**Cryptography, Network and Security**

Assignment 11

1] Demonstration of SSL using Wireshark

* Demonstration of SSL using Wireshark

Description:

A TLS connection is initiated using a sequence known as the [TLS handshake](https://www.cloudflare.com/learning/ssl/what-happens-in-a-tls-handshake/). When a user navigates to a website that uses TLS, the TLS handshake begins between the user's device (also known as the *client* device) and the web server.

During the TLS handshake, the user's device and the web server:

* Specify which version of TLS (TLS 1.0, 1.2, 1.3, etc.) they will use
* Decide on which cipher suites (see below) they will use
* Authenticate the identity of the server using the server's TLS certificate
* Generate session keys for encrypting messages between them after the handshake is complete

The TLS handshake establishes a cipher suite for each communication session. The cipher suite is a set of algorithms that specifies details such as which shared [encryption keys](https://www.cloudflare.com/learning/ssl/what-is-a-cryptographic-key/), or [session keys](https://www.cloudflare.com/learning/ssl/what-is-a-session-key/), will be used for that particular session. TLS is able to set the matching session keys over an unencrypted channel thanks to a technology known as [public key cryptography](https://www.cloudflare.com/learning/ssl/how-does-public-key-encryption-work/).

The handshake also handles authentication, which usually consists of the server proving its identity to the client. This is done using public keys. Public keys are encryption keys that use one-way encryption, meaning that anyone with the public key can unscramble the data encrypted with the server's private key to ensure its authenticity, but only the original sender can encrypt data with the private key. The server's public key is part of its TLS certificate.

Once data is encrypted and authenticated, it is then signed with a message authentication code (MAC). The recipient can then verify the MAC to ensure the integrity of the data. This is kind of like the tamper-proof foil found on a bottle of aspirin; the consumer knows no one has tampered with their medicine because the foil is intact when they purchase it.

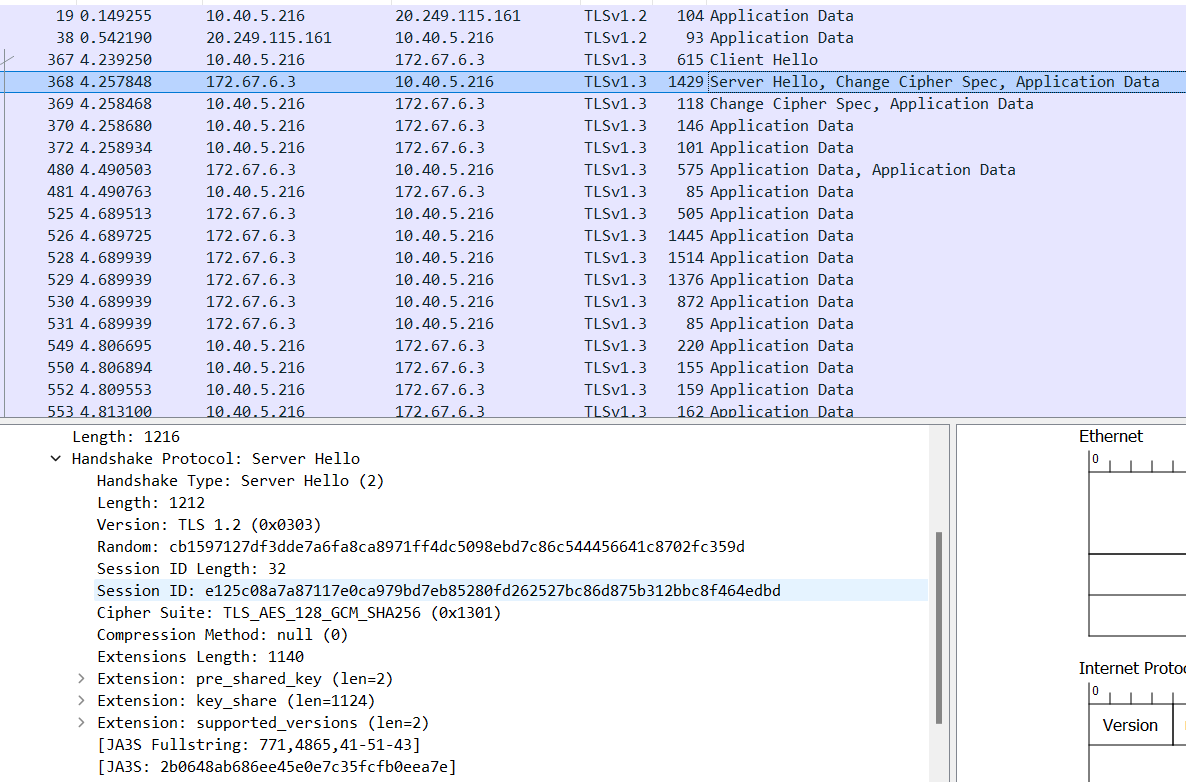
TLS Handshake:

**Step 1: Client Hello**

The client begins the communication. The first step is called **client hello**. The client lists**the versions of SSL/TLS** and **cipher suites** it’s able to use.

**Step 2: Server Hello**

The server will see the list of SSL/TLS versions and cipher suites and pick the newest the server is able to use. Then the server sends a message to the client containing the SSL/TLS version and cipher suite it chose.



**Step 3: Server Key Exchange**

After the server and client agree on the SSL/TLS version and cipher suite, the server sends two things.

* SSL/TLS certificate
* public key and signature

The first is its**SSL/TLS certificate** to the client. The client (web browser) validates the server’s certificate. Web browsers store a list of Root CA(Certificate Authority) in themselves. These root CAs are third parties that are trusted by web browsers. The server’s certificate is issued by root CA or intermediate CA. Intermediate CA is a CA that is trusted by root CA.

Web browsers trust Root CA. Root CA trusts immediate CA. If the server’s certificate is issued by a trusted root CA or immediate CA, then the browser trust the server’s certificate.

