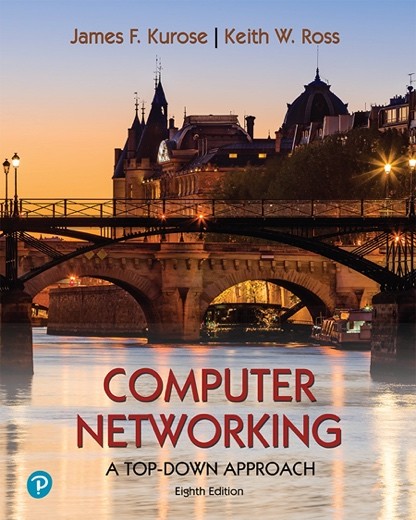
**Final Year B.Tech. (CSE) – VII [ 2024-25]**

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

**Date: 21/10/2024**

**Assignment10**

**PRN:** 21510042  **Name:** Omkar Rajesh Auti

Wireshark Lab:

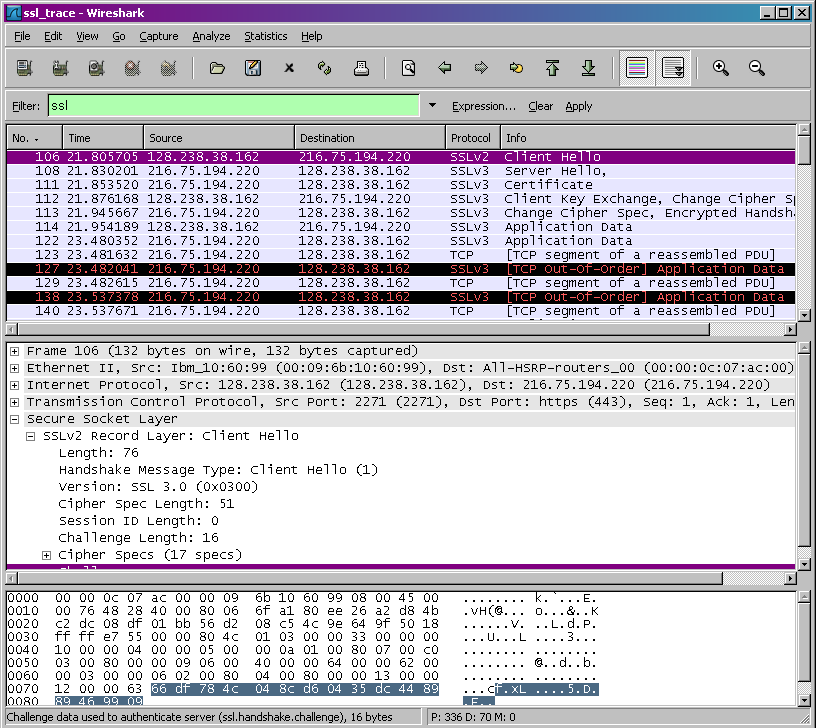
SSL v8.0

Supplement to *Computer Networking: A Top-Down Approach, 8th ed.,* J.F. Kurose and K.W. Ross

*“Tell me and I forget. Show me and I remember. Involve me and I understand.”* Chinese proverb

© 2005-2020, J.F Kurose and K.W. Ross, All Rights Reserved

In this lab, we will investigate the Secure Sockets Layer (SSL) protocol, focusing on the SSL records sent over a TCP connection. We will do so by analyzing a trace of the SSL records sent between your host and an e-commerce server. We will investigate the various SSL record types as well as the fields in the SSL messages.



# Capturing packets in an SSL session

The first step is to capture the packets in an SSL session. To do this, you should go to your favorite e-commerce site and begin the process of purchasing an item (but terminating before making the actual purpose!). After capturing the packets with Wireshark, you should set the filter so that 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.) You should obtain something like screenshot on the previous page.

If you have difficulty creating a trace, you should download the zip file <http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip>and extract the *ssl-ethereal- trace-1* packet trace.

