(1) Analyse TCP packets using the pcap file attached.demo file.pcapng Download demo file.pcapng

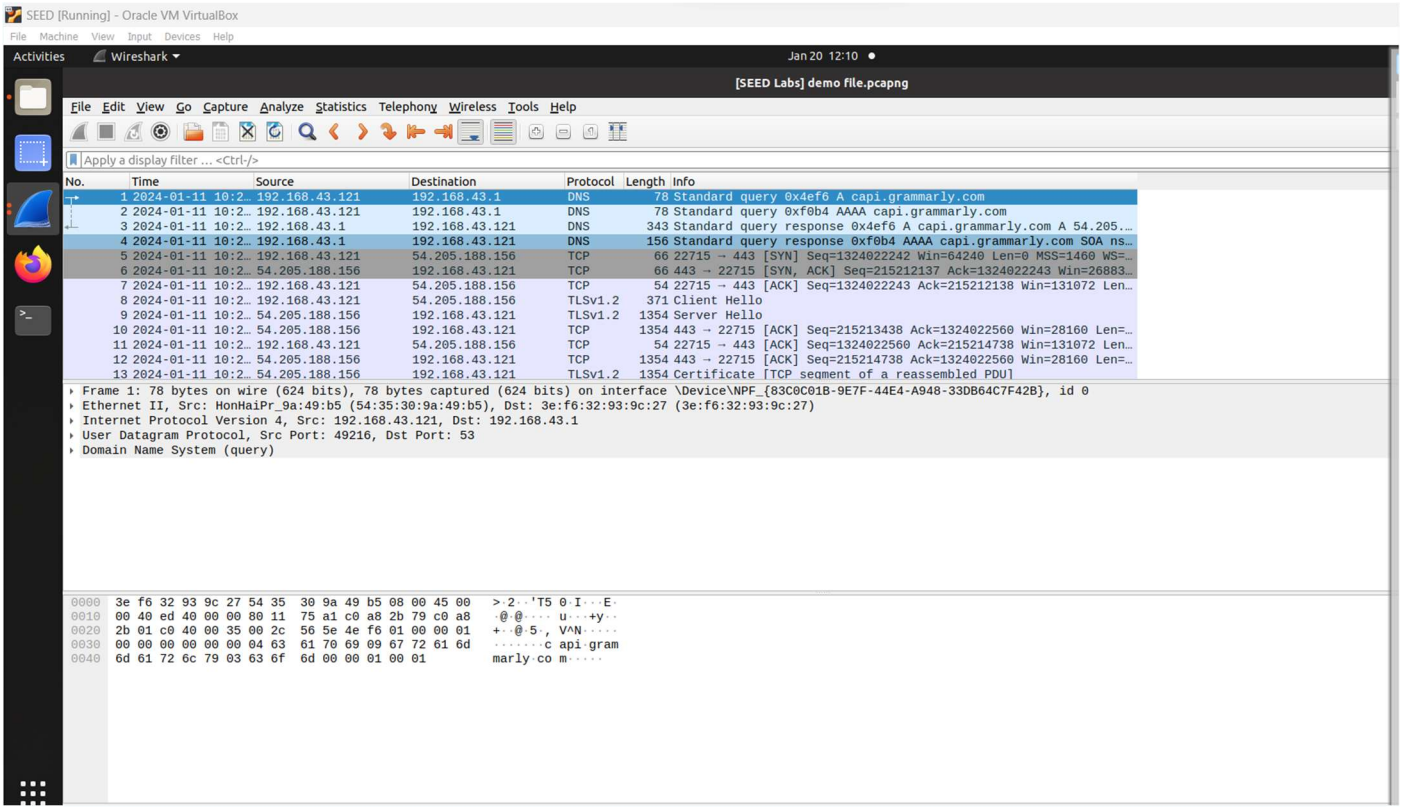
(2) Develop your own sniffer using PCAP API and filter out TCP packets.Sniff -1.c.txt Download Sniff -1.c.txt

(3) Make at report for TCL pcap sniff analysis

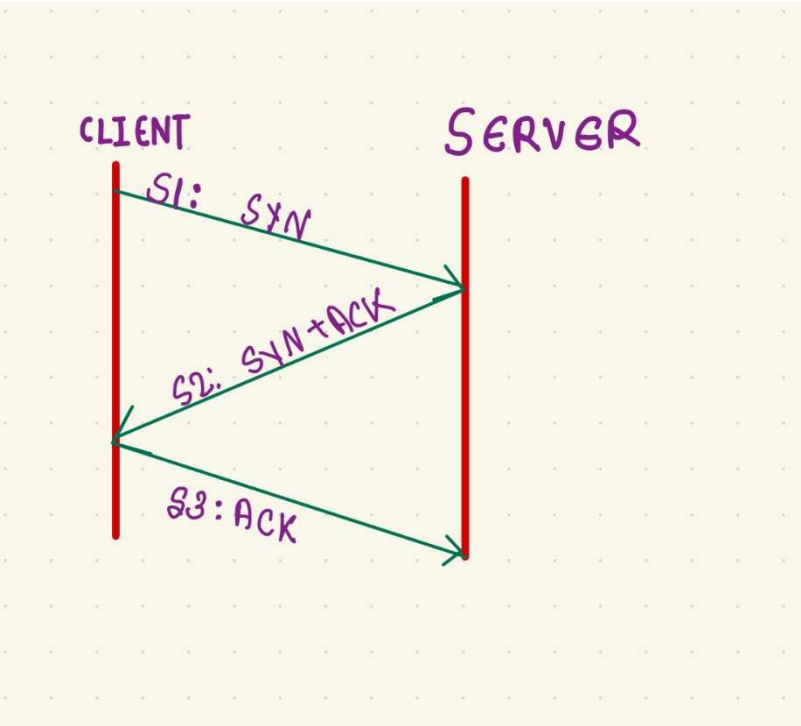
A) To Analyze TCP packets using the pcap file attached.