The second thing the server sends is its **public key and signature**. The public key is actually included in the certificate. The client and the server use the public key to encrypt messages, which can only be decrypted with the server’s private key. The server never shares its private key with anyone.

At the end of the server key exchange, the server sends a **server hello done** message.

Until now, all the information sent between the client and server is unencrypted. The client has the server’s public key, what will the client do now?

1. It generates a random **session key** (aka pre-master key).
2. Encrypt the session key with the server’s public key.
3. Sends the encrypted session key to the server.

The encrypted session key can only be decrypted with the server’s private key. Only the server has the private key, so only the client and server can know the session key.

**Step 5. Change Cipher Spec**

The change cipher spec message is sent by both the client and server to notify the receiving party that subsequent records will be protected under the just-negotiated CipherSpec and keys

**Step 6. Encrypted Handshake**

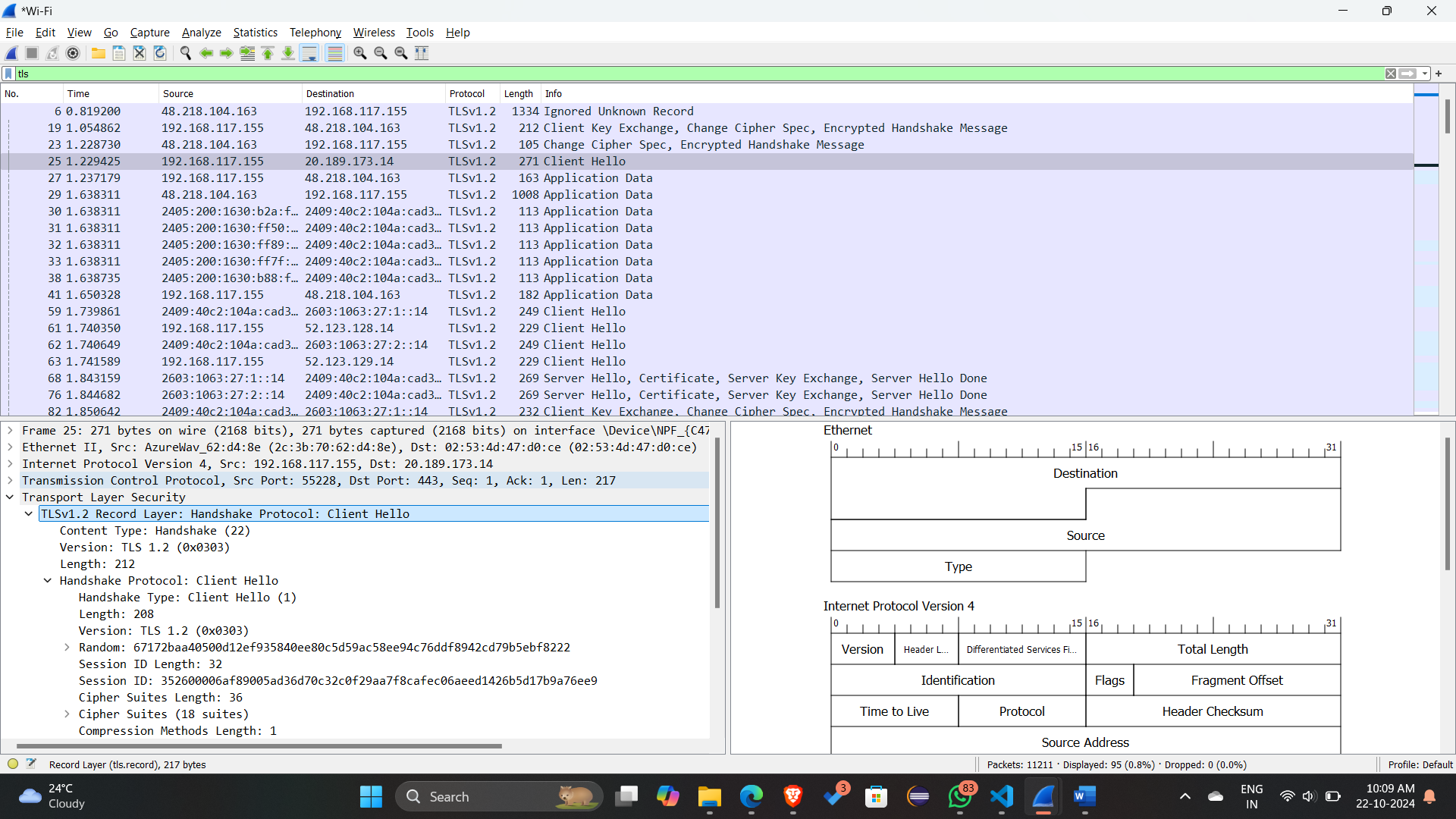
The client and the server send each other an encrypted message saying the key information is correct.

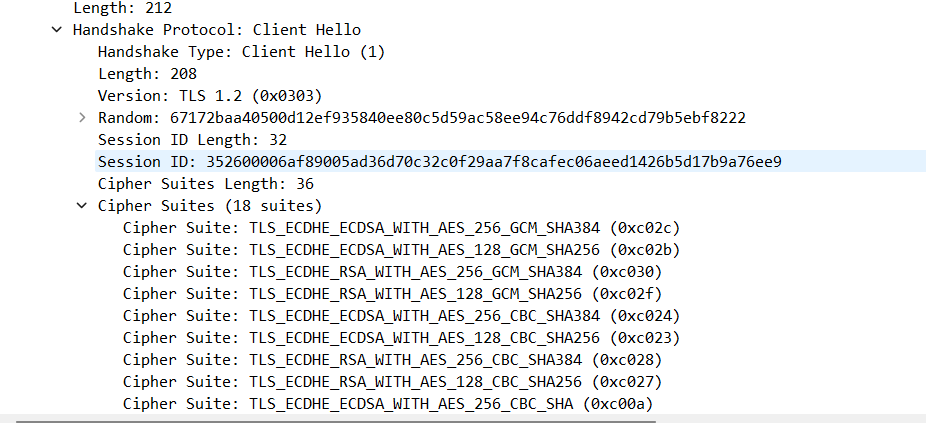
Now the client (web browser) will display a green lock in the address bar. The client and server encrypt HTTP traffic with the session key.