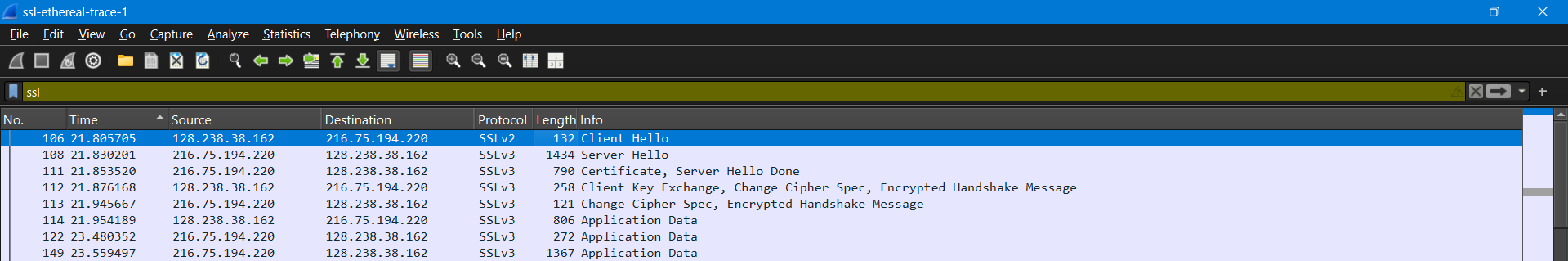
# A look at the captured trace

Your Wireshark GUI should be displaying only the Ethernet frames that have SSL records. It is important to keep in mind that 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.

Whenever possible, when answering a question below, you should hand in a printout of the packet(s) within the trace that you used to answer the question asked. Annotate the printout2 to explain your answer. To print a packet, use *File->Print*, choose *Selected packet only*, choose *Packet summary line,* and select the minimum amount of packet detail that you need to answer the question

* 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.



first 8 Ethernet frames to determine:

1. Source (client or server)
2. Number of SSL records in the frame
3. SSL record types

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)
  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.

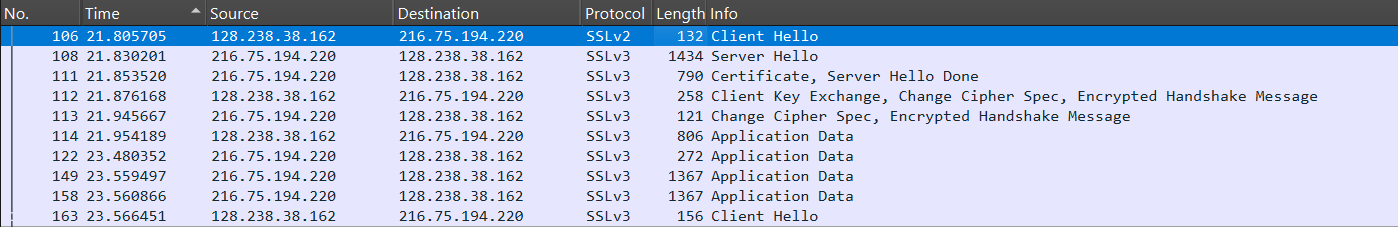
Each SSL record starts with the following three fields:

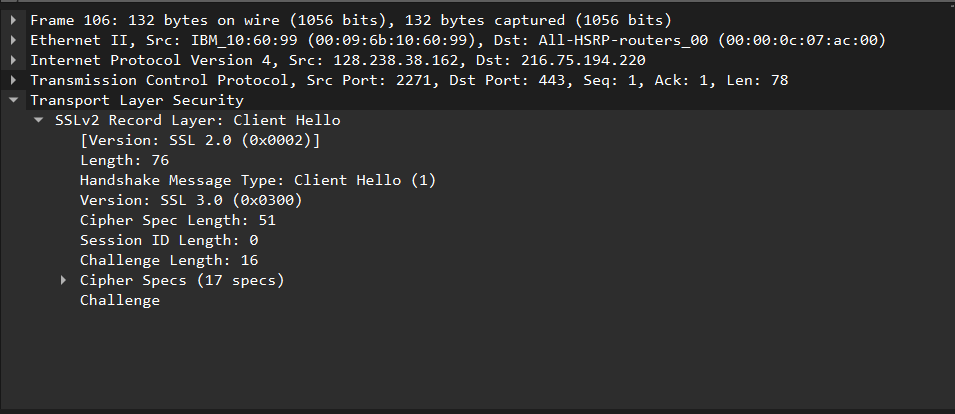
* **Content Type:** 1 byte
* **Version:** 2 bytes
* **Length:** 2 bytes