A TCP protocol works as a transport layer in the OSI model; it works as transport. For example, we have a car that works as data transfer; the data from one place to another, and the TCP works the same. The item of the application layer transfers from destination and source by transport protocol; also, it is a connection protocol, which means before the transfer of data, it will establish some connection between source and destination.

Before understanding the Pcap(Picture Capture) file, lets understand what a wire shark is. A wire shark is a packet sniffing tool that captures and sees the packet in the network between sender and receiver by sniffing, which protocol is used, its length, and what type of message is sent.



In this window we have seen at top we will able to seen the type , source , destination , length and information of packets that are being transferred. after going to first tcp packet we have seen multiple things such as internet protocol, ethernet , frame number and details of its TCP protocol.

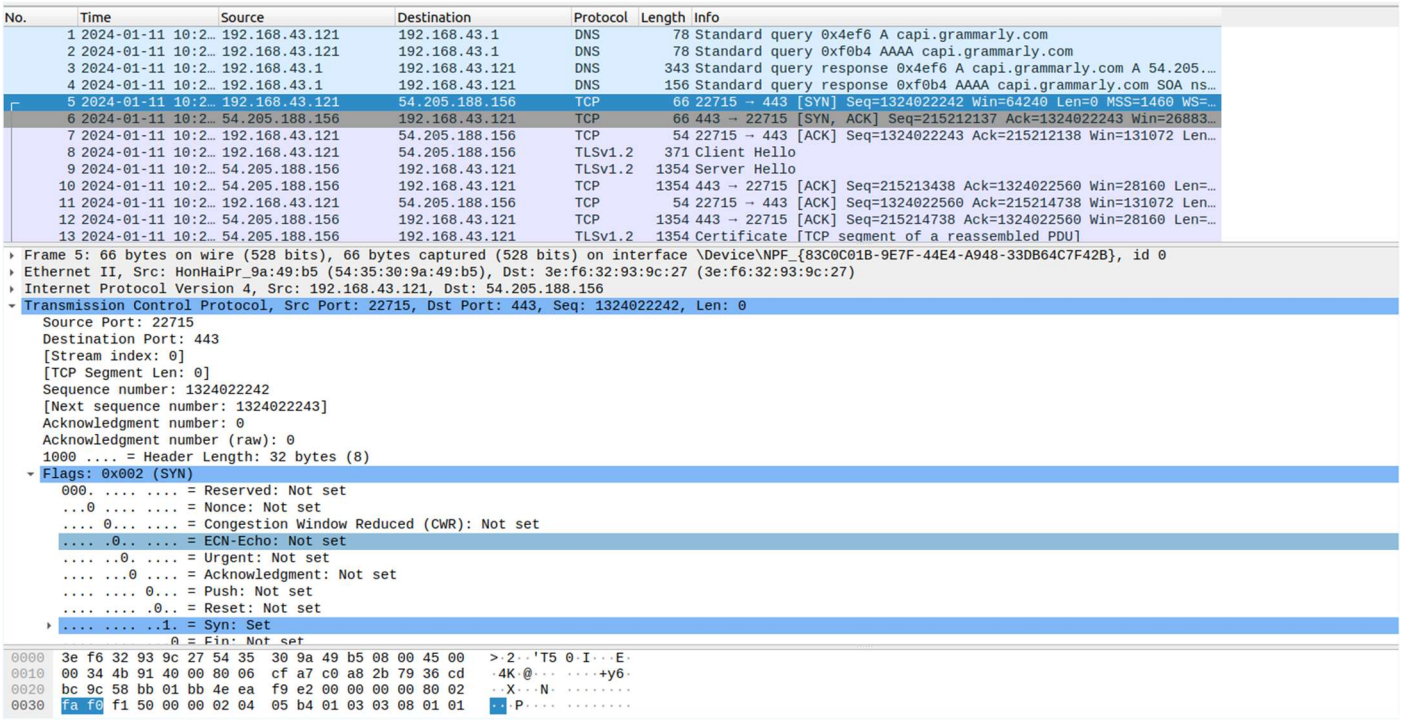


In the TCP protocol, the initial connection establishment involves three steps, as illustrated in the diagram above. Additionally, we are examining this process in Wireshark, with the corresponding image provided below.