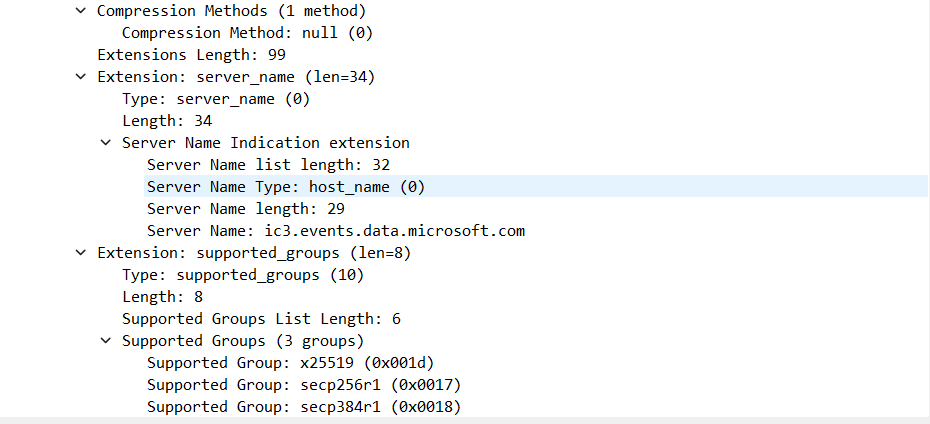
Session key is only valid in one session. If the user closes the client and visits the same server the next day, a new session key will be generated by the client.

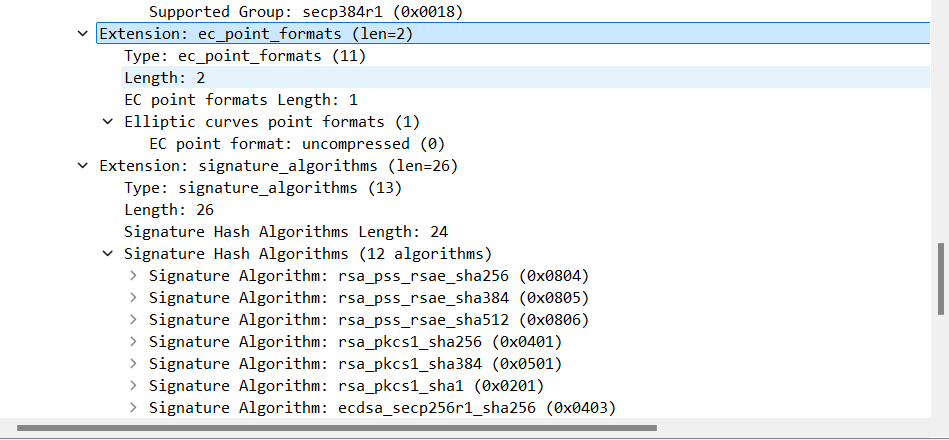
For demonstration I used - [https://youtube.com](https://youtube.com/)

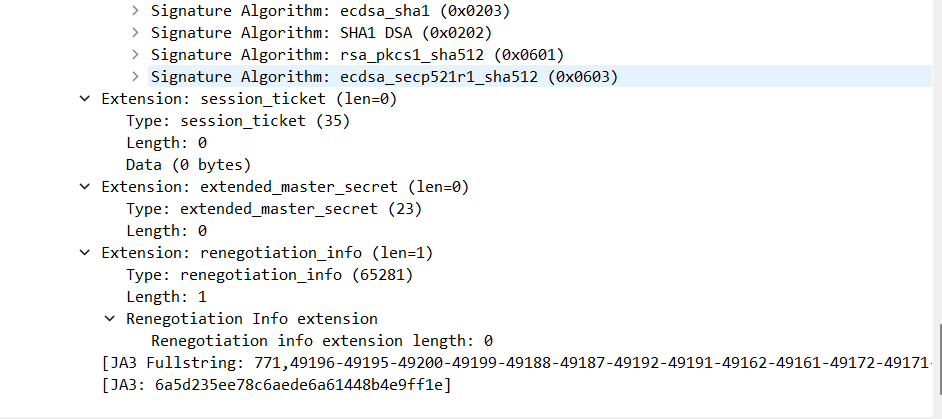
Client Hello:



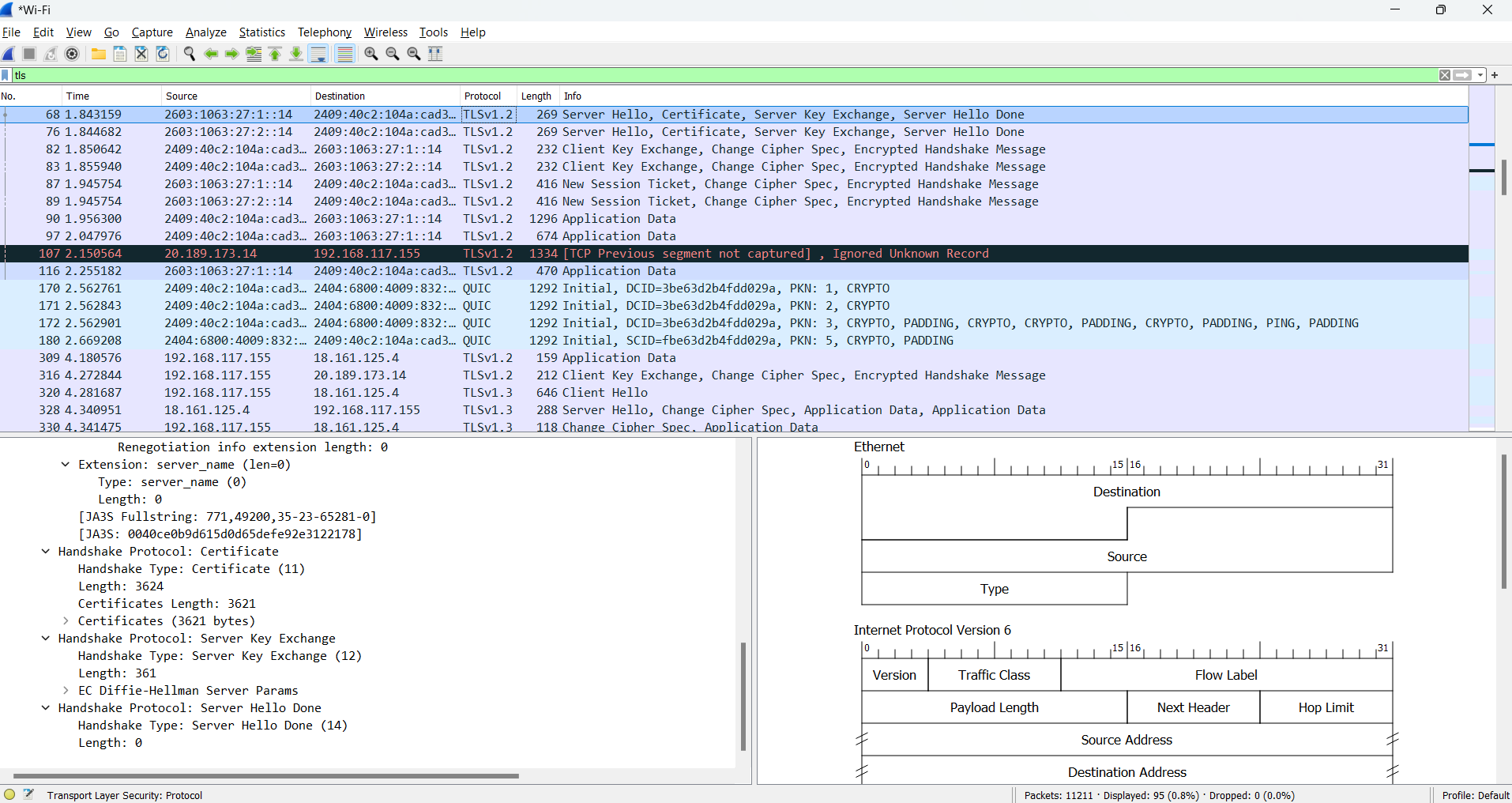


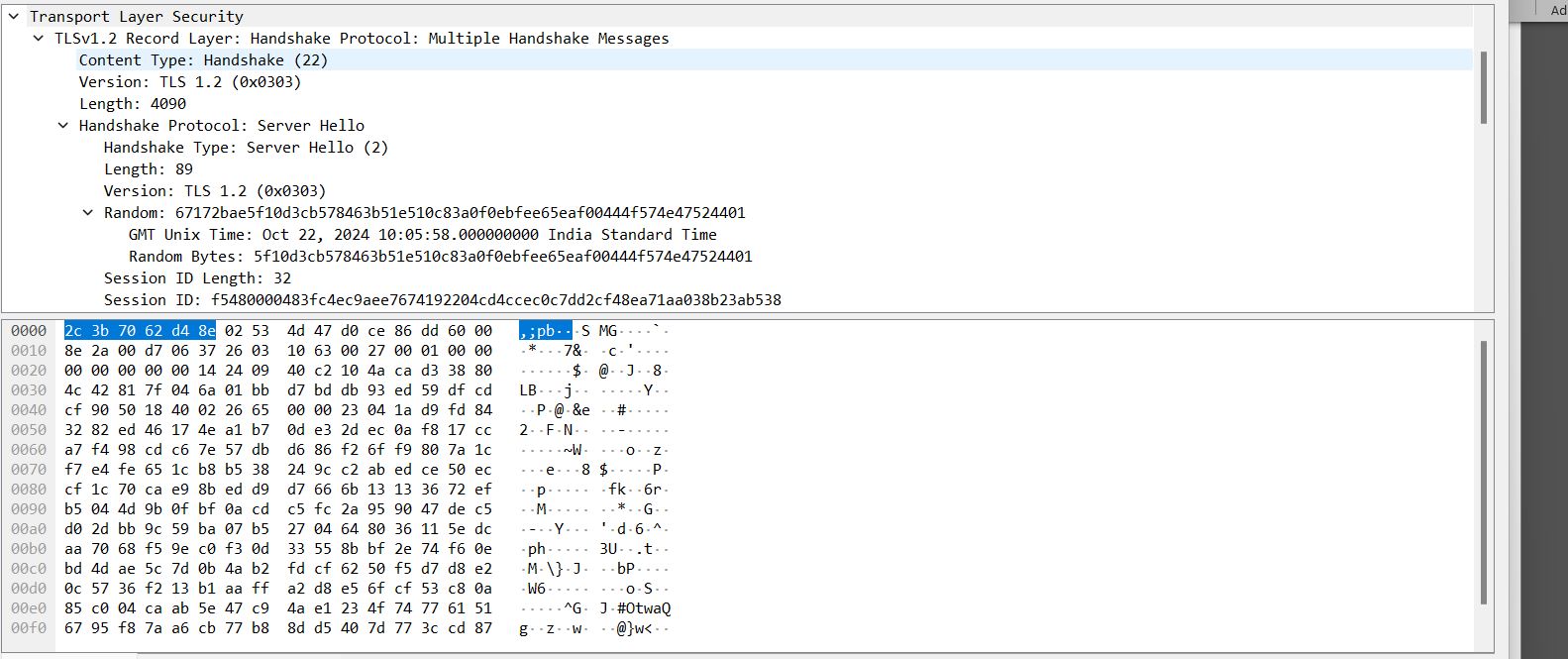




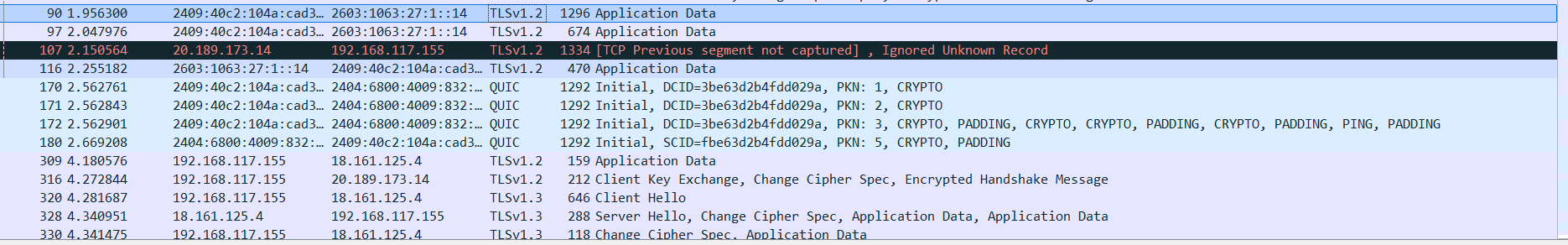


Server Hello:

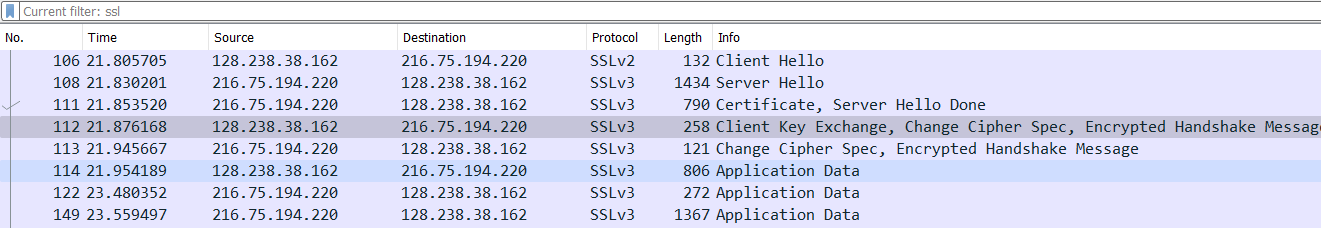


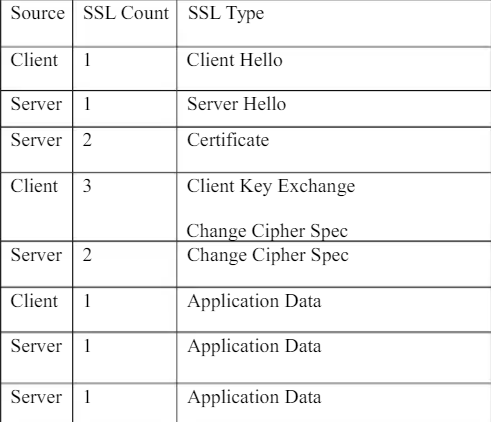


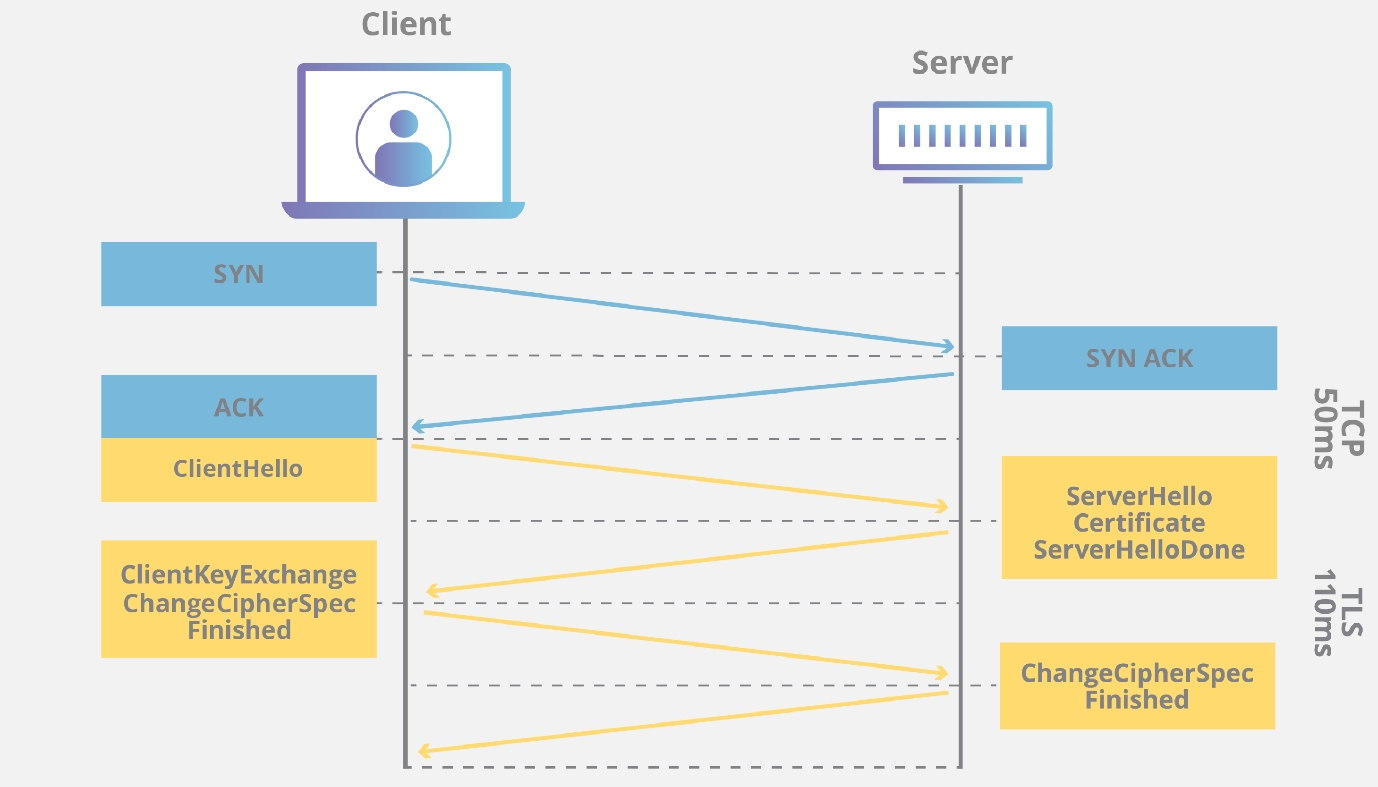
Application Data:



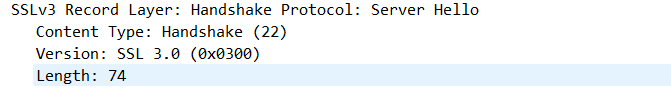
1. For each of the first 8 Ethernet frames, specify the source of the frame (client or server), determine the number of SSL records that are included in the frame, and list the SSL record types that are included in the frame. Draw a timing diagram between client and server, with one arrow for each SSL record.



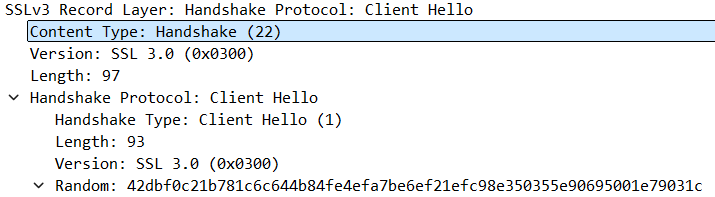




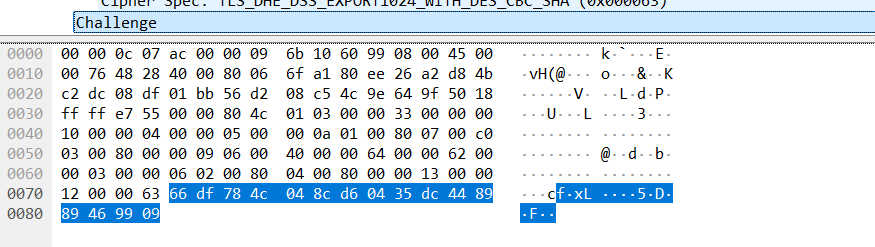
1. Each of the SSL records begins with the same three fields (with possibly different values). One of these fields is “content type” and has length of one byte. List all three fields and their lengths.



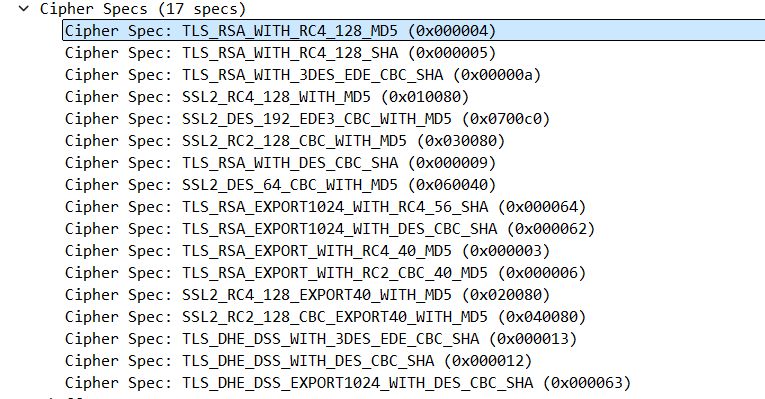
1. Expand the ClientHello record. (If your trace contains multiple ClientHello records, expand the frame that contains the first one.) What is the value of the content type?



1. Does the ClientHello record contain a nonce (also known as a “challenge”)? If so, what is the value of the challenge in hexadecimal notation?



1. Does the ClientHello record advertise the cyber suites it supports? If so, in the first listed suite, what are the public-key algorithm, the symmetric-key algorithm, and the hash algorithm?

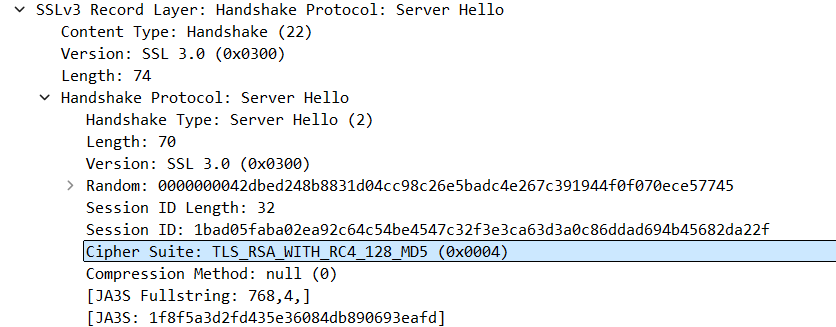


RSA – public key encryption

RC4 – Symmetric key algorithm

MD5 – For hashing

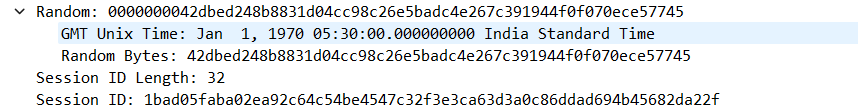
1. Locate the ServerHello SSL record. Does this record specify a chosen cipher suite? What are the algorithms in the chosen cipher suite?



