**Final Year B.Tech. (CSE) – VII [ 2024-25]**

**6CS451: Cryptography and Network Security Lab (C&NS Lab)**

**Date: 15/10/2024**

**Assignment 11**

**PRN:** 21510017  **Name:** Onkar Anand Yemul

**11. Demonstration of SSL using Wireshark.**

**Ans:**

**SSL (Secure Sockets Layer)** and **TLS (Transport Layer Security)** are cryptographic protocols designed to secure communication over a network, especially the internet. They ensure that data transmitted between a client (e.g., web browser) and a server (e.g., website) is encrypted, authenticated, and tamper-proof.

TLS is the more recent and secure version of SSL. SSL is now considered obsolete, and TLS is widely used today (referred to as "SSL/TLS" in many contexts).

**SSL vs TLS:**

* **SSL**: Older protocol, now considered insecure due to vulnerabilities like **POODLE** and **BEAST**.
* **TLS**: Successor to SSL, currently in use with versions like **TLS 1.2** and **TLS 1.3**. TLS 1.3 is the most secure and efficient version available today.

**Goals of SSL/TLS:**

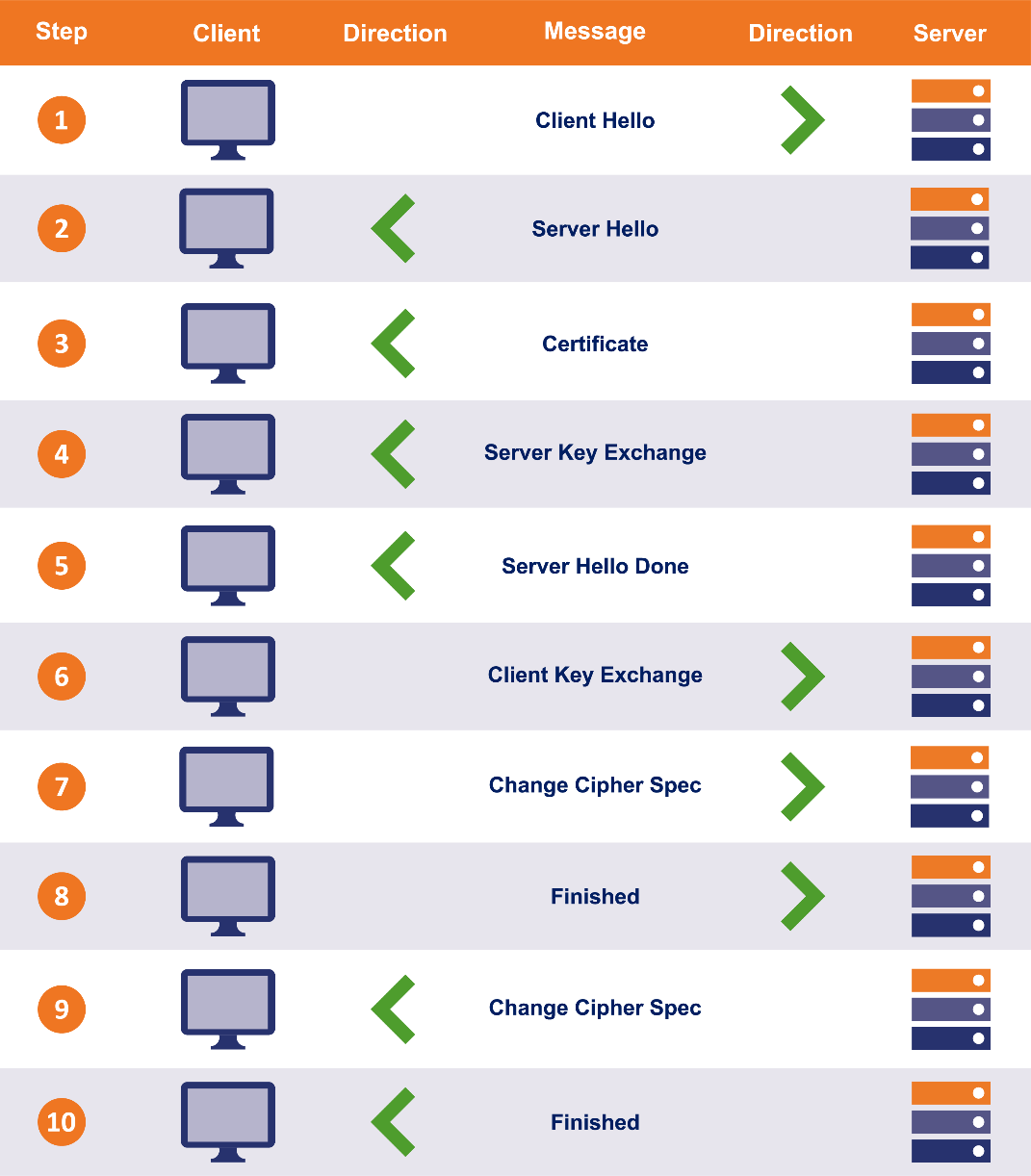
1. **Confidentiality**: Encrypts data to prevent unauthorized access.
2. **Integrity**: Ensures data has not been altered during transmission.
3. **Authentication**: Verifies the identity of the server, and optionally the client.

**How SSL/TLS Works:**

SSL/TLS secures communication through a series of steps that establish a **secure connection** before data exchange. These steps are known as the **SSL/TLS handshake**. The handshake is a process where the client and server agree on encryption methods and share keys for secure communication.  
 **Steps Involved in SSL/TLS Handshake:**

1. **Client Hello**:
   * The client (e.g., a web browser) sends a **Client Hello** message to the server.
   * The message includes:
     + SSL/TLS version the client supports.
     + A list of supported **cipher suites** (encryption algorithms).
     + A randomly generated value called the **Client Random**.
     + Optional information like session resumption data.
2. **Server Hello**:
   * The server responds with a **Server Hello** message.
   * It selects the SSL/TLS version and cipher suite from the list provided by the client.
   * It also generates and sends a **Server Random** value.
3. **Server Certificate**:
   * The server sends its **SSL/TLS certificate** to the client.
   * This certificate contains the server's **public key** and is signed by a trusted **Certificate Authority (CA)**.
   * The client uses this certificate to verify the server’s identity.
4. **Key Exchange**:
   * The client and server exchange information that allows both parties to generate a shared **session key**.
   * The session key is used to encrypt data exchanged during the session.
   * The key exchange can use algorithms like **RSA** (Rivest-Shamir-Adleman) or **Diffie-Hellman**.
5. **Client Key Exchange**:
   * The client generates a **Pre-master Secret** (a random number) and encrypts it using the server’s public key from the certificate.
   * This pre-master secret is sent to the server.
6. **Session Key Generation**:
   * Both the client and the server use the pre-master secret, along with the Client Random and Server Random values, to independently generate the same **session key**.
   * The session key is a symmetric key used for encrypting and decrypting the communication during the session.
7. **Change Cipher Spec**:
   * The client and server send a **Change Cipher Spec** message to inform each other that future messages will be encrypted using the session key.
8. **Finished**:
   * Both the client and the server send a **Finished** message, encrypted with the session key, to indicate that the handshake is complete.
   * If both messages are successfully decrypted and verified, the SSL/TLS connection is established.
9. **Encrypted Communication**:
   * After the handshake, the client and server use the session key to encrypt and decrypt all subsequent data exchanged during the session.
   * This ensures that sensitive data, such as login credentials, is transmitted securely.