S1: SYN: Initiating Connection

The client triggers the connection initiation process by dispatching a SYN packet, signifying its desire to establish a connection. The TCP segment length of 0 suggests that this pertains to the preliminary handshake stage, with no data exchange occurring at this point.

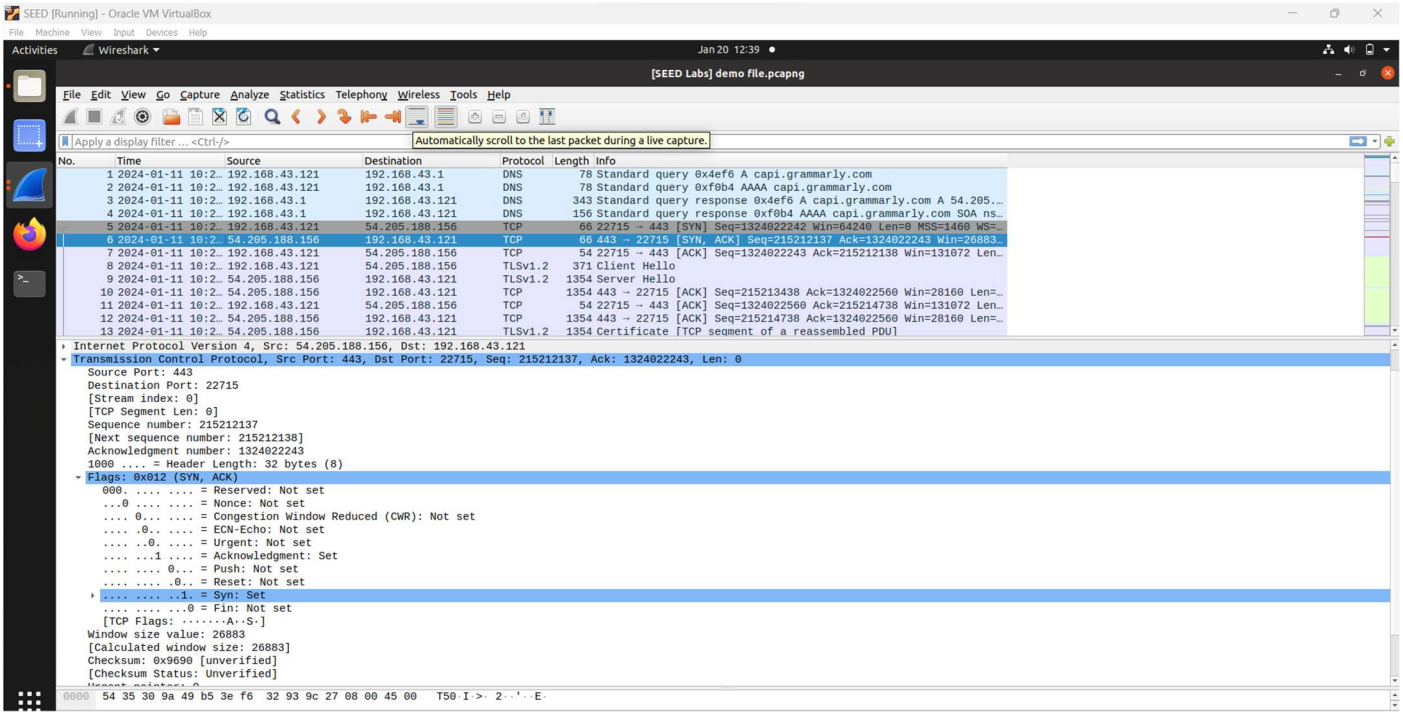
In attached we can see that **the SYN is set but ACK is not set** that means first step for connection establishment is done.



S2: SYN-ACK: Acknowledging Connection Request

The server acknowledges the client's SYN packet by responding with a SYN-ACK packet, signifying the acknowledgment of synchronization. The flags in the packet indicate that both SYN and ACK are set, confirming the successful establishment of the connection.