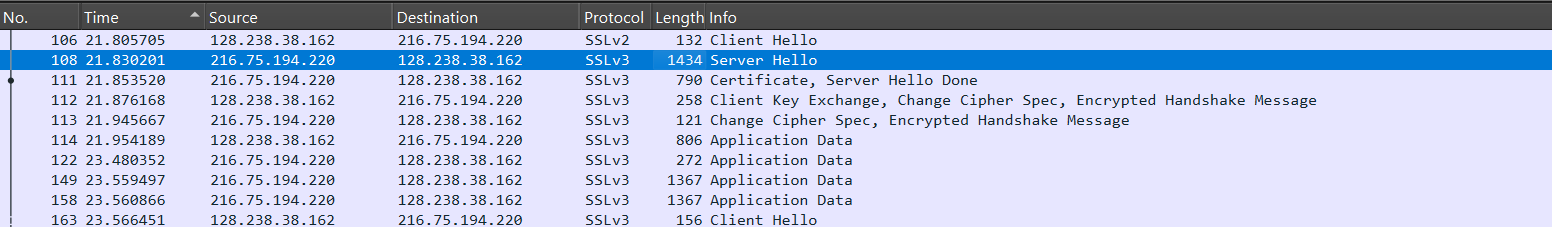
**How to Find These Fields:**

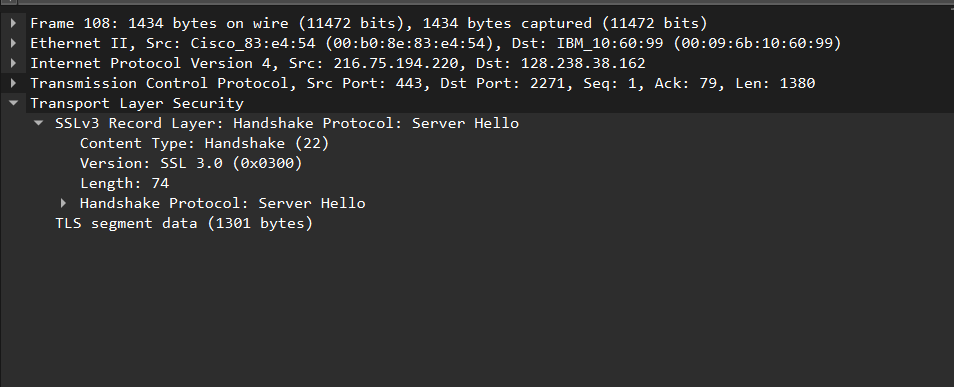
If you are using packet capture software like **Wireshark**, you can find these fields in the packet capture by:

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.





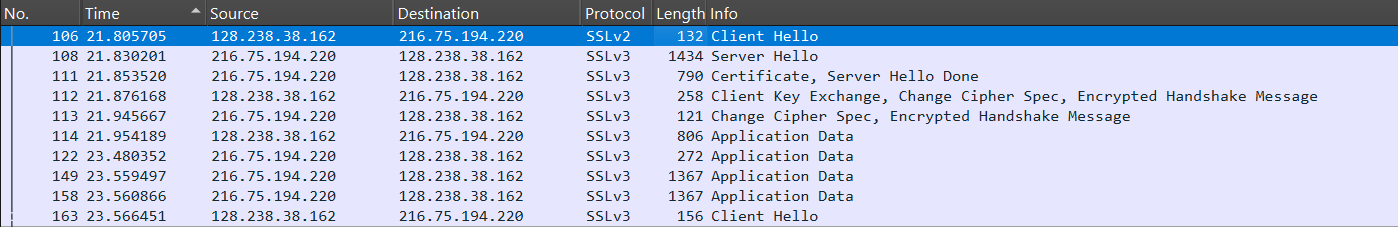


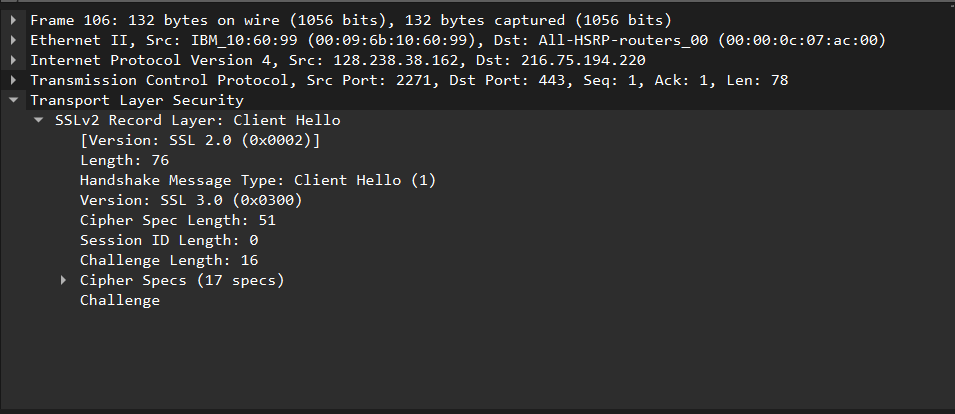


ClientHello Record:

* 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?