1. Does this record include a nonce? If so, how long is it? What is the purpose of the client and server nonces in SSL?

Yes, record contains the nonce, which is under the Random tab,

It is 32 bits long



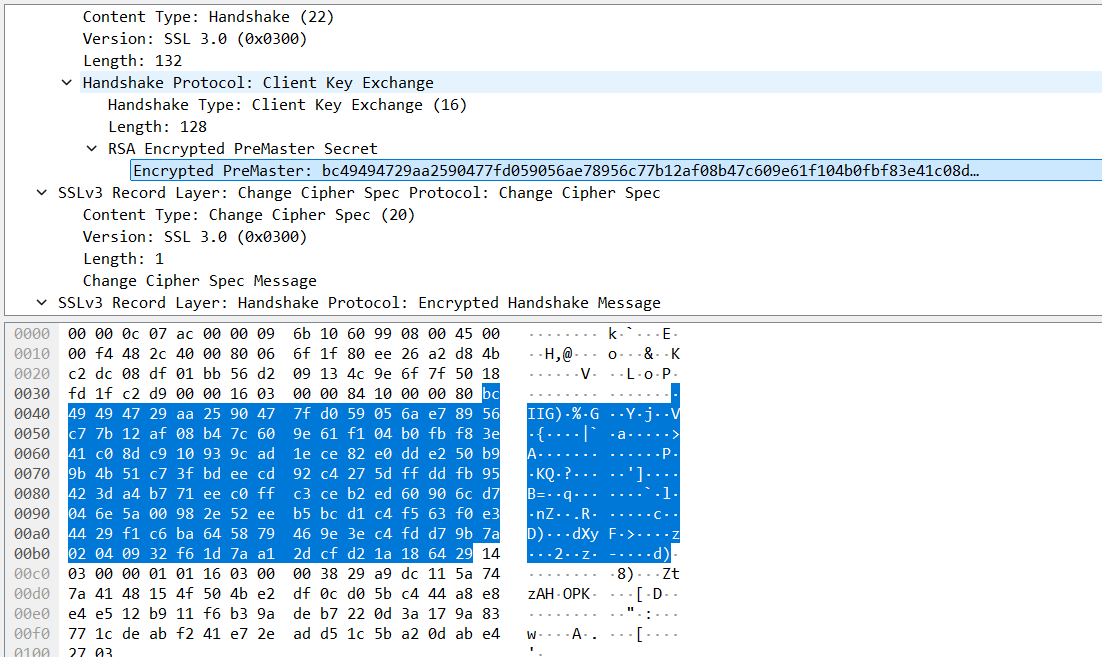
1. Does this record include a session ID? What is the purpose of the session ID?

Yes it includes session Id, It is unique identifier for SSL session

1. Does this record contain a certificate, or is the certificate included in a separate record. Does the certificate fit into a single Ethernet frame?

There is no certificate

1. Locate the client key exchange record. Does this record contain a pre-master secret? What is this secret used for? Is the secret encrypted? If so, how? How long is the encrypted secret?



Yes, it contains premaster secret

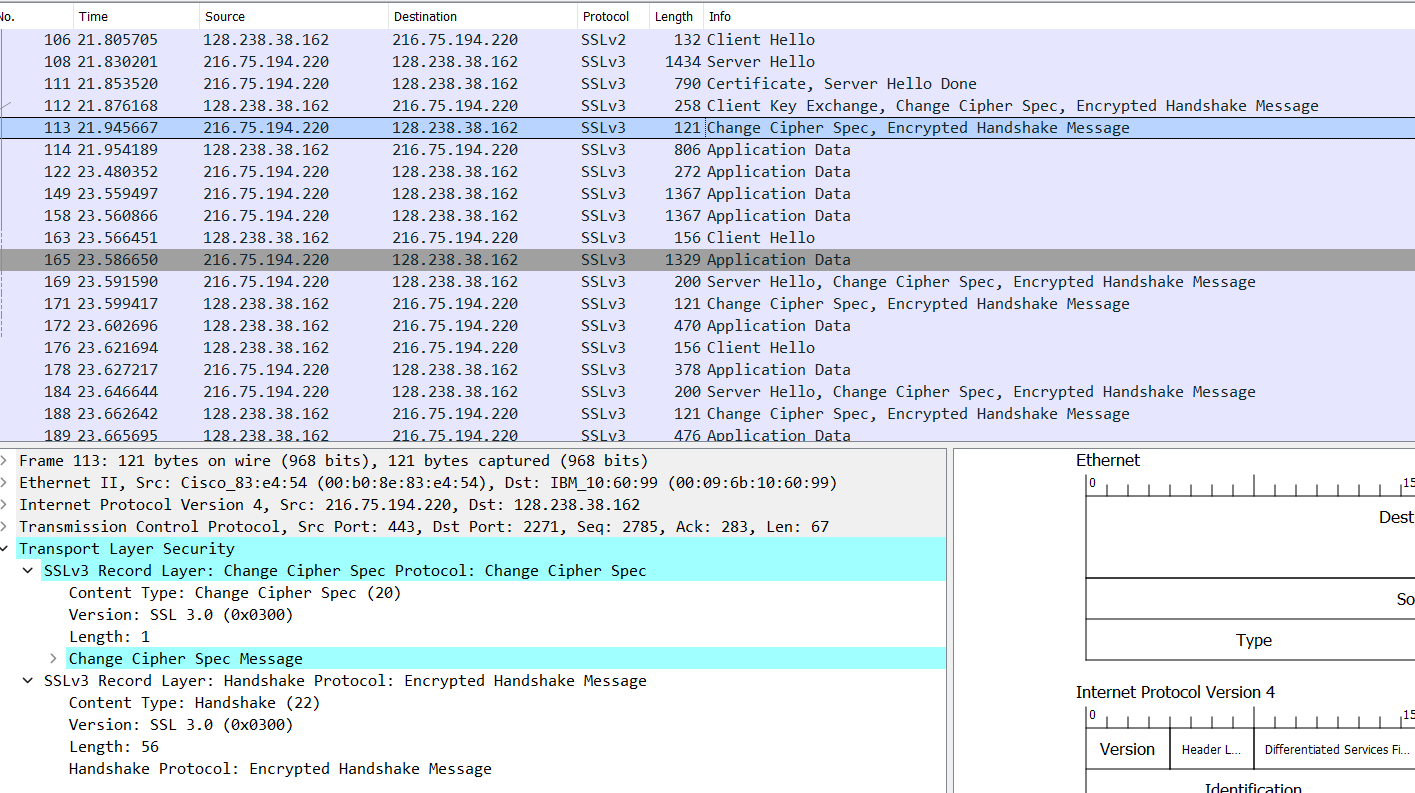
It is used to generate session keys for encryption