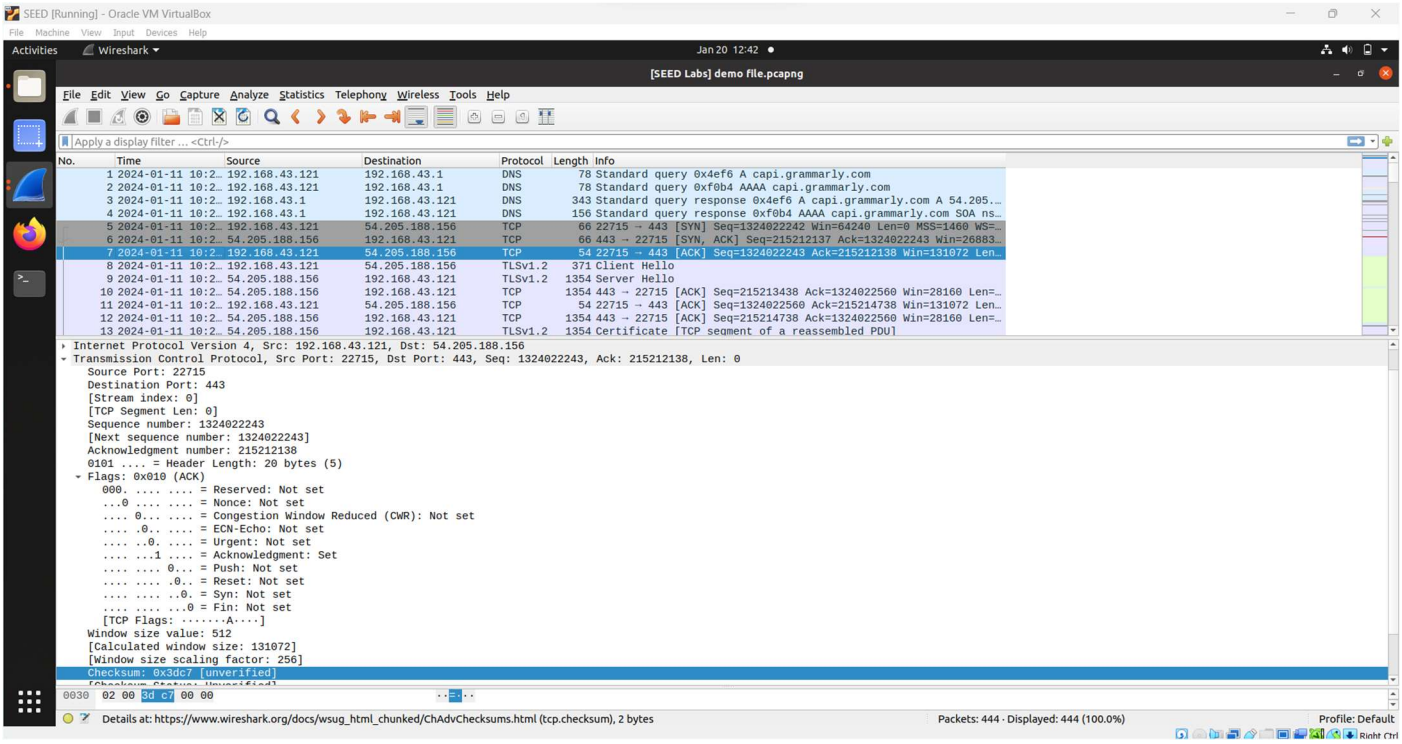
Also, in below snap shot we can see that **SYN and ACK both flags are set** that means step 2 of connection establishment.



S3: ACK: Completing Connection Establishment

The client, upon receiving the SYN-ACK packet from the server, responds with a TCP packet containing the ACK flag set, This ACK packet serves as confirmation to the server that the client has received the acknowledgment, and both ends are now synchronized.The TCP segment length remains at 0, indicating that this is an acknowledgment without any data exchange at this stage.

