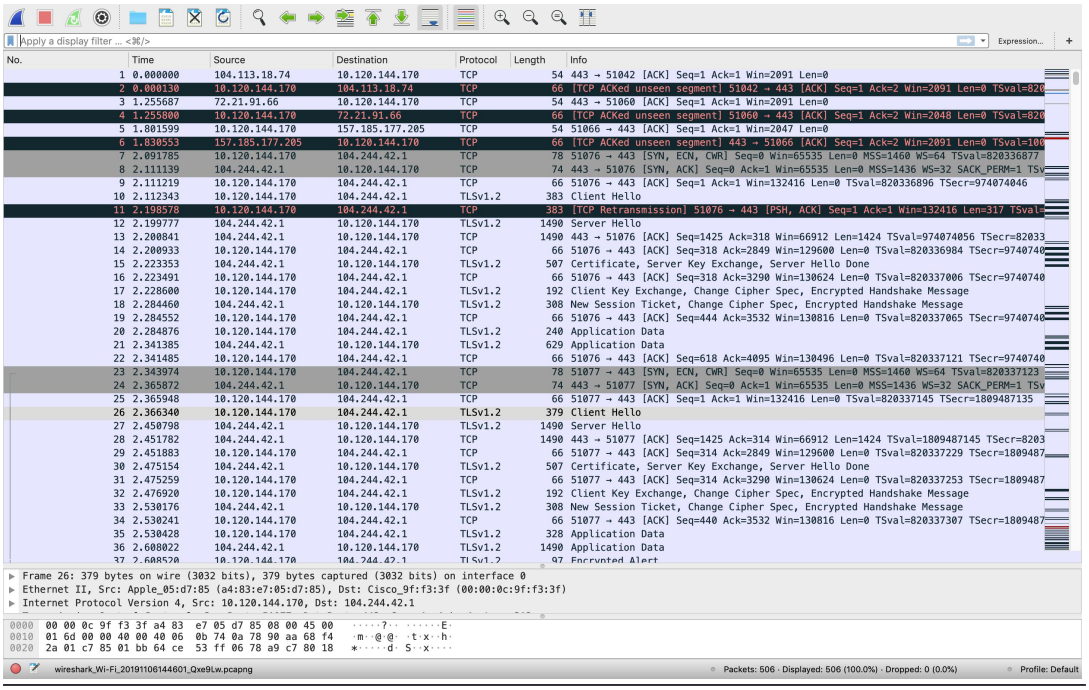
Step 1. Capture a trace

1) Find a URL, I am using this one, http://www.twitter.com.
2) Fetch the URL with wget command.

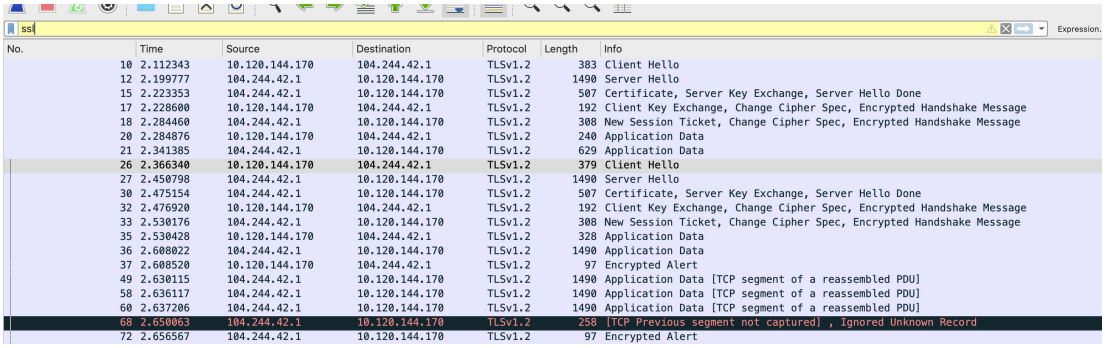
```
➜  ~ wget --secure-protocol=TLSv1 --no-check-certificate https://www.twitter.com

--2019-11-06 14:46:17--  https://www.twitter.com/
Resolving www.twitter.com (www.twitter.com)... 104.244.42.1, 104.244.42.65
Connecting to www.twitter.com (www.twitter.com)|104.244.42.1|:443... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://twitter.com/ [following]
--2019-11-06 14:46:17--  https://twitter.com/
Resolving twitter.com (twitter.com)... 104.244.42.1, 104.244.42.65
Connecting to twitter.com (twitter.com)|104.244.42.1|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 311425 (304K) [text/html]
Saving to: ‘index.html.4’

index.html.4        100%[===================>] 304.13K   685KB/s    in 0.4s

2019-11-06 14:46:18 (685 KB/s) - ‘index.html.4’ saved [311425/311425]
```

3) Set up the filter to be ‘tcp port 443’, then repeat the wget command in 2) step.



Step 2. Inspect the Trace and TCP Segment Structure

To begin with, type in ‘ssl’ in the filter.