**Summary of the SSL/TLS Handshake:**

1. **Client Hello** (Client proposes SSL/TLS version, cipher suite, and sends a random number)
2. **Server Hello** (Server selects version, cipher suite, and sends a random number)
3. **Server Certificate** (Server sends its certificate for client to verify)
4. **Key Exchange** (Client and server exchange information to generate a shared session key)
5. **Change Cipher Spec** (Both agree to switch to encrypted communication)
   1. **Finished** (Handshake is complete, and encrypted communication begins)  
        
      

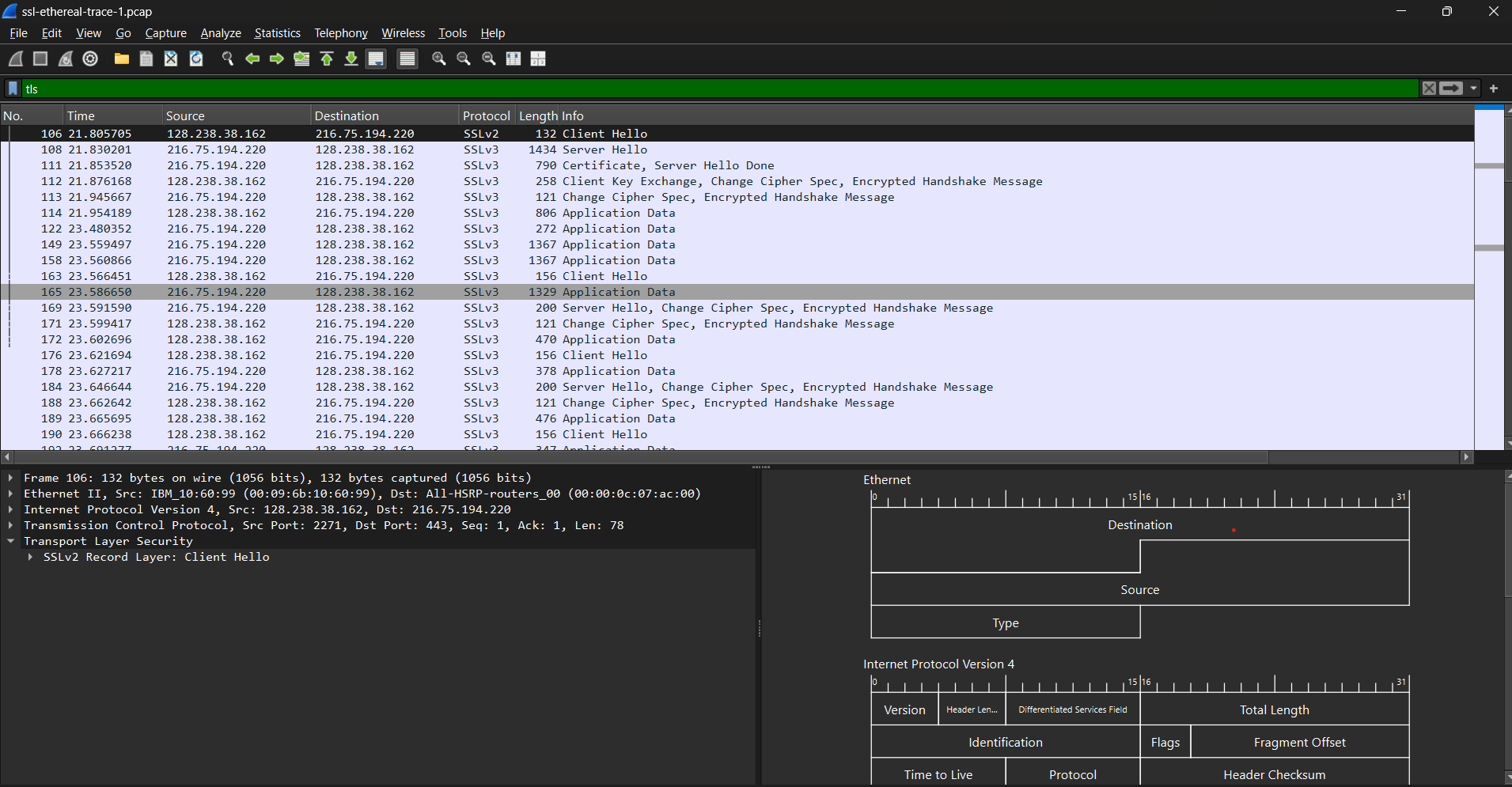
**Importance of SSL/TLS:**

* **Web Security**: SSL/TLS ensures that sensitive information (such as passwords, credit card details) is transmitted securely over the internet.
* **Authentication**: Verifies the identity of the website to prevent man-in-the-middle (MITM) attacks.
* **Data Privacy**: Prevents eavesdropping and tampering during communication.

In conclusion, SSL/TLS is crucial for ensuring secure and trusted communication over networks, particularly for websites and online services.

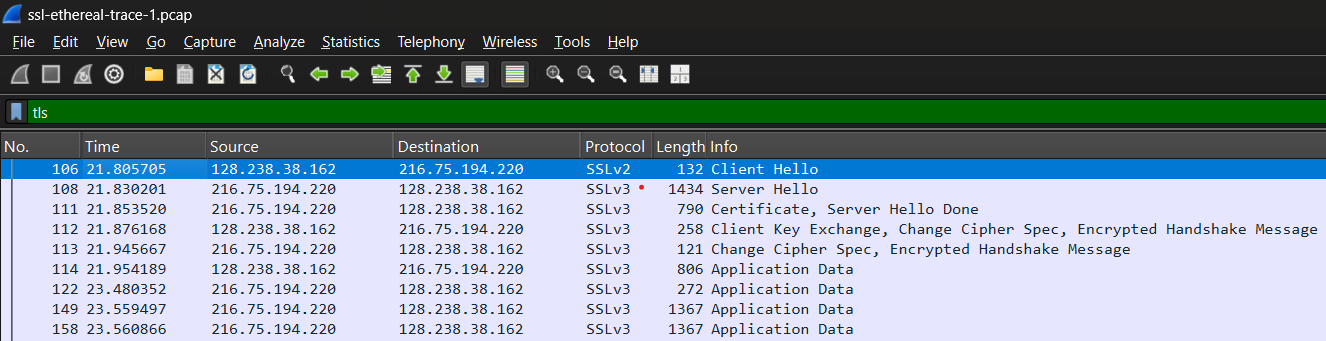
**Wireshark Lab: SSL v8.0**  
  
Download the zip file [**http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip**](http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip) and extract the ***ssl-ethereal trace-1*** packet trace.  
  
**1. Capturing packets in an SSL session**

It displays only the Ethernet frames that contain SSL records sent from and received by your host



An SSL record is the same thing as an SSL message.  
  
**2. A look at the captured trace**An Ethernet frame may contain one or more SSL records. (This is very different from HTTP, for which each frame contains either one complete HTTP message or a portion of a HTTP message.) Also, an SSL record may not completely fit into an Ethernet frame, in which case multiple frames will be needed to carry the record.

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.

**Ans:  
  
**

Frame 1: (Frame 106)

* Source: Client (128.238.38.162)
* Number of SSL Records: 1
* SSL Record Type: Client Hello (SSLv2)

Frame 2: (Frame 108)

* Source: Server (216.75.194.220)
* Number of SSL Records: 1
* SSL Record Type: Server Hello (SSLv3)

Frame 3: (Frame 111)

* Source: Server (216.75.194.220)
* Number of SSL Records: 2
* SSL Record Types:
  1. Certificate (SSLv3)
  2. Server Hello Done (SSLv3)

Frame 4: (Frame 112)

* Source: Client (128.238.38.162)
* Number of SSL Records: 3
* SSL Record Types:
  1. Client Key Exchange (SSLv3)
  2. Change Cipher Spec (SSLv3)
  3. Encrypted Handshake Message (SSLv3)

Frame 5: (Frame 113)

* Source: Server (216.75.194.220)
* Number of SSL Records: 2
* SSL Record Types:
  1. Change Cipher Spec (SSLv3)
  2. Encrypted Handshake Message (SSLv3)

Frame 6: (Frame 114)

* Source: Client (128.238.38.162)
* Number of SSL Records: 1
* SSL Record Type: Application Data (SSLv3)

Frame 7: (Frame 122)

* Source: Server (216.75.194.220)
* Number of SSL Records: 1
* SSL Record Type: Application Data (SSLv3)

Frame 8: (Frame 149)

* Source: Server (216.75.194.220)
* Number of SSL Records: 1
* SSL Record Type: Application Data (SSLv3)

2. 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.