Also in snapshot attached below **SYN flag is not set while ACK flag is set**

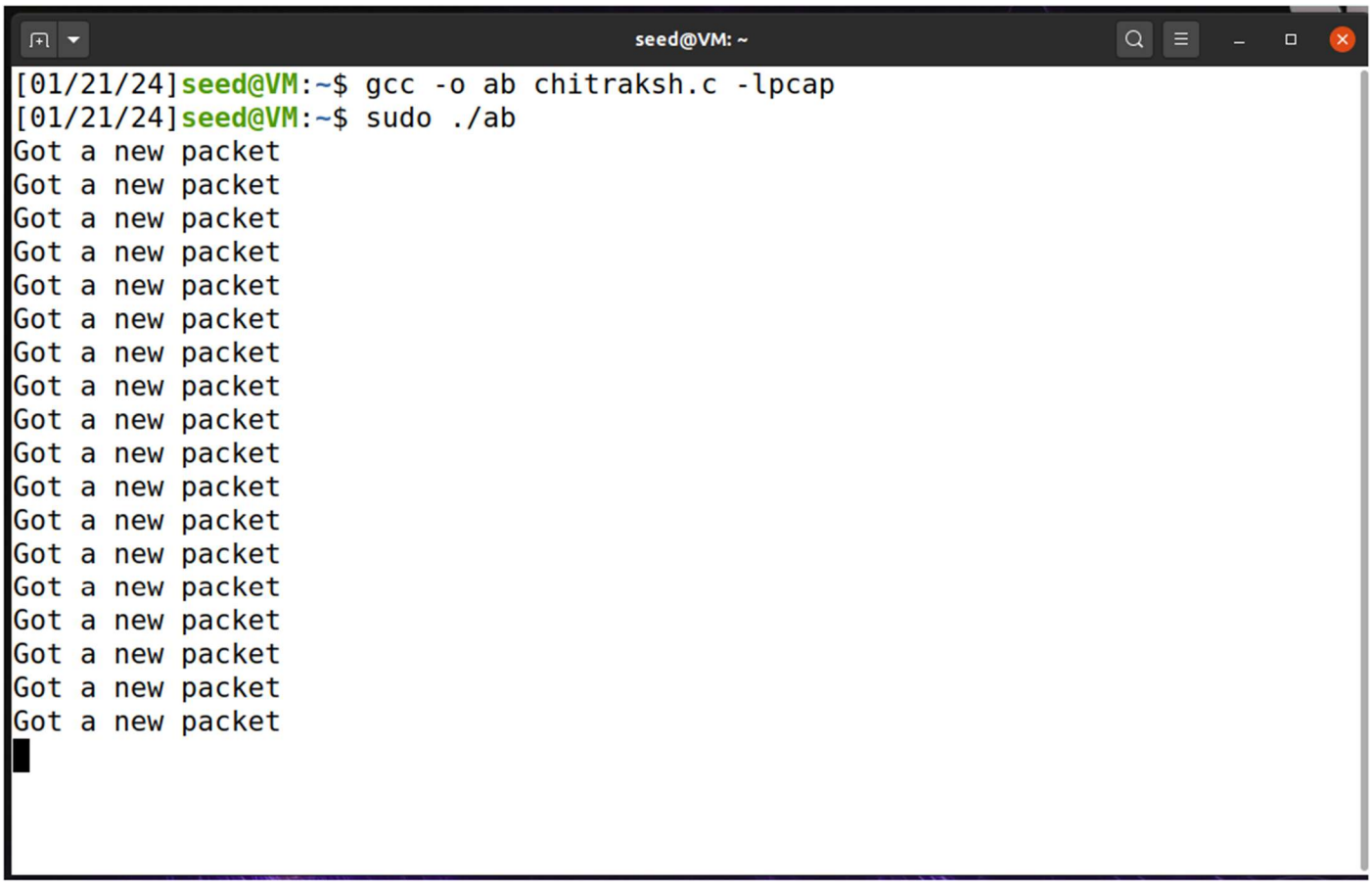


The three-way handshake is like a conversation between a client and a server when they want to connect. First, the client starts the conversation by asking to connect. Then, the server responds, saying it's ready to connect. Finally, the client confirms that it got the message from the server. This whole process ensures that the client and server are on the same page and ready to share information in a reliable and coordinated way before any actual data is sent between them.

B) Develop your own sniffer using PCAP API and filter out TCP packets.

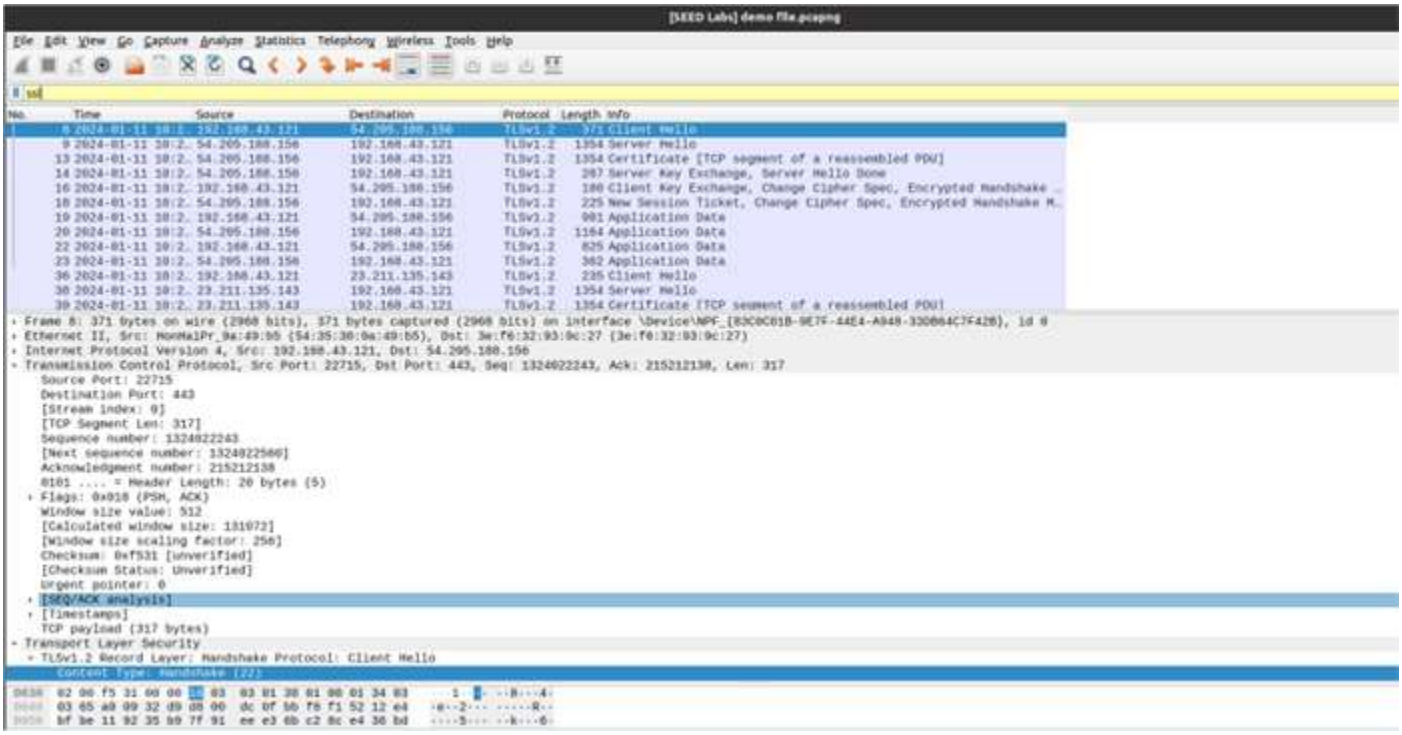
```
#include <stdlib.h>
#include <stdio.h>
#include <pcap.h>
void got_packet(u_char *args, const struct pcap_pkthdr *header, const u_char *packet)
{
   printf("Got a new packet\n");
}
int main()
{
   pcap_t *handle;
   char errbuf[PCAP_ERRBUF_SIZE];
   struct bpf_program fp;
   char filter_exp[] = "tcp";
   bpf_u_int32 net;
```

```
    handle = pcap_open_live("any", BUFSIZ, 1, 1000, errbuf);

    pcap_compile(handle, &fp, filter_exp, 0, net);
    if (pcap_setfilter(handle, &fp) != 0)
    {
        pcap_perror(handle, "Error:");
        exit(EXIT_FAILURE);
    }

    pcap_loop(handle, -1, got_packet, NULL);
    pcap_close(handle);
    return 0;
}
```