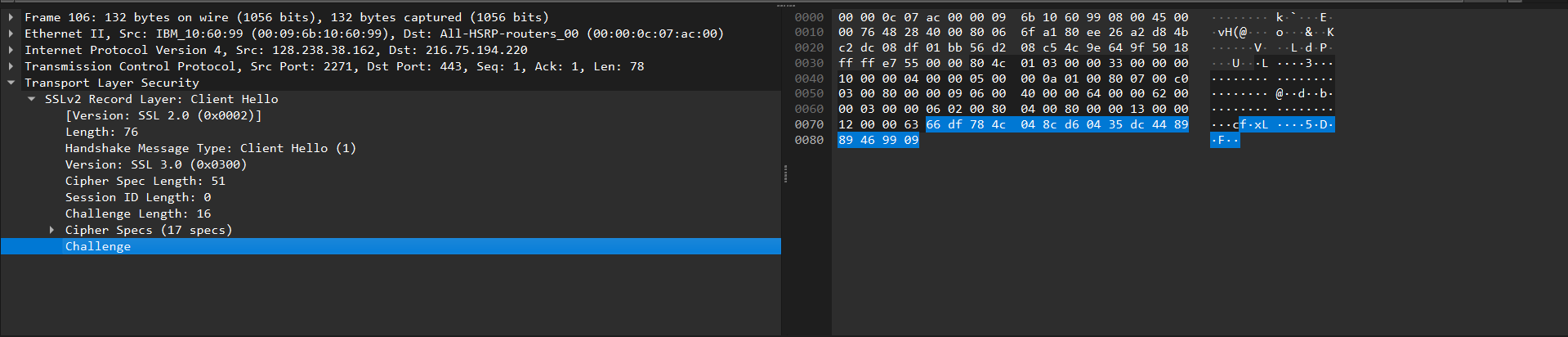
The **ClientHello** record in **Frame 106** is an SSLv2 message with a handshake message type of **Client Hello (1)**.





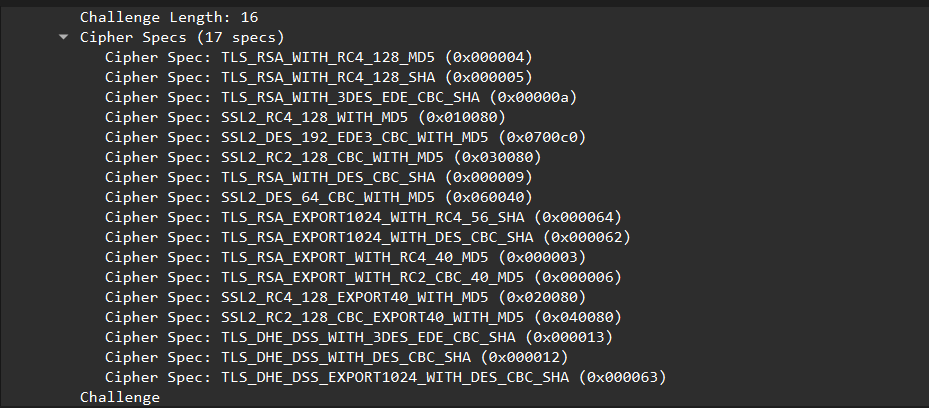
* 1. Does the ClientHello record contain a nonce (also known as a “challenge”)? If so, what is the value of the challenge in hexadecimal notation?

