Server Hello message is: **162.**



```
162 5.047730373   128.119.240.84   10.250.65.136   TLSv1.2   1514 Server Hello
163 5.047730463   128.119.240.84   10.250.65.136   TCP       1514 443 → 53488 [PSH
```

2. Which cipher suite has been chosen by the server from among those offered in the earlier *Client Hello* message?

Ans: The cipher suite chosen by the server is: **TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0Xc02f).**



```
Session ID Length: 0
Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)
```

3. Does the *Server Hello* message contain random bytes, similar to how the *Client Hello* message contained random bytes? And if so, what is/are their purpose(s)?

Ans: ***Yes***, the server Hello message contain random bytes, similar to how the Client Hello message contained random bytes. The purpose of the random bytes is to provide assurance that the person you want to connect with is genuinely present and participating in the conversation, rather than just playing back a recorded session to you and possibly passing for someone else.

This is made possible by the randomization, which stops attackers from simulating a number of sessions beforehand and then choosing the most pertinent one for you.

4. What is the packet number in your trace for the TLS message part that contains the public key certificate for the www.cics.umass.edu server (actually the www.cs.umass.edu server)?

Ans: The packet number in the trace for the TLS message part that contains the public key certificate is: **168**

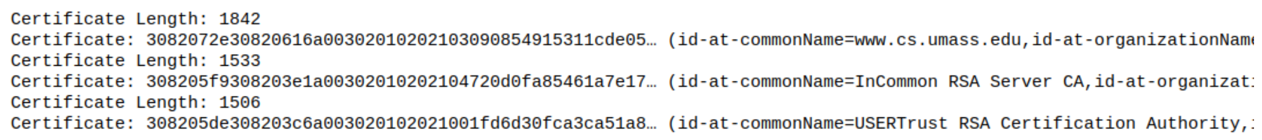
5. A server may return more than one certificate. If more than one certificate is returned, are all of these certificates for www.cs.umass.edu? If not all are for www.cs.umass.edu, then who *are* these other certificates for? You can determine who the certificate is for by checking the id-at-commonName field in the retuned certificate.

Ans: The server return 3 certificates, This all are not for www.cs.umass.edu.

The certificates are for:

- **www.cs.umass.edu**

- **InCommon RSA Server CA**
- **USERTrust RSA Certification Authority**



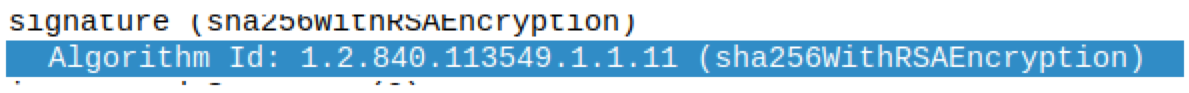
```
Certificate Length: 1842
Certificate: 3082072e30820616a003020102021030908549153 11cde05… (id-at-commonName=www.cs.umass.edu,id-at-organizationNam
Certificate Length: 1533
Certificate: 308205f9308203e1a00302010202104720d0fa85461a7e17… (id-at-commonName=InCommon RSA Server CA,id-at-organizat
Certificate Length: 1506
Certificate: 308205de308203c6a003020102021001fd6d30fca3ca51a8… (id-at-commonName=USERTrust RSA Certification Authority,
```

6. What is the name of the certification authority that issued the certificate for id-at-commonName=www.cs.umass.edu?

Ans: The name of the certification authority that issued the certificate for id-at-commonName = www.cs.umass.edu is ***University of Massachusetts Amherst***

7. What digital signature algorithm is used by the CA to sign this certificate? Hint: this information can be found in the signature subfield of the SignedCertificate field of the certificate for www.cs.umass.edu.

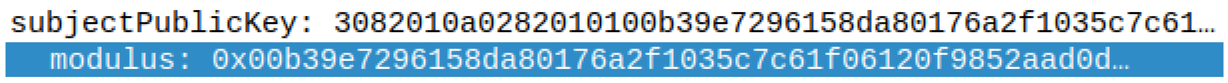
Ans: The digital signature algorithm used by the CA to sign this certificate is **1.2.840.113549.1.1.11 (sha256WithRSAEncryption).**



```
signature (sha256WithRSAEncryption)
    Algorithm Id: 1.2.840.113549.1.1.11 (sha256WithRSAEncryption)
```

8. Let's take a look at what a real public key looks like! What are the first four hexadecimal digits of the modulus of the public key being used by www.cics.umass.edu? Enter the four hexadecimal digits (without spaces between the hex digits and without any leading '0x' , using lowercase letters where needed, and including any leading 0s after '0x'). Hint: this information can be found in subjectPublicKeyInfo subfield of the SignedCertificate field of the certificate for www.cs.umass.edu.

Ans: The first 4 hexadecimal digits of the modulus of the public key being used bu www.cics.umass.edu are: **00b3.**



```
subjectPublicKey: 3082010a0282010100b39e7296158da80176a2f1035c7c61…
    modulus: 0x00b39e7296158da80176a2f1035c7c61f06120f9852aad0d…
```

9. Look in your trace to find messages between the client and a CA to get the CA's public key information, so that the client can verify that the CA-signed certificate sent by the server is indeed valid and has not been forged or altered. Do you see such a message in your trace? If so, what is the number in the trace of the first packet sent from your client to the CA? If not, explain why the client did not contact the CA.

Ans: We didn't find any messages between the client and a CA to get the CA's public key information

10. What is the packet number in your trace for the TLS message part that contains the *Server Hello Done* TLS record?

Ans: The packet number in the trace for the TLS message part that contains the Server Hello Done TLS record is: **168**

Part-5: The TLS Handshake: wrapping up the handshake

1. What is the packet number in your trace for the TLS message that contains the public key information, *Change Cipher Spec,* and *Encrypted Handshake* message, being sent from client to server?

Ans: The packet number in the trace for the TLS message that contains the public key information, Change Cipher Spec and Encrypted Handshake message is: **170.**

2. Does the client provide its own CA-signed public key certificate back to the server? If so, what is the packet number in your trace containing your client's certificate?

Ans: **No.** the client doesn't provide its own CA-signed public key certificate back to the server.

Part-6: Application data

1. What symmetric key cryptography algorithm is being used by the client and server to encrypt application data (in this case, HTTP messages)?

Ans: The given below symmetric key cryptography algorithm is being used by the client and server to encrypt application data:
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)

2. In which of the TLS messages is this symmetric key cryptography algorithm finally decided and declared?

Ans: The symmetric key cryptography algorithm to be used for securing the communication is finally decided and declared during the "**Cipher Suite Negotiation**" step in the "**ClientHello**" and "**ServerHello**" messages in the TLS handshake protocol.

3. What is the packet number in your trace for the first encrypted message carrying application data from client to server?

Ans: Packet no **302** contains the the first encrypted message carrying application data from client to server.

4. What do you think the content of this encrypted application-data is, given that this trace was generated by fetching the homepage of www.cics.umass.edu?

Ans: We will see the content based on the home page and then it will get encrypted.

5. What packet number contains the client-to-server TLS message that shuts down the TLS connection? Because TLS messages are encrypted in our Wireshark traces, we can't actually look *inside* a TLS message and so we'll have to make an educated guess here.

Ans: The client-to-server TLS message that terminates the TLS connection is contained in packet number **3719**.