C) Make at report for TCL pcap sniff analysis using the questionnaire.

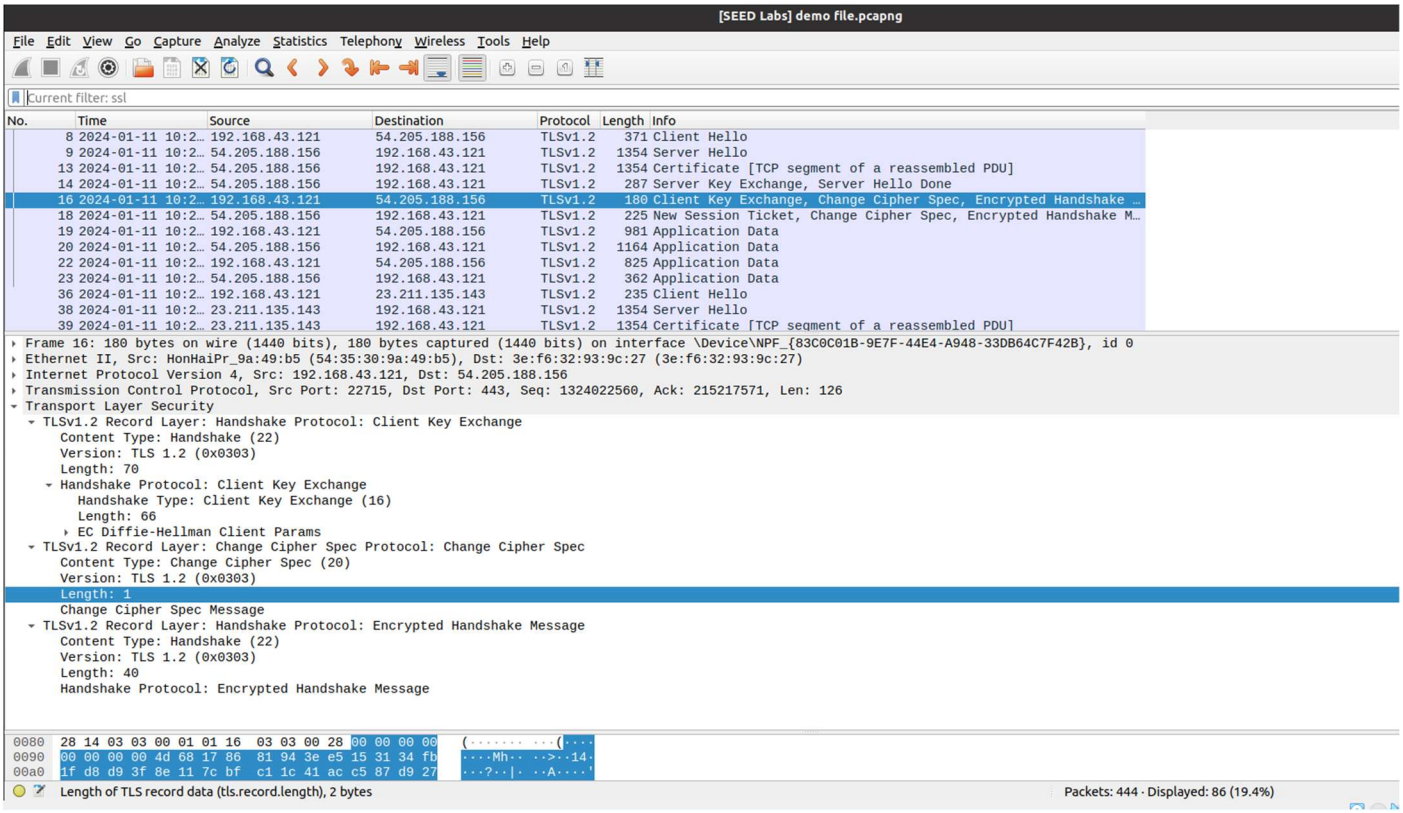
1. **For each of the first 8 Ethernet frames, specify the source of the frame (client or server), determine the number of SSL records that are included in the frame, and list the SSL record types that are included in the frame. Draw a timing diagram between client and server, with one arrow for each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