Answer: YES

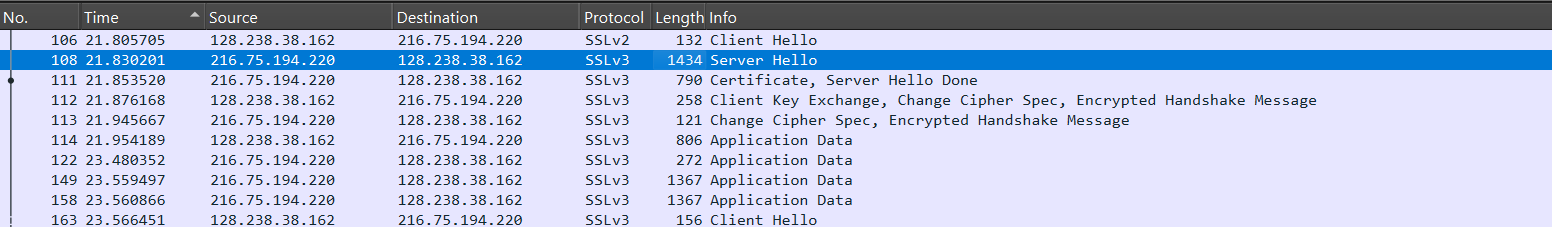


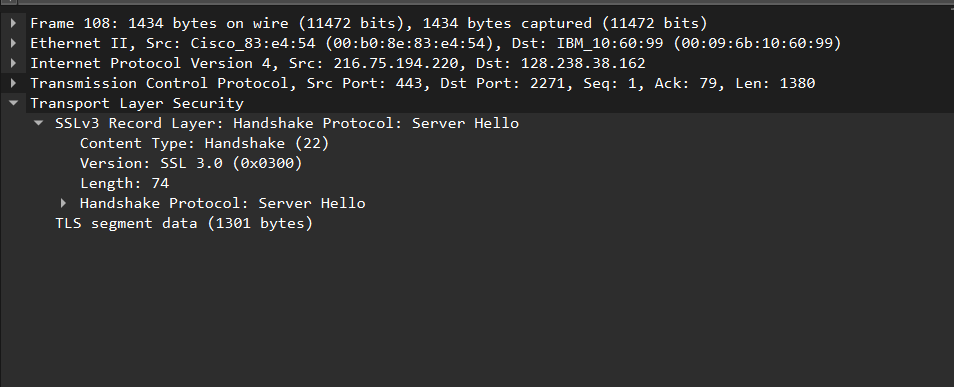
* 1. Does the ClientHello record advertise the cyber suites it supports? If so, in the first listed suite, what are the public-key algorithm, the symmetric-key algorithm, and the hash algorithm?

Answer: YES



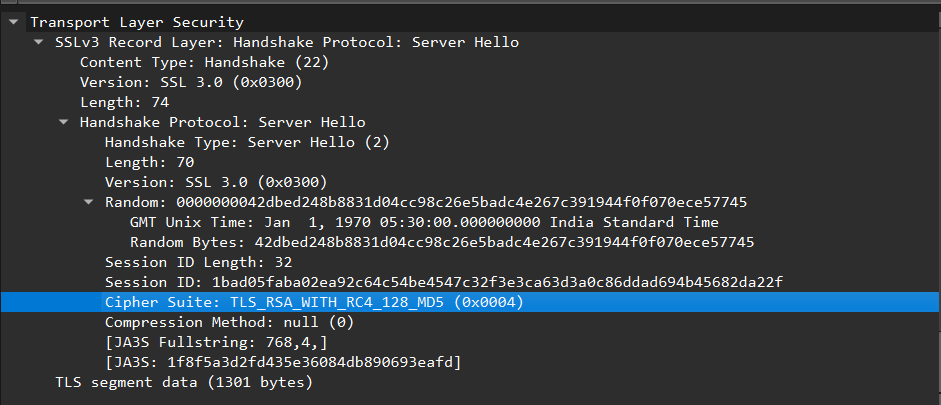
ServerHello Record:





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

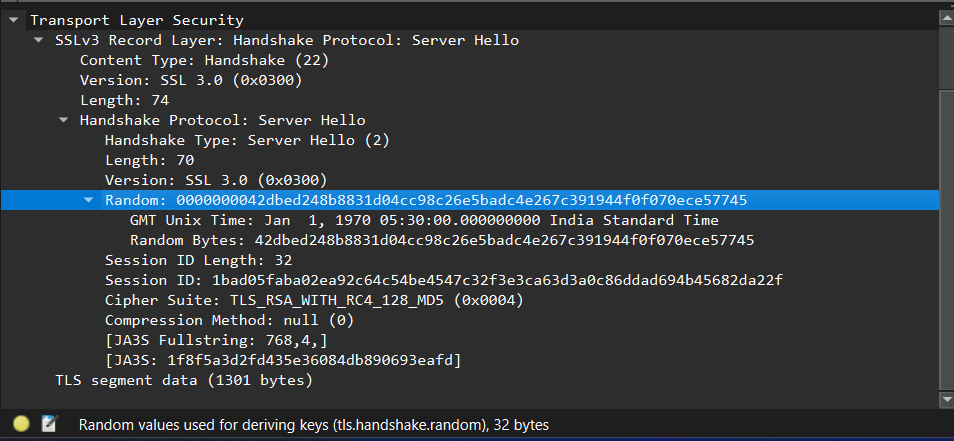
Answer: YES



* 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?

**Locate the Nonce**:

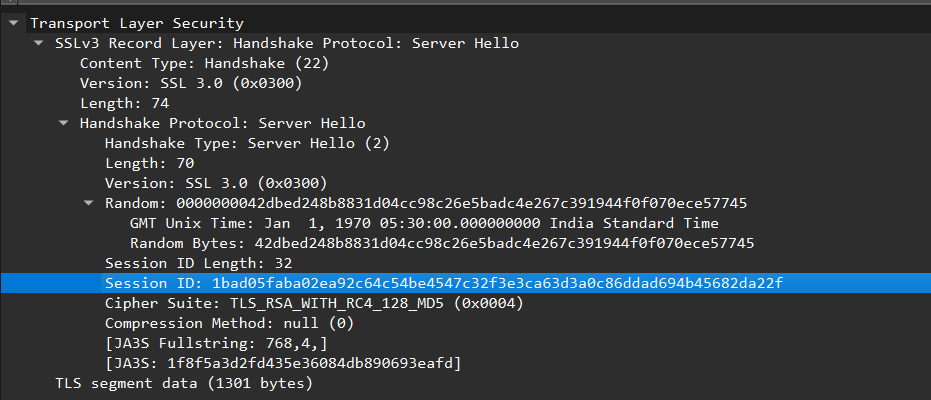
* The **ServerHello** response may not explicitly list a nonce like the **ClientHello** does, but it usually includes a **Session ID** and potentially a **Server Random** value (which acts similarly to a nonce).
* Look for fields labeled **Session ID Length**, **Session ID**, and **Random**.