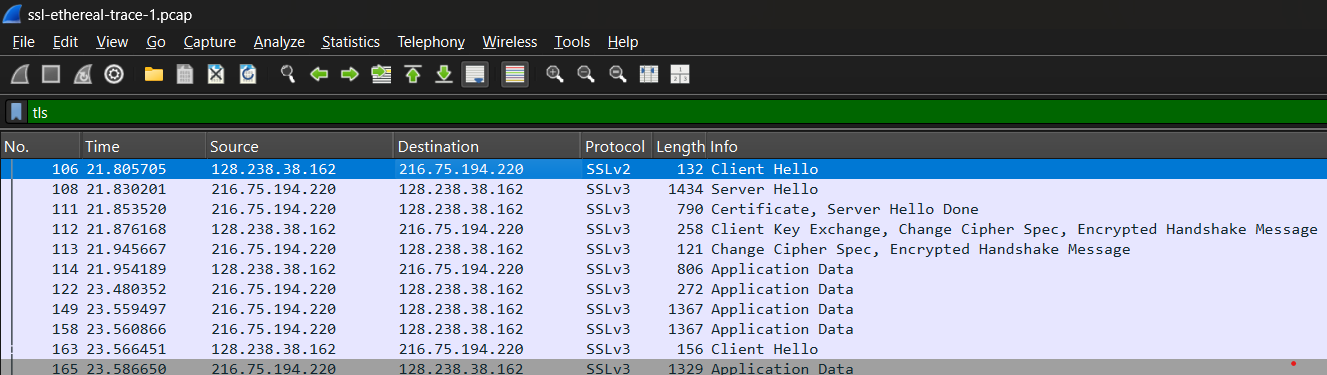
**Ans:**Each SSL record starts with the following three fields:

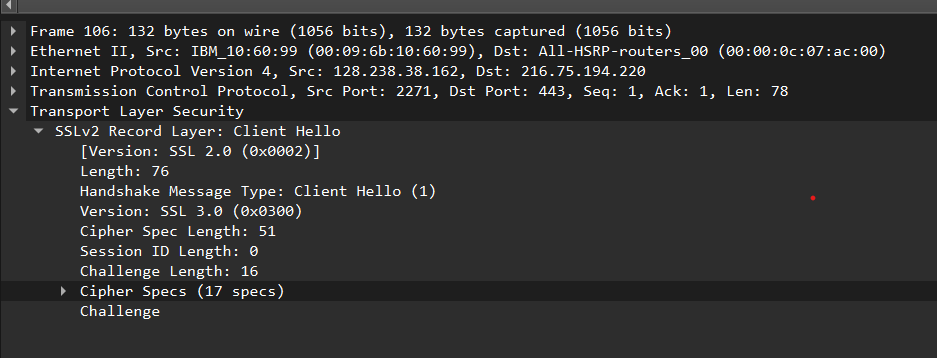
* **Content Type:** 1 byte
* **Version:** 2 bytes
* **Length:** 2 bytes

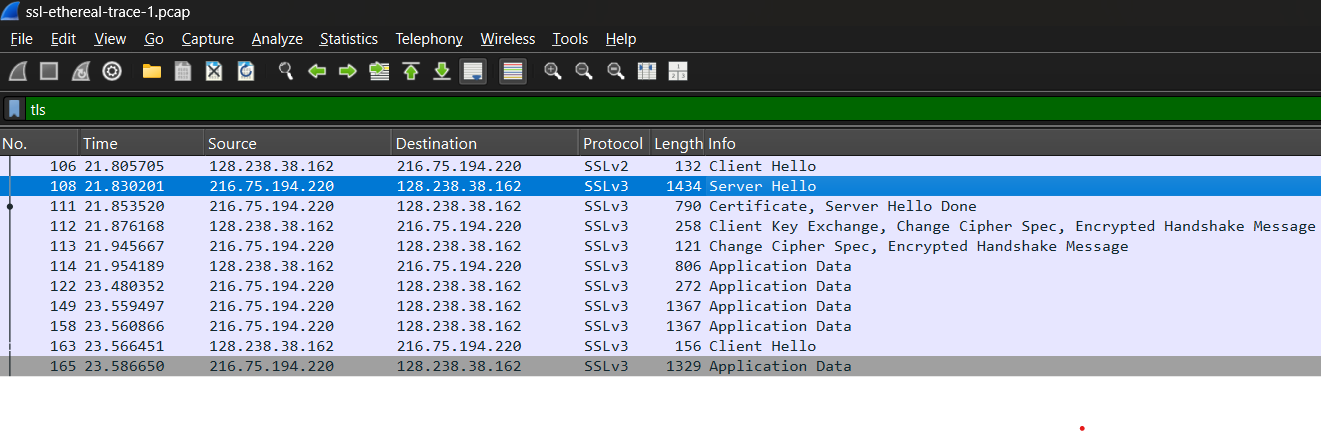
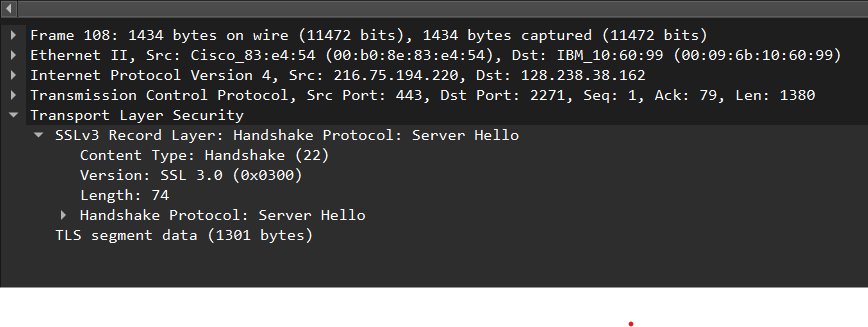
**How to Find These Fields:**

If you are using packet capture software like **Wireshark**, you can find these fields in the packet capture by:

1. **Open Wireshark** and load the captured SSL/TLS packet data (the one you listed).
2. **Select an SSL/TLS packet** from the list and expand the **"Secure Sockets Layer"** or **"Transport Layer Security"** section in the detailed packet view.
3. You will see the **Record Layer** header information, where these fields will be listed:
   * **Content Type:** Displays the type of SSL/TLS record (Handshake, Application Data, etc.)
   * **Version:** The protocol version (e.g., TLS 1.2)
   * **Length:** The size of the encrypted data.