Frame	Source	Destination	Info/Type
8	192.168.43.121	54.205.188.156	Client Hello
9	54.205.188.156	192.168.43.121	Server Hello
13	54.205.188.156	192.168.43.121	Certificte
14	54.205.188.156	192.168.43.121	Server Hello Done
16	192.168.43.121	54.205.188.156	Client Key Exchange
18	54.205.188.156	192.168.43.121	Change Cipher Spec
19	192.168.43.121	54.205.188.156	Application Data
20	54.205.188.156	192.168.43.121	Application Data

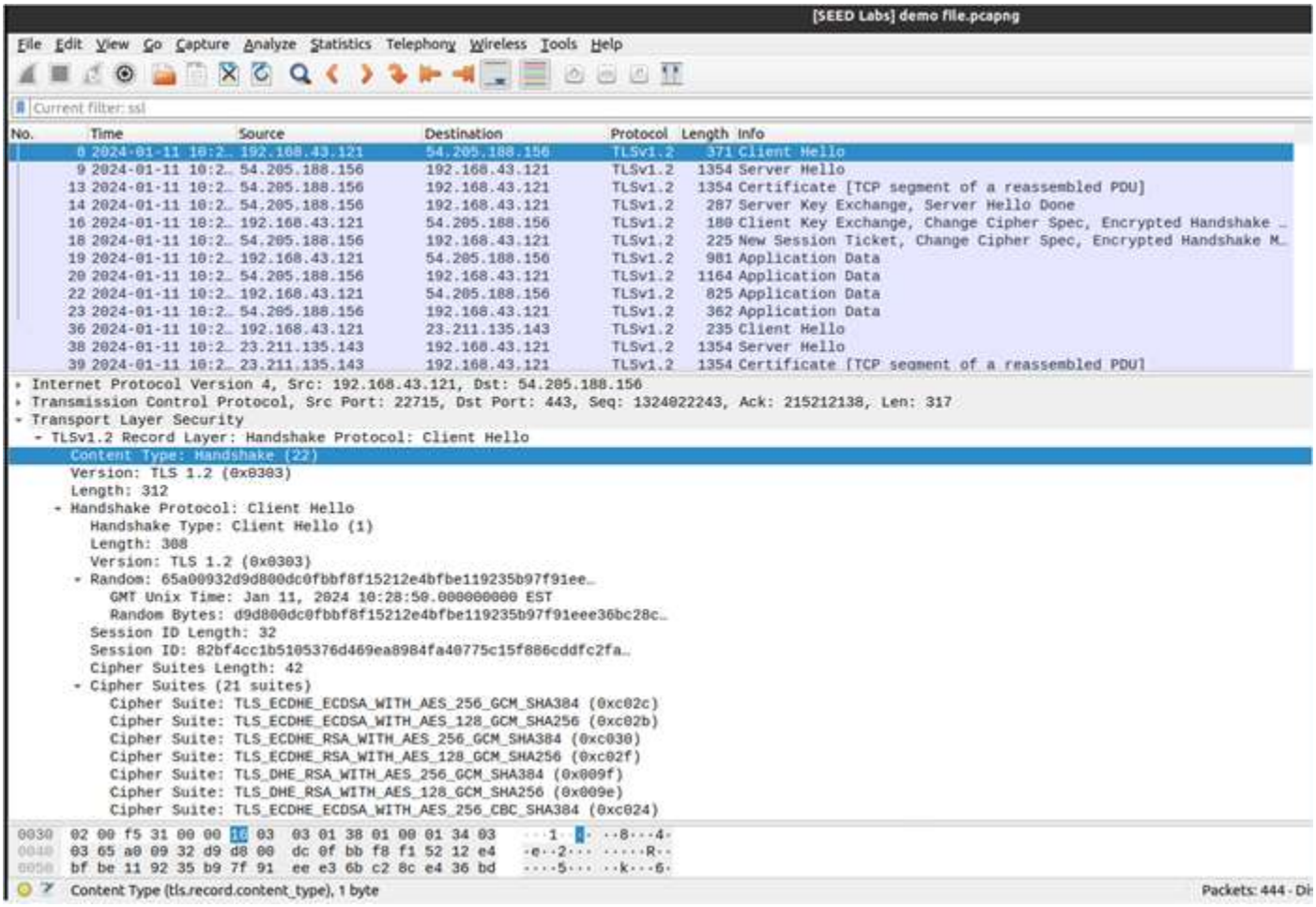
2. Each of the SSL records begins with the same three fields (with possibly different values). One of these fields is "content type" and has length of one byte. List all three fields and their lengths.