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

Answer YES

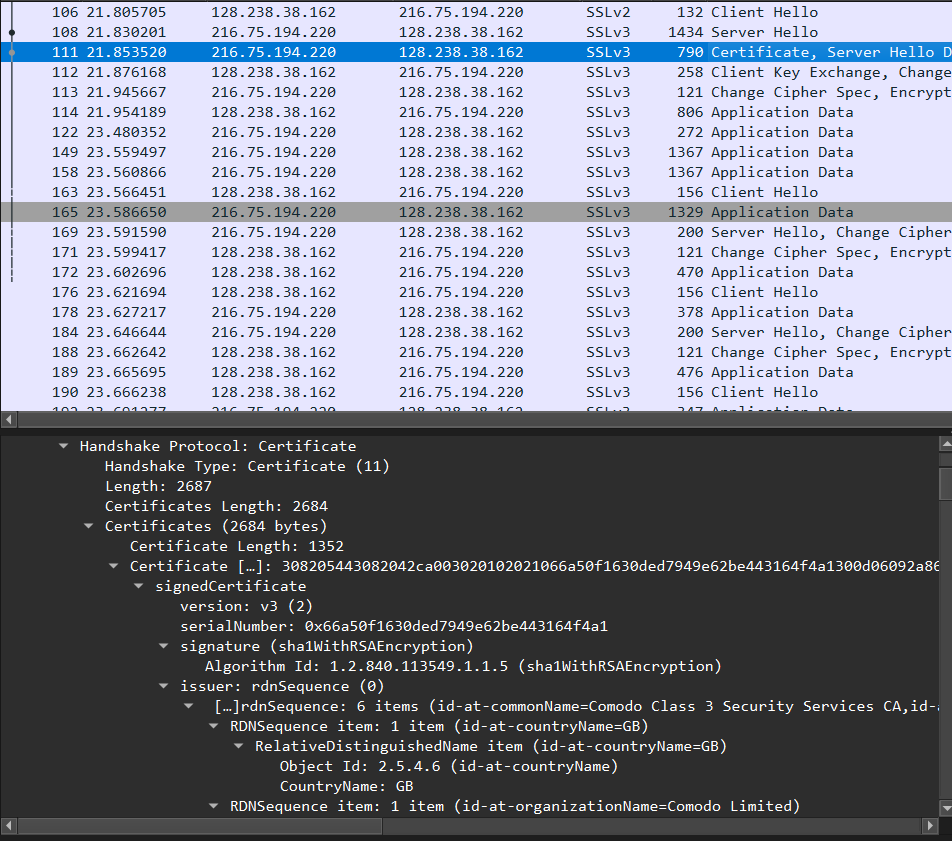


* 1. Does this record contain a certificate, or is the certificate included in a separate record. Does the certificate fit into a single Ethernet frame?