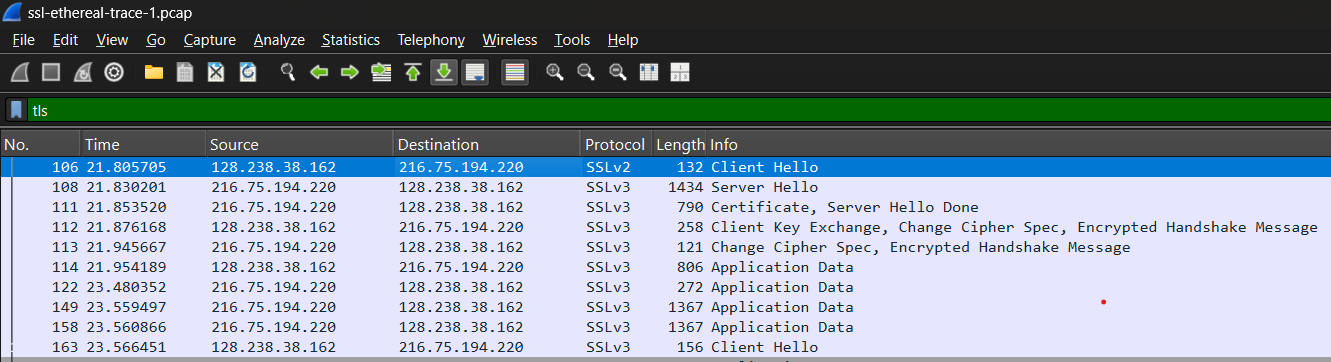
**Selecting Client Hello packet:**  


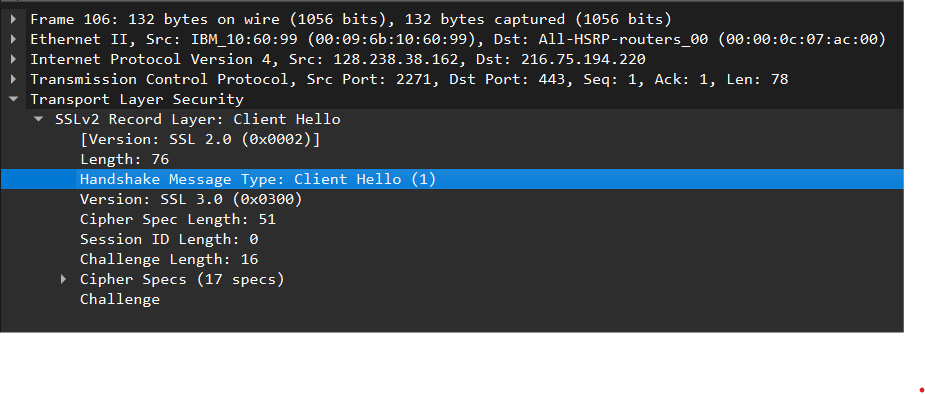
****

**Selecting Server Hello packet:  
  
  
  
**

**ClientHello Record:**

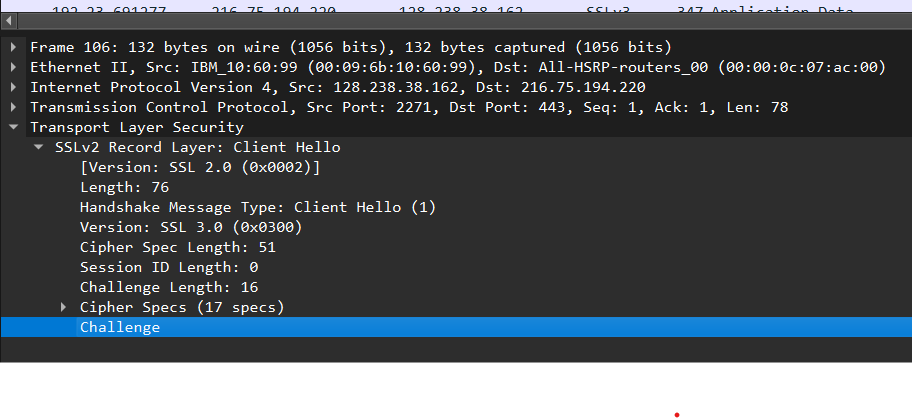
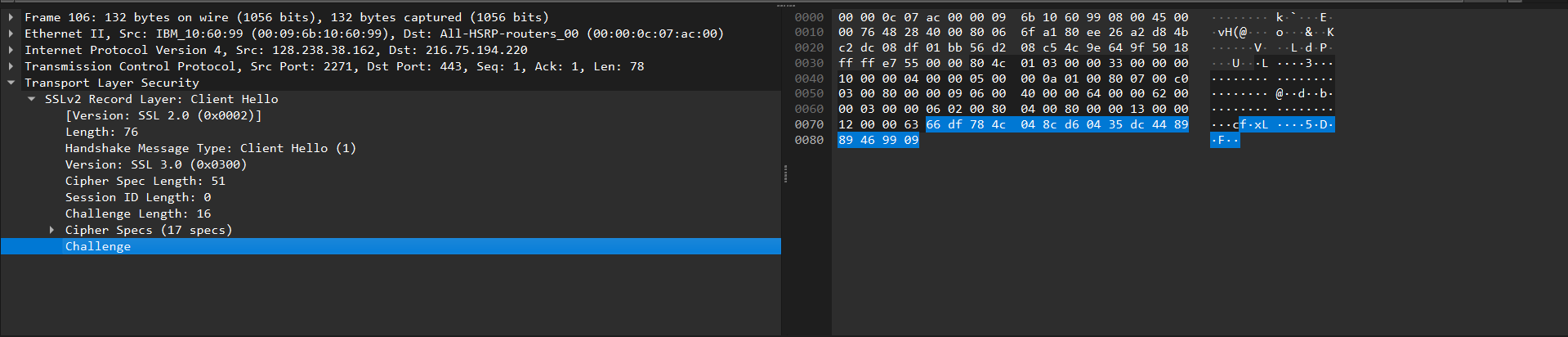
3. Expand the ClientHello record. (If your trace contains multiple ClientHello records, expand the frame that contains the first one.) What is the value of the content type?  
  
**Ans:**  
  
The **ClientHello** record in **Frame 106** is an SSLv2 message with a handshake message type of **Client Hello (1)**.

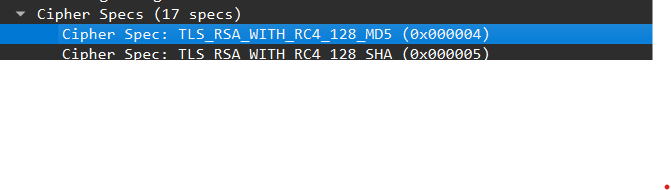




4. Does the ClientHello record contain a nonce (also known as a “challenge”)? If so, what is the value of the challenge in hexadecimal notation?

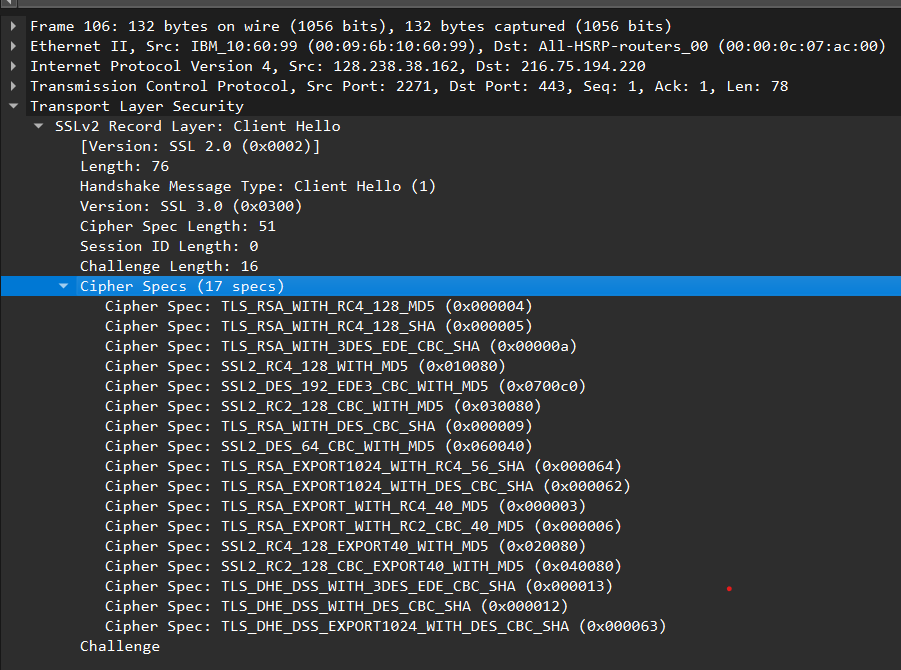
**Ans:**Yes, the ClientHello record contains a nonce (also known as a “challenge”).