Secret is encrypted using public key of server

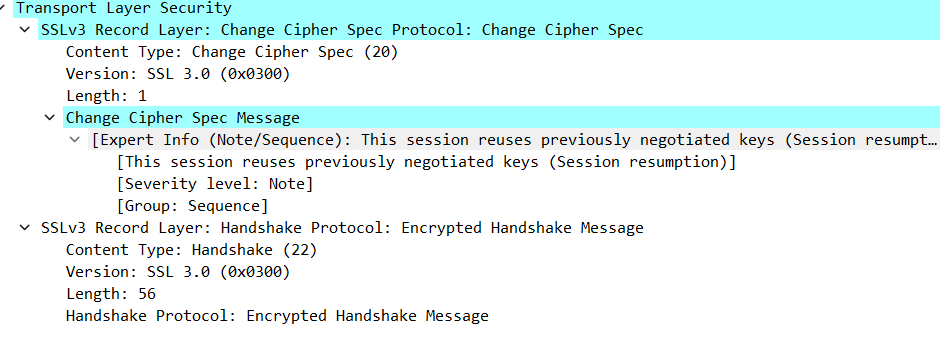
Client gets this key which it extracted from certificate from server

Secret is 128 bytes long

1. What is the purpose of the Change Cipher Spec record? How many bytes is the record in your trace?



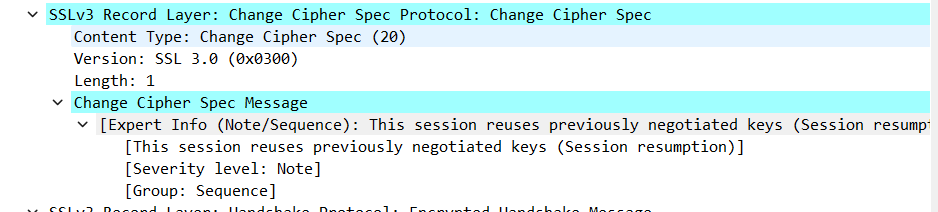
Purpose of change cipher spec record is to indicate that the contents of the following SSL record sent by client will be encrypted



1. In the encrypted handshake record, what is being encrypted? How?

MAC of the concatenation of all previous handshake message sent from this client is generated and sent to server

1. Does the server also send a change cipher record and an encrypted handshake record to the client? How are those records different from those sent by the client?

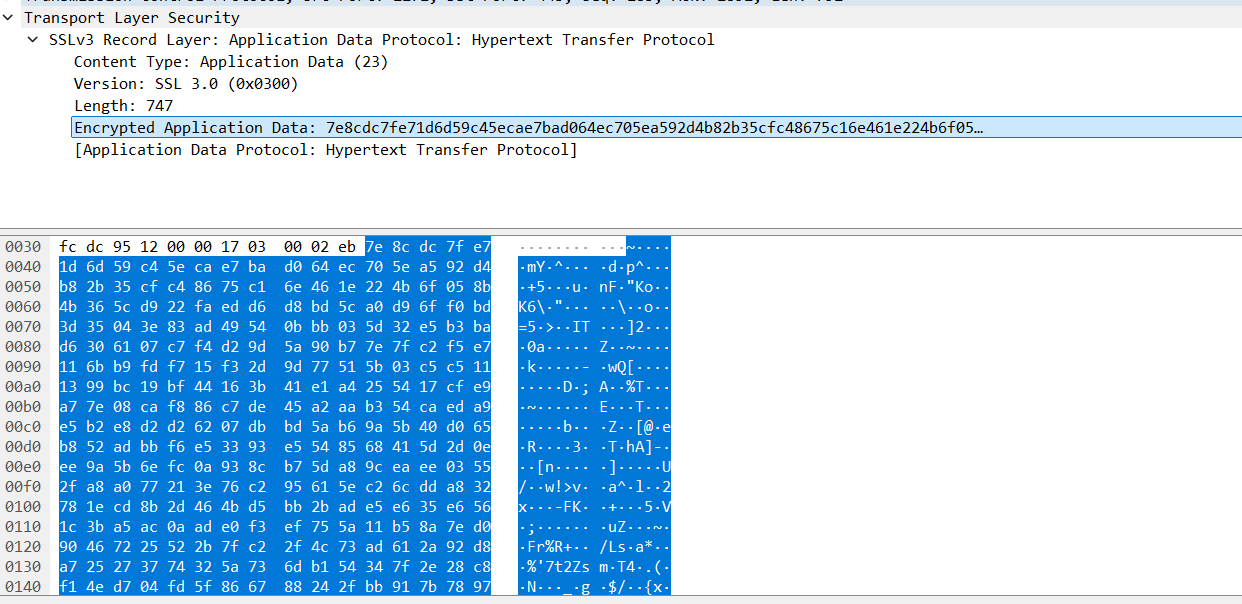


Yes server also send the change cipher record encrypted handshake to the client. It is different from the client’s one as it contains the concatenation of all handshake messages sent from server rather than from client

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

Symmetric key encryption algorithm is used to encrypt the application data in the handshake phase RC4, it uses key generated using pre-master key and nonces from both sides.

Client encryption key is used to encrpt the data being sent from client to server and server encryption key is used to encrypt the data being sent from server to client



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

Only in the first frame i.e frame number 106 version of SSL is 2 (SSLv2) and after that it changes to SSLv3