Answer: YES

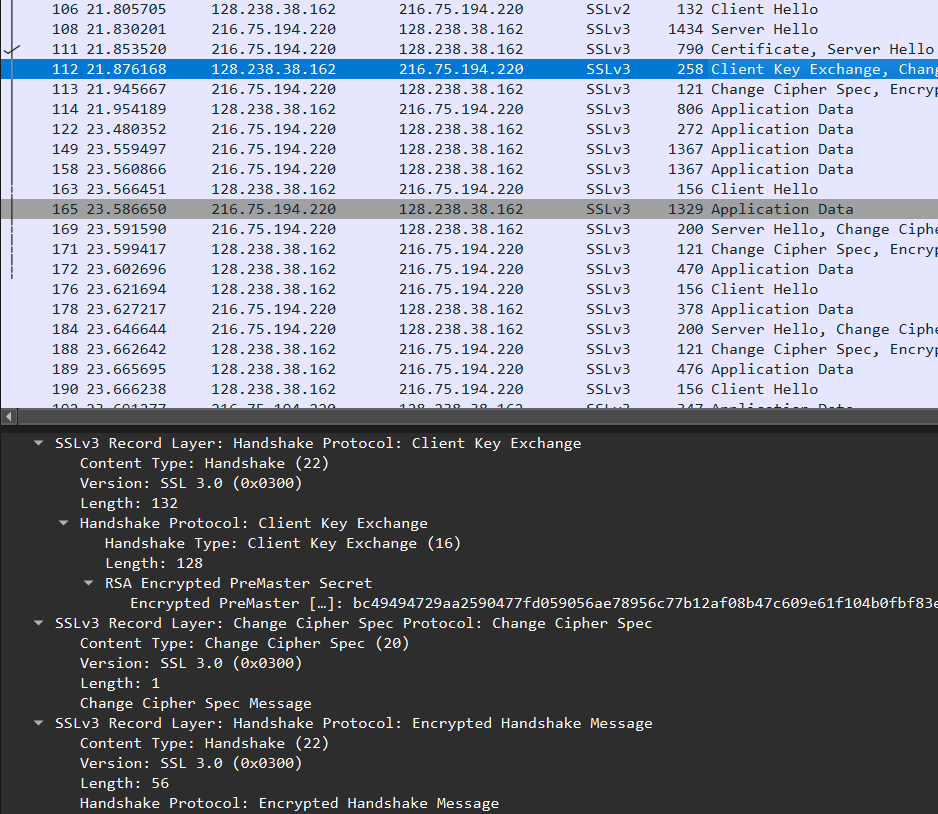
After the **ServerHello** frame, there should be another frame labeled something like **Certificate**. This frame contains the actual server certificate sent by the server.

If the certificate size is less than or equal to 1500 bytes, it will fit into a single Ethernet frame. If it exceeds this size, it will be fragmented across multiple frames.



Client Key Exchange Record:

* 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?



**Presence of Pre-Master Secret**: The **Client Key Exchange** record does contain the pre-master secret, which is crucial for establishing session keys.

**Purpose of the Pre-Master Secret**: The pre-master secret is used to derive symmetric session keys that will encrypt the data exchanged between the client and server after the handshake is complete.

**Encryption**: The pre-master secret is typically encrypted with the server’s public key, ensuring that only the server can decrypt it using its private key.

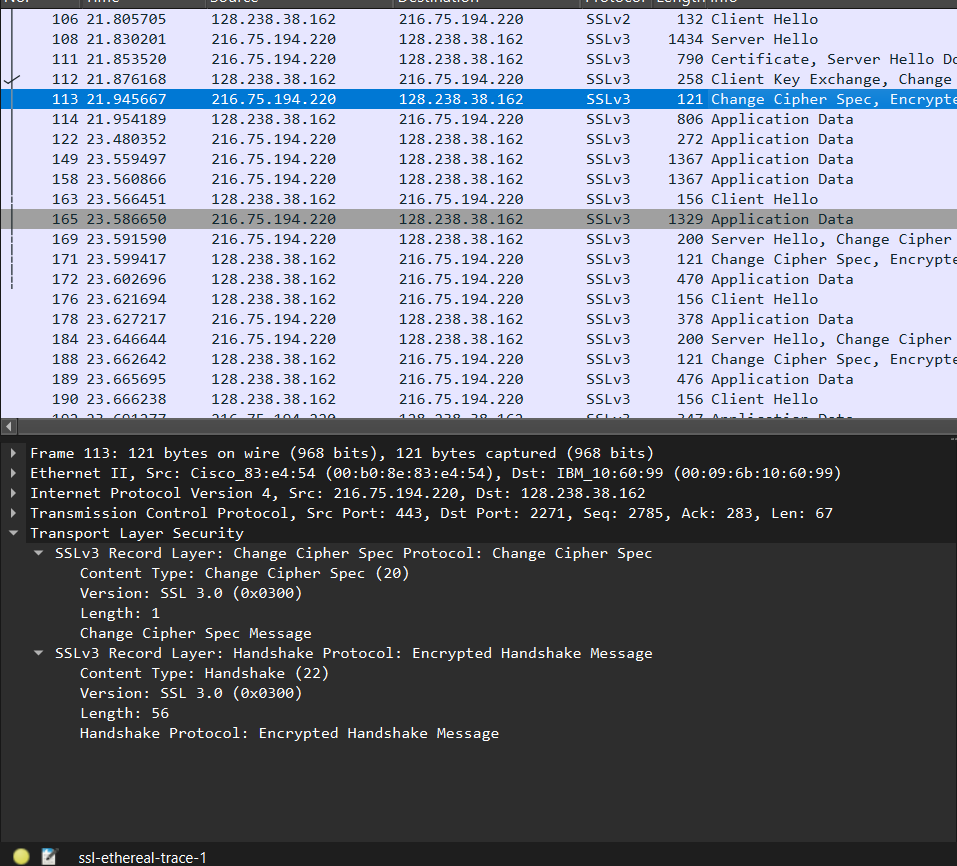
**Length of the Encrypted Secret**: The length of the encrypted pre-master secret is usually around 128 bytes but can vary based on the cipher suite and specific implementation.