****

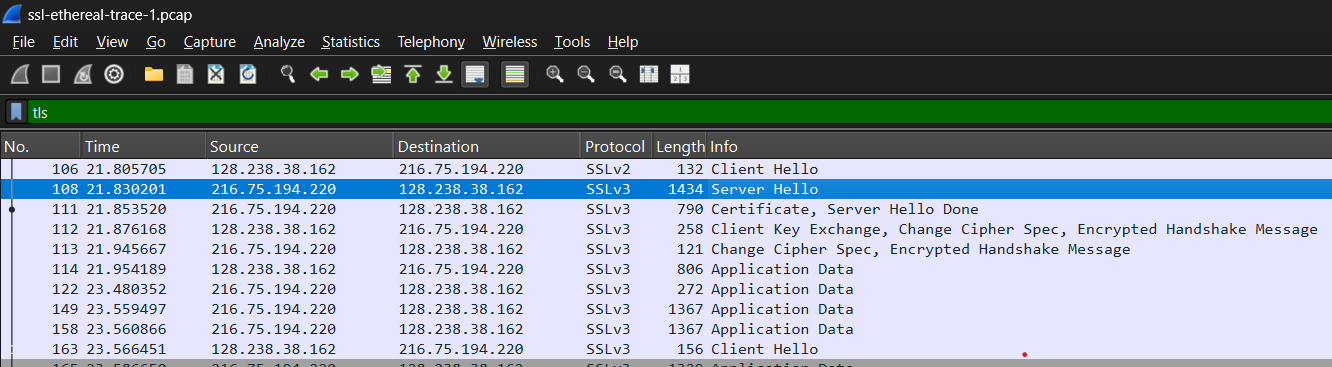
5. Does the ClientHello record advertise the cyber suites it supports? If so, in the first listed suite, what are the public-key algorithm, the symmetric-key algorithm, and the hash algorithm?   
  
Ans:  
  
Yes, the ClientHello record does advertise the cipher suites it supports.   
  
  
  
In the first listed cipher suite, which is **TLS\_RSA\_WITH\_RC4\_128\_MD5 (0x000004)**, the following algorithms are used:

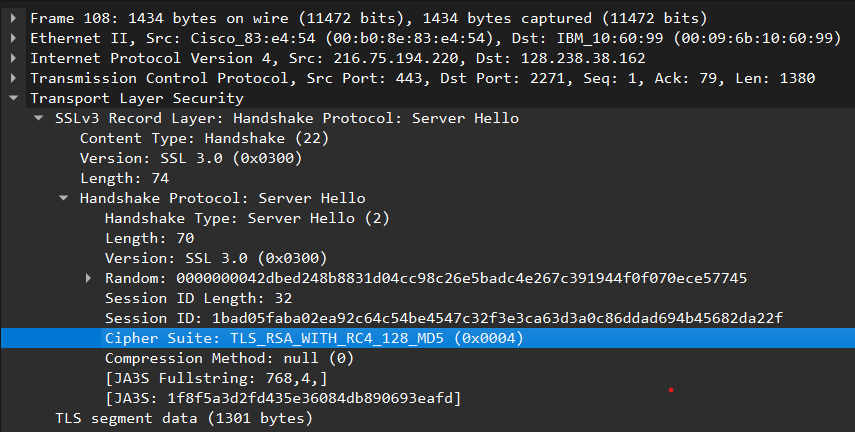
* **Public-key algorithm**: RSA
* **Symmetric-key algorithm**: RC4 (with a key length of 128 bits)
* **Hash algorithm**: MD5

This combination indicates that the client supports this suite for secure communication.



**ServerHello Record:**

6. Locate the ServerHello SSL record. Does this record specify a chosen cipher suite? What are the algorithms in the chosen cipher suite?  
  
**Ans:**  
  
 



Yes, the ServerHello SSL record specifies a chosen cipher suite. The chosen cipher suite is TLS\_RSA\_WITH\_RC4\_128\_MD5 (0x0004).

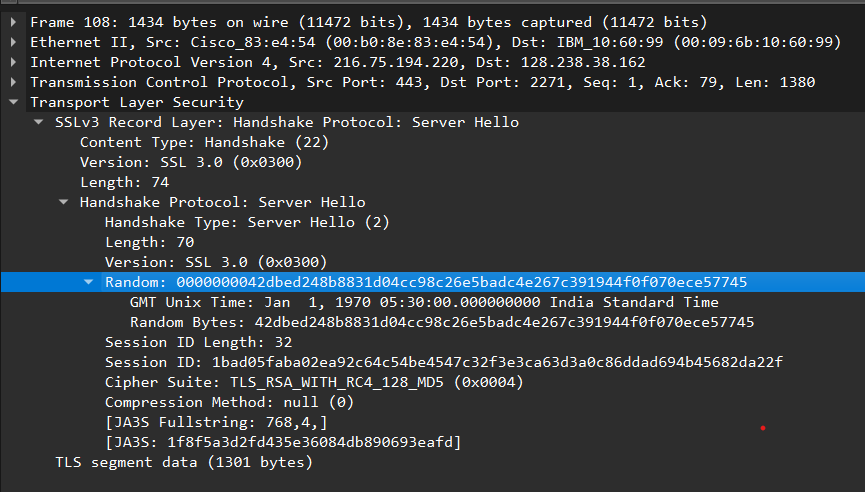
The algorithms in this chosen cipher suite are:

* **Public-key algorithm**: RSA
* **Symmetric-key algorithm**: RC4 (with a key length of 128 bits)
* **Hash algorithm**: MD5

This indicates the server's selected encryption method for the session.

7. Does this record include a nonce? If so, how long is it? What is the purpose of the client and server nonces in SSL?   
  
**Ans:**  
  
**Locate the Nonce**:

* The **ServerHello** response may not explicitly list a nonce like the **ClientHello** does, but it usually includes a **Session ID** and potentially a **Server Random** value (which acts similarly to a nonce).
* Look for fields labeled **Session ID Length**, **Session ID**, and **Random**.

  
  
  
Yes, the ServerHello record includes a nonce, which is referred to as the "Random" value. In this case, the nonce is 32 bytes long (256 bits).

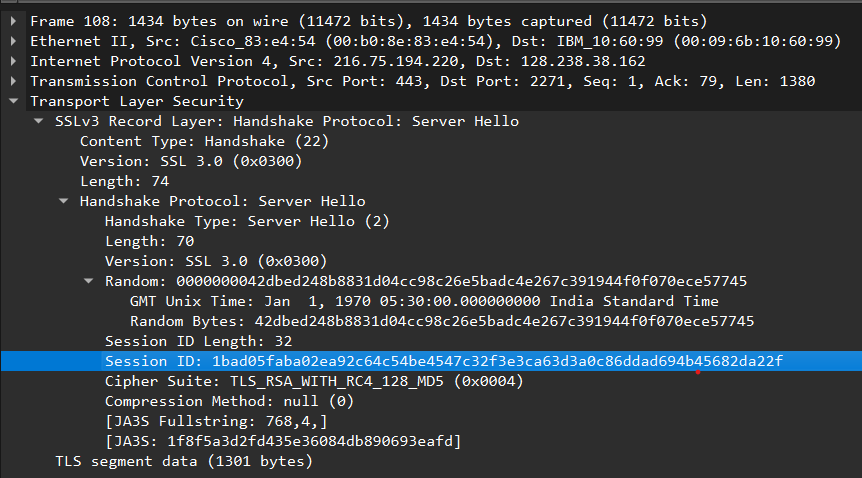
**Purpose of Nonces in SSL:**

1. **Prevent Replay Attacks**: Nonces ensure that each session is unique, preventing attackers from reusing old messages to impersonate a user or a session.
2. **Key Generation**: Nonces are used in the key generation process during the SSL handshake. They contribute to creating session keys that are unique for each session, ensuring that even if the same keys were used in different sessions, the actual keys derived will be different due to the unique nonces.

By using nonces, SSL enhances the security and integrity of the communication between the client and server.

**Purpose of Nonce in the ServerHello Record:**

1. **Session Uniqueness**:
   * Similar to the **ClientHello**, the **Server Random** value helps ensure that the session is unique. It differentiates this session from previous ones.
2. **Key Derivation**:
   * The **Server Random** value is combined with the **Client Random** value (from the **ClientHello**) during the key derivation process to create session keys for encrypting the data exchanged in the session. This ensures that the keys are unique for each session.
3. **Preventing Replay Attacks**:
   * Just as with the client, the server's nonce (or **Server Random**) helps protect against replay attacks, ensuring that each session is independent and cannot be reused maliciously.