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



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?



The content type is 22, for Handshake Message, with a handshake type: Client Hello

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?

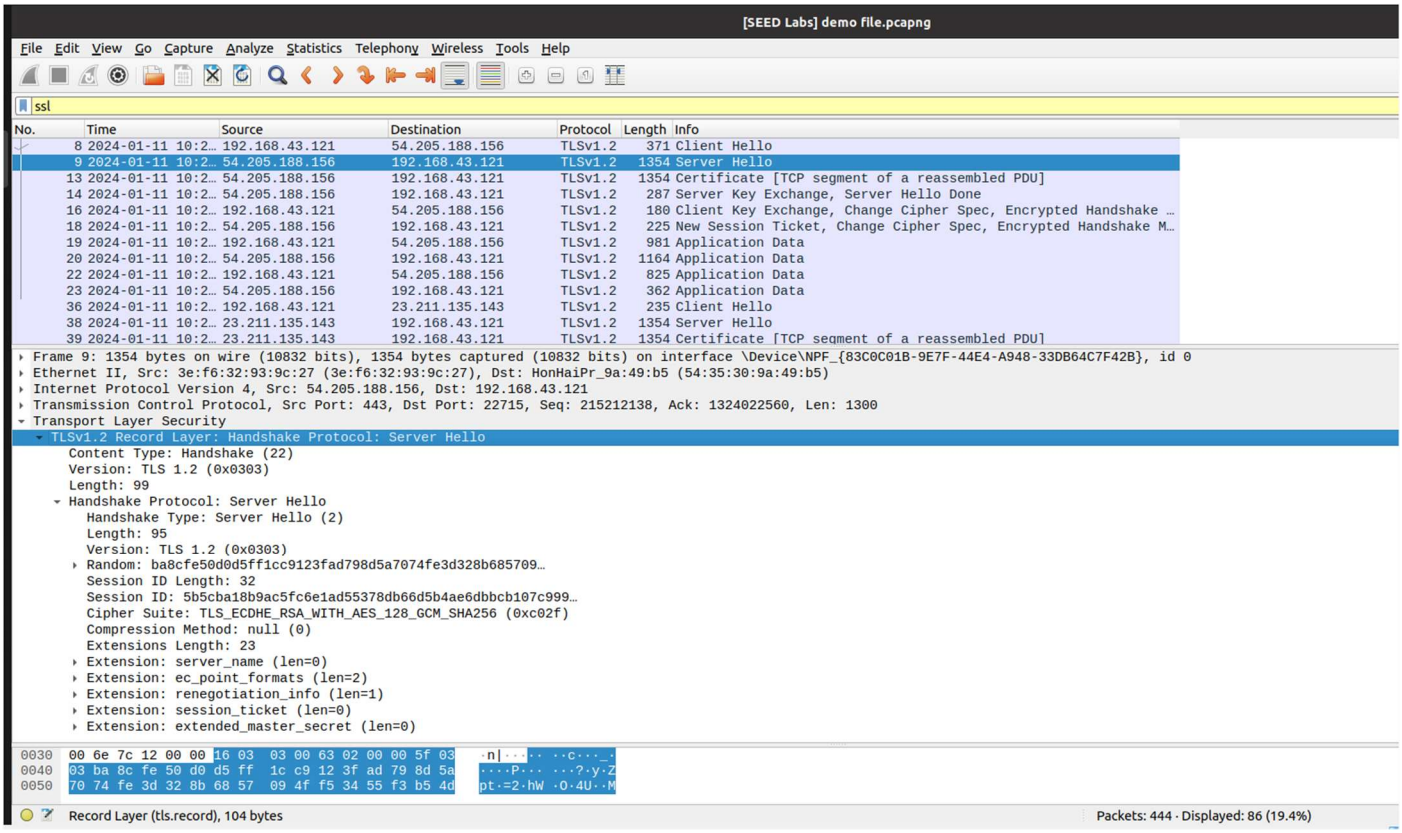
5. Does the ClientHello record advertise the cyber suites it supports? If so, in the first listed suite, what are the public-key algorithm, the symmetric-key algorithm, and the hash algorithm?

- Public key algorithm: RSA
- Symmetric-key algorithm: RC4
- Hash algorithm: MD5

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?

- The cipher suite utilizes RSA for public key cryptography.
- RC4 is employed as the symmetric-key cipher in the suite.
- The MD5 hash algorithm is used within the cipher suite.



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?

- The record includes a nonce listed under "Random."
- The nonce is 32 bits long, with 28 bits for data and 4 bits for the time.
- The primary purpose of the nonce is to prevent a replay attack.

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

It does offer a distinct and enduring identifier for the SSL session, which is transmitted without encryption**.**

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?

There isn't a certificate; it's stored in a different record. It does fit within a sole Ethernet frame.