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

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

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

**Size**: The record is generally **2 bytes** in total (1 byte for the content type and 1 byte for the Change Cipher Spec message itself).



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

**What is Encrypted**: Handshake messages exchanged during the SSL/TLS handshake process.

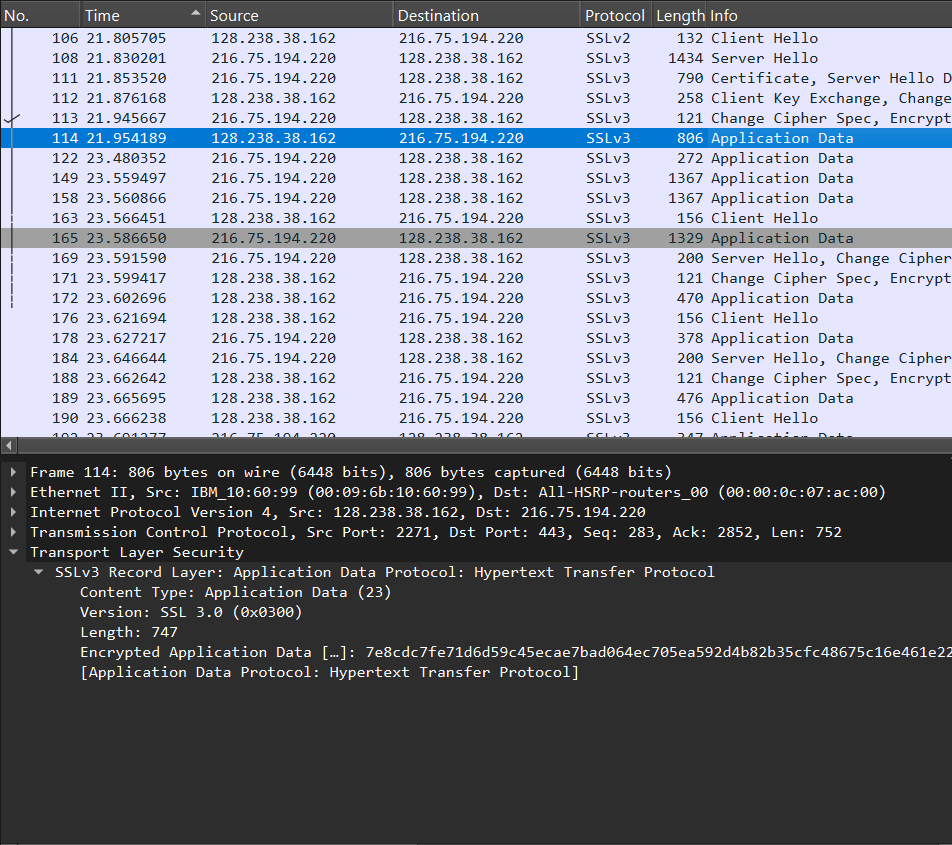
**How it is Encrypted**: Using symmetric-key algorithms determined by the negotiated cipher suite, leveraging session keys derived from the pre-master secret. The messages are encrypted and often accompanied by a MAC for integrity and authenticity.

* 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?

**Both the client and server** send Change Cipher Spec and encrypted handshake records.

The **Change Cipher Spec records** signify readiness for encrypted communication.

The **encrypted handshake records** finalize the handshake and vary in content based on whether they originate from the client or the server, with each party indicating completion of their Application Data



* 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?

**Encryption of Application Data**:

* In SSL/TLS, application data is encrypted using the symmetric encryption algorithms agreed upon during the handshake process. After the Change Cipher Spec record has been exchanged, both the client and server use the session keys derived from the pre-master secret to encrypt and decrypt application data.
* The specific symmetric encryption algorithm (such as AES, DES, etc.) is part of the cipher suite chosen during the handshake.

**Inclusion of MAC (Message Authentication Code)**:

* Yes, the records containing application data include a MAC. The MAC is calculated over the plaintext data (the application data) along with additional information like sequence numbers and the session keys.
* The MAC serves as a form of integrity check, ensuring that the data has not been tampered with during transmission.

**Wireshark Distinction**:

* In Wireshark, encrypted application data and the MAC are typically bundled together in the same record. However, Wireshark does display a breakdown of the decrypted application data, allowing users to view the plaintext contents after decryption.
* If the application data is decrypted (for instance, if the session keys are available to Wireshark), the MAC may not be separately shown in the decrypted data, as it is used internally to verify integrity but does not need to be displayed in the application layer.
  1. Comment on and explain anything else that you found interesting in the trace.

**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.