8. Does this record include a session ID? What is the purpose of the session ID?   
  
**Ans:  
  
**  
Yes, the ServerHello record includes a session ID, which has a length of 32 bytes in this case.

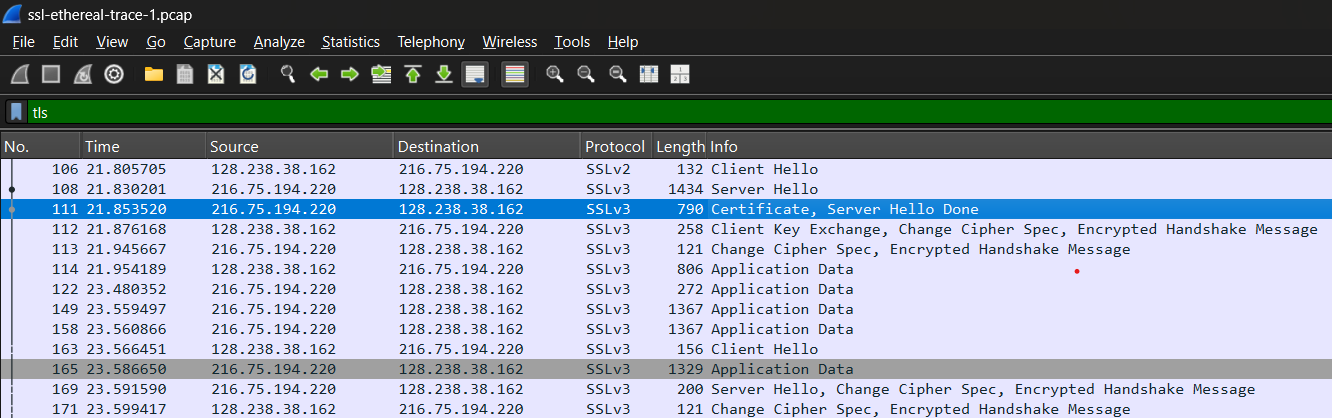
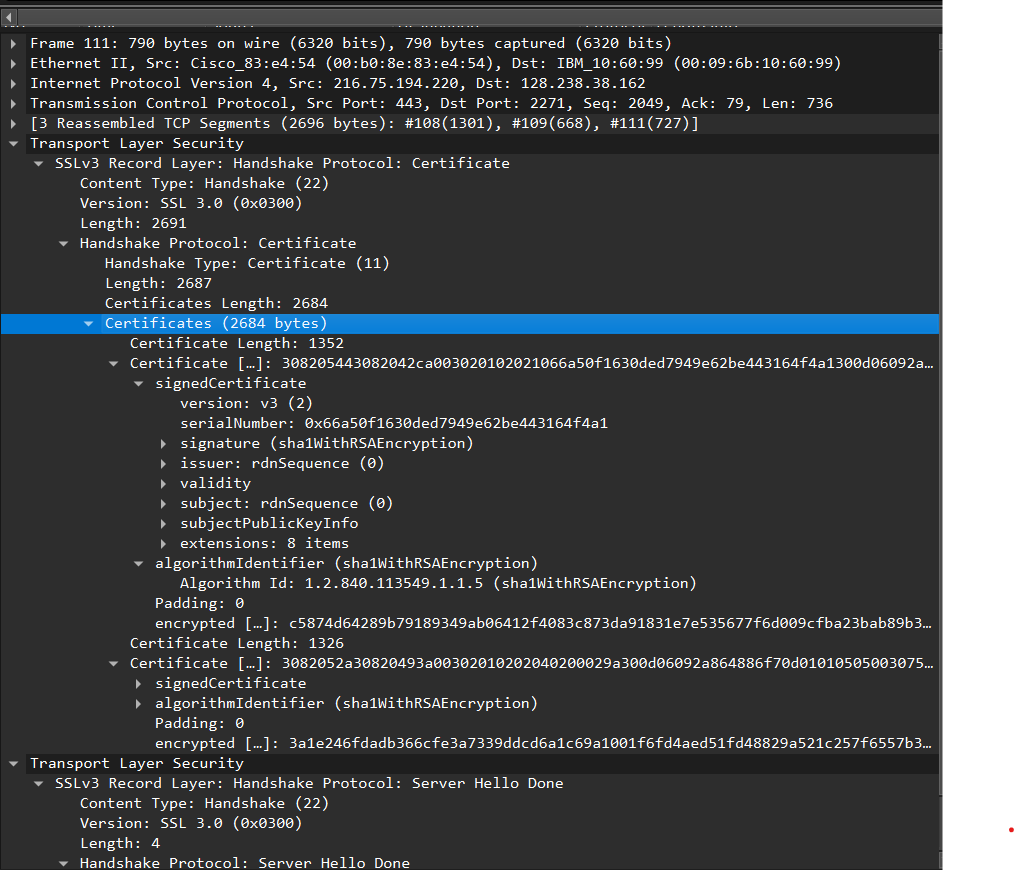
**Purpose of the Session ID:**

1. **Session Resumption:** The session ID allows clients and servers to resume a previous SSL/TLS session without needing to perform a full handshake again. This speeds up the connection process for subsequent sessions between the same client and server.
2. **State Management:** The session ID helps manage session states on the server. It allows the server to recognize and retrieve session parameters (like the cipher suite and keys) associated with that ID, facilitating quicker reconnections.
3. **Security:** By using a session ID, SSL/TLS can provide continuity and consistency for secure communications, ensuring that established session parameters are reused securely.

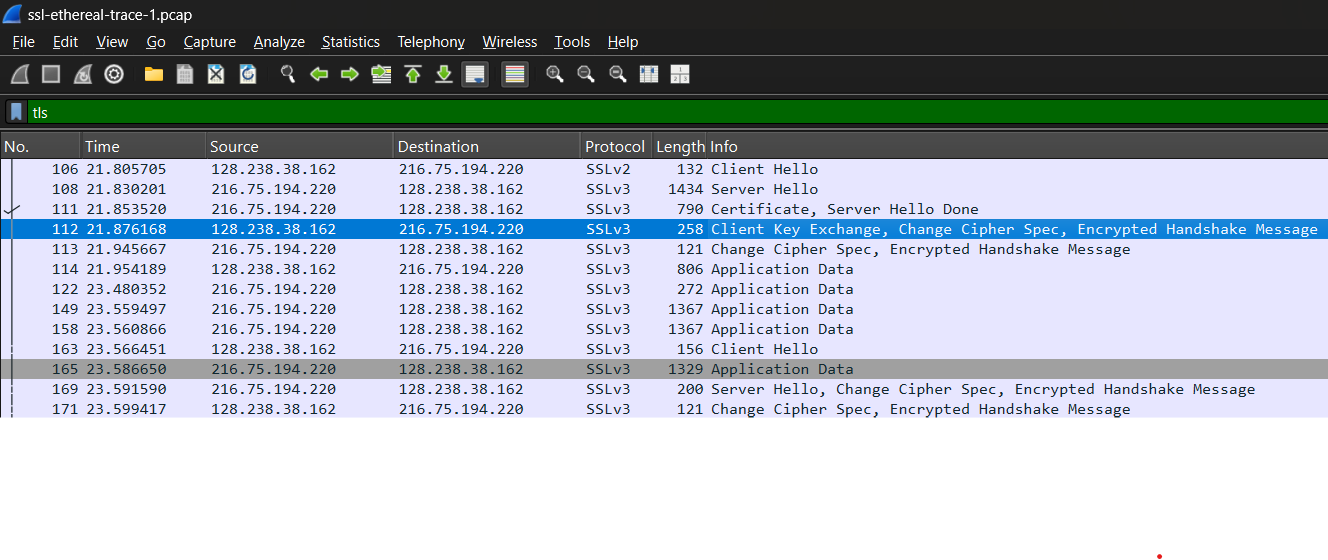
Overall, the session ID is crucial for improving performance and maintaining security during repeated connections.

9. Does this record contain a certificate, or is the certificate included in a separate record. Does the certificate fit into a single Ethernet frame?   
  
**Ans:**  
  
The ServerHello record itself does not contain a certificate; certificates are included in a separate record, specifically in the Certificate message that follows the ServerHello during the handshake process.

Regarding the size of the certificate, whether it fits into a single Ethernet frame depends on the actual size of the certificate being transmitted. Ethernet frames typically have a maximum payload size of around 1500 bytes. If the certificate's size is within this limit, it can fit into a single frame; otherwise, it will be fragmented across multiple frames.

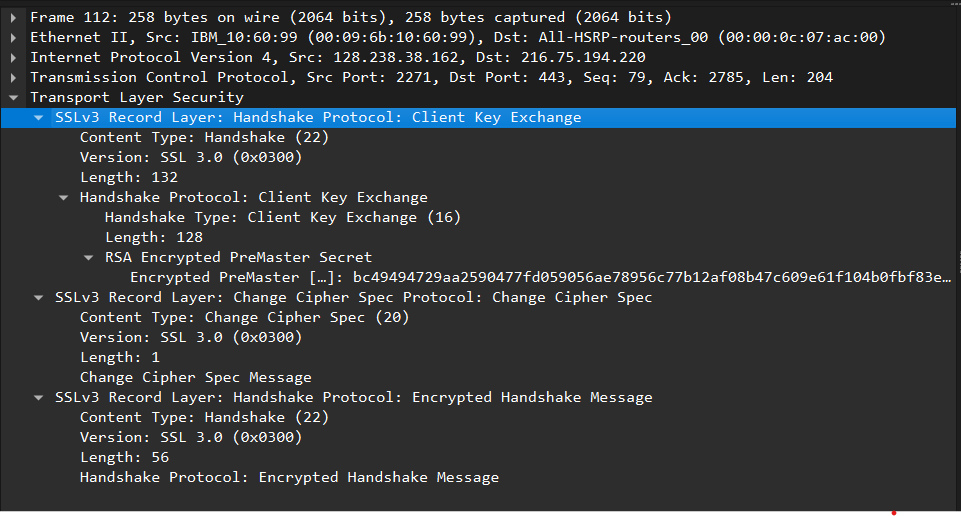
  
  


**Client Key Exchange Record:**  
10. Locate the client key exchange record. Does this record contain a pre-master secret? What is this secret used for? Is the secret encrypted? If so, how? How long is the encrypted secret?   
**Ans:**

****  
  
Yes, the Client Key Exchange record contains a pre-master secret. This secret is used to derive the session keys for encryption and integrity in the SSL/TLS session.

**Details:**

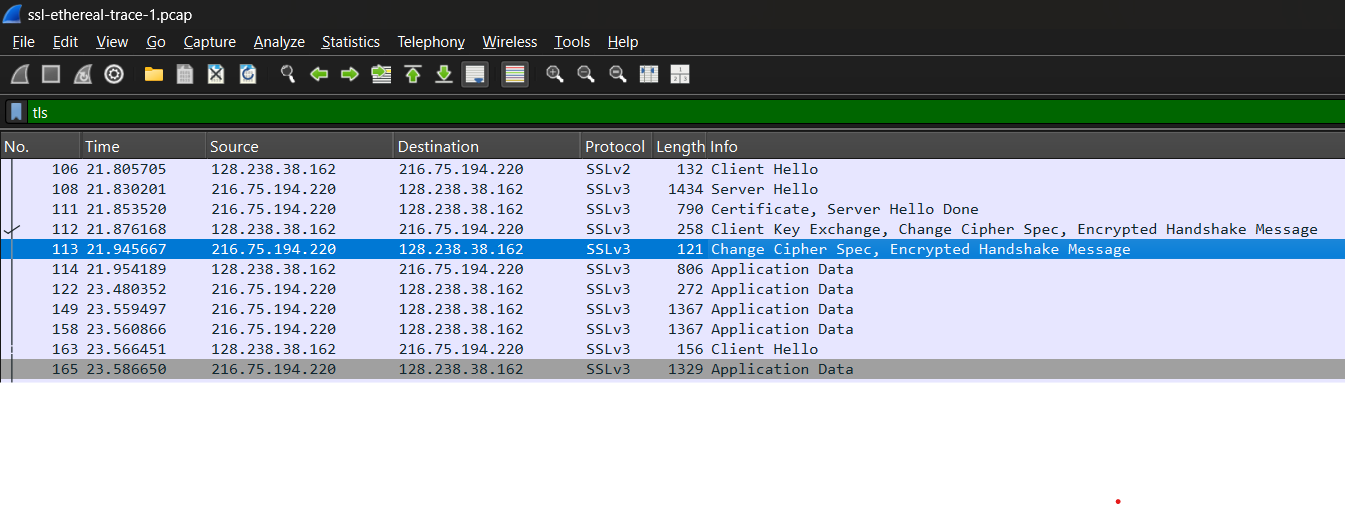
1. **Use of the Pre-Master Secret**: The pre-master secret is combined with the client and server nonces to generate the session keys used for symmetric encryption and message authentication during the session.
2. **Encryption of the Pre-Master Secret**: The pre-master secret is encrypted using RSA encryption. This ensures that only the intended recipient (the server) can decrypt it using its private key.
3. **Length of the Encrypted Secret**: The length of the encrypted pre-master secret is 128 bytes (1024 bits). This length corresponds to the size of the RSA-encrypted data, which typically is larger than the original pre-master secret due to the encryption overhead.

In summary, the Client Key Exchange record does contain an encrypted pre-master secret, which is essential for establishing secure communication between the client and server.  


**Change Cipher Spec Record (sent by client) and Encrypted Handshake Record:**

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

**Ans:**

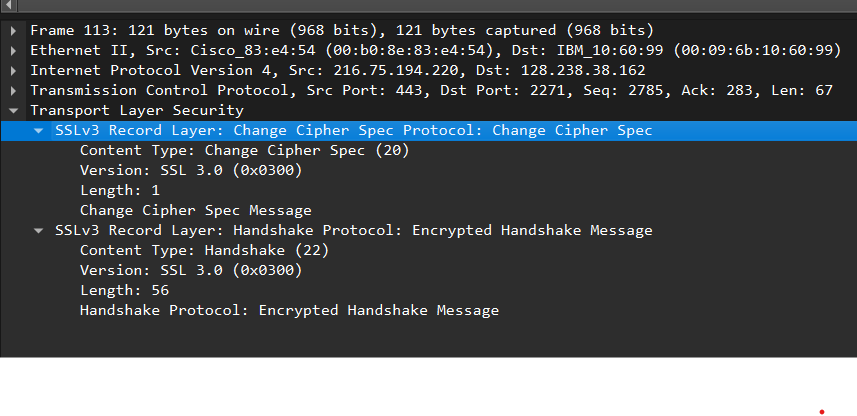


**Purpose**: The Change Cipher Spec record indicates that the sender is ready to switch to encrypted communication using the new cipher suite and keys.

The **Change Cipher Spec** record signals that the sender will start using the newly negotiated cipher suite and keys for subsequent messages. This record is part of the SSL/TLS handshake process and indicates that the sender is ready to switch from using the previous cipher settings to the new ones established during the handshake.

**Size:** The **Change Cipher Spec record** in above trace is **121 bytes** in total. This includes the Change Cipher Spec message itself, which is 1 byte, plus additional bytes related to the transmission (such as the Ethernet and IP headers).

This transition ensures that the communication is now encrypted using the new parameters established during the handshake.



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

**Ans:**In the Encrypted Handshake Message record, the **data being encrypted is the handshake messages that were exchanged during the SSL/TLS handshake process after the Client Key Exchange**. This typically includes messages such as the ServerHelloDone message and any other necessary messages required to complete the handshake.

**How is it encrypted?**

1. Encryption Method: The messages are encrypted using the symmetric encryption algorithm specified by the negotiated cipher suite (in this case, TLS\_RSA\_WITH\_RC4\_128\_MD5).
2. Keying Material: The encryption uses the session keys derived from the pre-master secret, along with the client and server nonces. This ensures that the data can only be decrypted by the intended recipient (the server in this case) who possesses the corresponding session key.
3. Integrity Check: Additionally, a Message Authentication Code (MAC) is often applied to ensure the integrity and authenticity of the messages, helping to prevent tampering.

Using symmetric-key algorithms determined by the negotiated cipher suite, leveraging session keys derived from the pre-master secret. The messages are encrypted and often accompanied by a MAC for integrity and authenticity.

In summary, the Encrypted Handshake Message record contains encrypted handshake messages that are protected using the newly established symmetric keys and algorithms.

13. 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?

**Ans:**Yes, the server also sends a Change Cipher Spec record and an Encrypted Handshake Message record to the client. Here’s how these records differ from those sent by the client:

Change Cipher Spec Record:

* Client: The Change Cipher Spec record sent by the client indicates that it is ready to switch to the new cipher suite and keys established during the handshake.
* Server: The server sends a Change Cipher Spec record to notify the client that it is also switching to the new cipher settings. The purpose is the same: to confirm that the server will now use the new encryption parameters.

Encrypted Handshake Message:

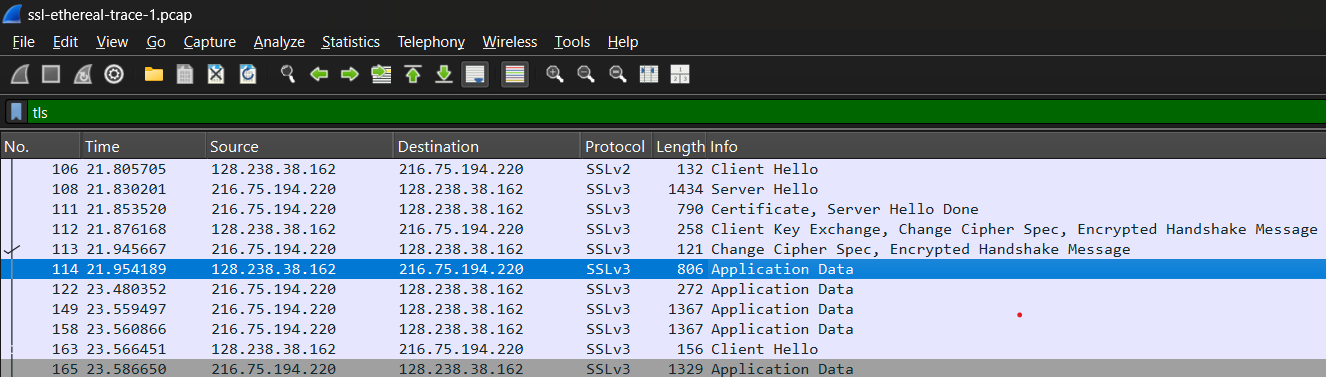
* Client: The Encrypted Handshake Message sent by the client contains handshake messages that have been encrypted using the negotiated cipher suite and session keys.
* Server: The server sends its own Encrypted Handshake Message that contains its handshake messages (such as ServerHelloDone) encrypted in the same way.

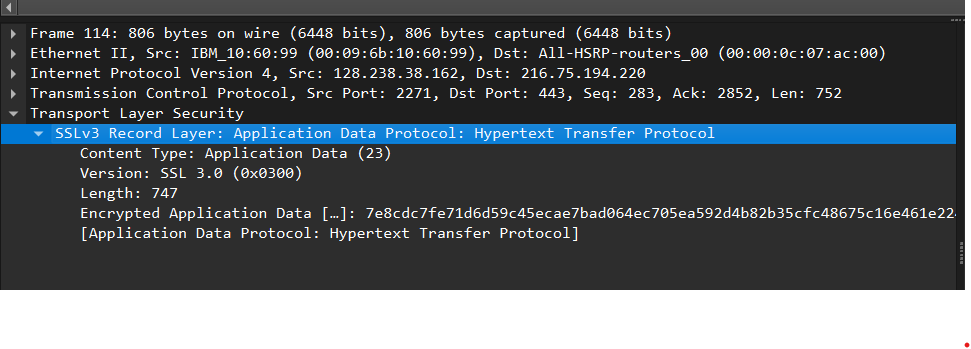
Key Differences:

1. Content: The content of the records differs. The client's records will contain its specific handshake messages, while the server's records will contain its own messages.
2. Direction: The client’s Change Cipher Spec and Encrypted Handshake records are sent from the client to the server, while the server’s records are sent in the opposite direction.

Overall, while the structure and purpose of the Change Cipher Spec and Encrypted Handshake Message records are consistent between client and server, the actual content and direction of communication differ.

**Application Data:**

14. 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?   
  
**Ans:**  


  
  
  
**How is the Application Data being Encrypted?**

The application data is encrypted using the symmetric encryption algorithm specified by the negotiated cipher suite (in this case, likely TLS\_RSA\_WITH\_RC4\_128\_MD5). The encryption is done using the session keys derived during the handshake process, which are based on the pre-master secret and the nonces.

**Does the Record Containing Application Data Include a MAC?**

Yes, records containing application data include a Message Authentication Code (MAC). The MAC ensures the integrity and authenticity of the data, protecting it from tampering and ensuring that it comes from a legitimate source.

**Does Wireshark Distinguish Between the Encrypted Application Data and the MAC?**

Wireshark typically does not display the MAC as a separate field in the decrypted view of the application data. Instead, the MAC is included in the encrypted payload. When the encrypted application data is analyzed, Wireshark shows it as a single encrypted block. The MAC is calculated over the plaintext data and is used during decryption to verify the integrity of the received data.

In summary, the application data is encrypted with session keys, includes a MAC for integrity, and Wireshark displays the data as a single encrypted entity without separating the MAC in its output.

15. Comment on and explain anything else that you found interesting in the trace.  
  
**Ans:  
  
Use of Different SSL Versions**:

The trace indicates a transition from SSLv2 to SSLv3. It's interesting to note the evolution of the SSL protocol versions, as SSLv2 is considered outdated and insecure. Modern applications primarily use TLS, which is the successor to SSL. The presence of SSLv2 could indicate compatibility settings or legacy systems.

**Cipher Suite Negotiation**:

The ClientHello message lists multiple cipher suites supported by the client. The server chooses one from this list for the session, which can reveal insights into the security posture and configurations of both the client and server. Observing this negotiation process can be critical for understanding potential vulnerabilities.

**Challenge and Nonce Usage**:

The ClientHello message includes a nonce (challenge), which is a random value used to prevent replay attacks. This is an interesting feature of SSL/TLS that enhances security by ensuring that each session is unique. The presence of nonces shows the protocols' design to handle specific security threats effectively.

**Certificate Exchange**:

The certificate exchange step during the ServerHello message and subsequent records is crucial for establishing trust. This trace shows the server providing its certificate, which may be signed by a trusted Certificate Authority (CA). The ability to verify this certificate is essential for the client to ensure that it is communicating with the legitimate server.

**Packet Sizes and Performance**:

Analyzing the sizes of the packets in the trace could provide insights into network performance. Larger packets may indicate bulk data transfers, while smaller packets might signify many small transactions. Identifying patterns in packet sizes could help in optimizing application performance and network resource utilization.

**Timing of Records**:

Observing the timing between records can provide insights into latency and performance issues. For example, if there are significant delays between the ClientHello and ServerHello messages, it could indicate network congestion or processing delays.

**Application Data Records**:

The presence of application data records after the handshake signals that secure communication has commenced. Analyzing the types of application data exchanged can provide insights into the nature of the application traffic, whether it's HTTP requests, file transfers, etc.

**Network Security Considerations**:

The trace can help identify potential security concerns, such as unencrypted traffic, or weak cipher suites. It is important to ensure that strong encryption practices are followed, as vulnerabilities in these areas could lead to exposure of sensitive data.

These points provide a deeper understanding of the SSL handshake process and the resulting secure communication, illustrating both the complexity and importance of cryptographic protocols in modern network security.