Answer the following questions to show your understanding of SSL records:

1. What is the Content-Type for a record containing “Application Data”?

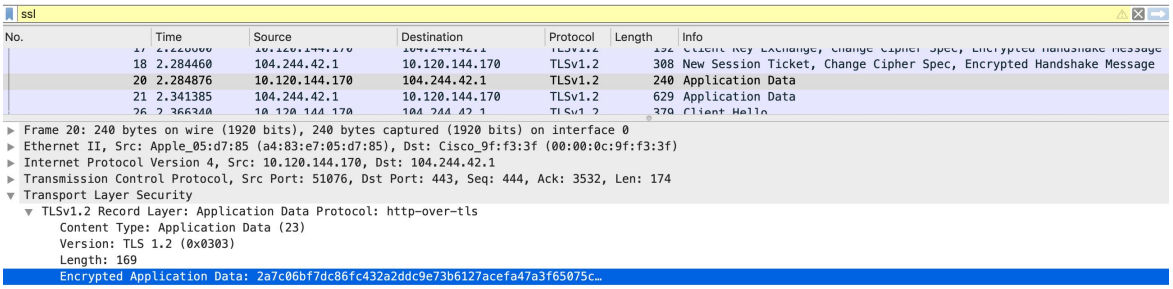
Application Data (23)

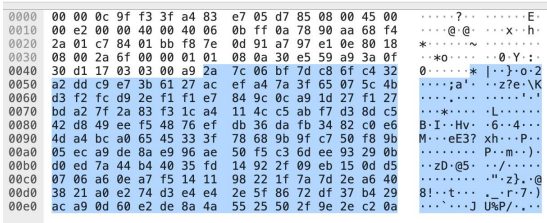
2. What version constant is used in your trace, and which version of TLS does it represent?

TLS 1.2 (0x0303)

3. Does the Length cover the Record Layer header as well as payload, or only the payload?

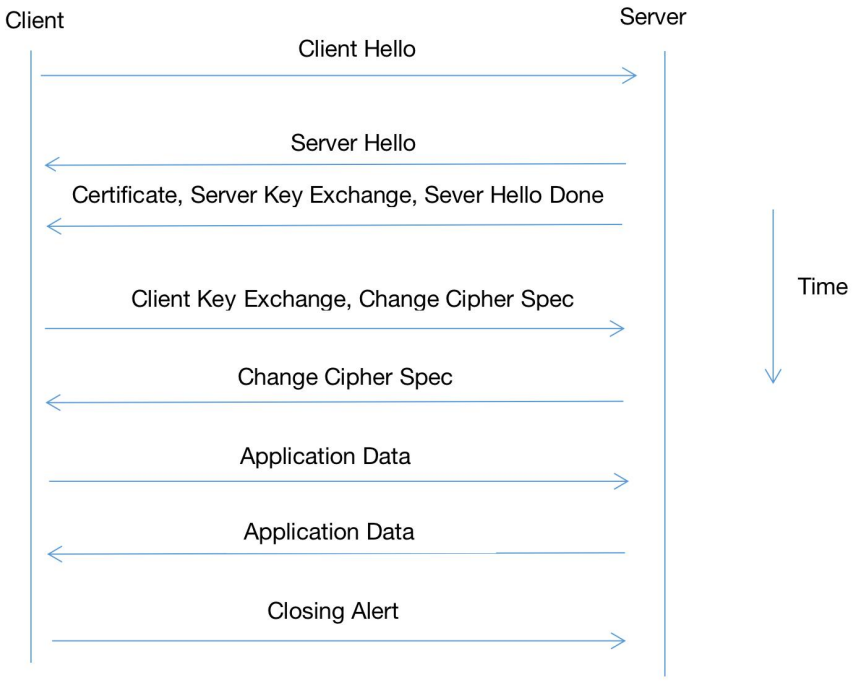
Only the payload.





Step 3. The SSL Handshake

Overall Handshake

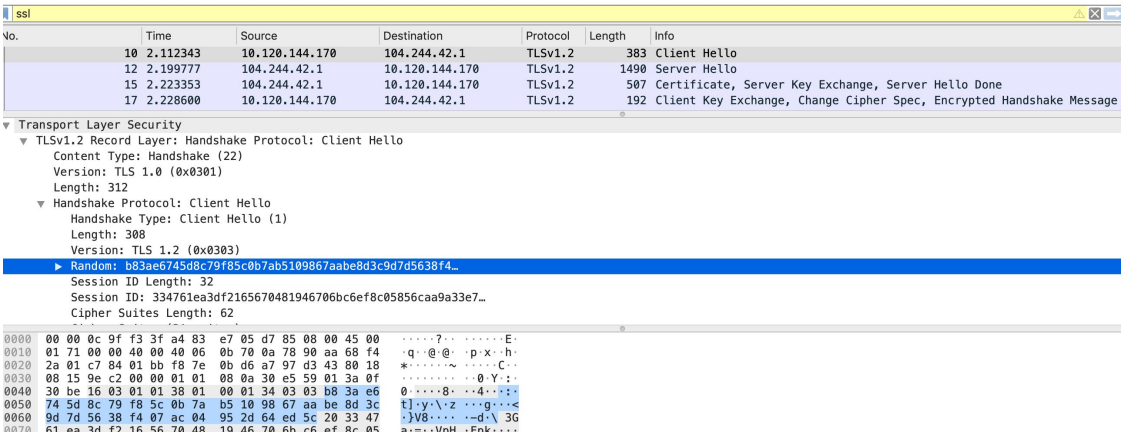


Hello Messages

Answer the following questions:

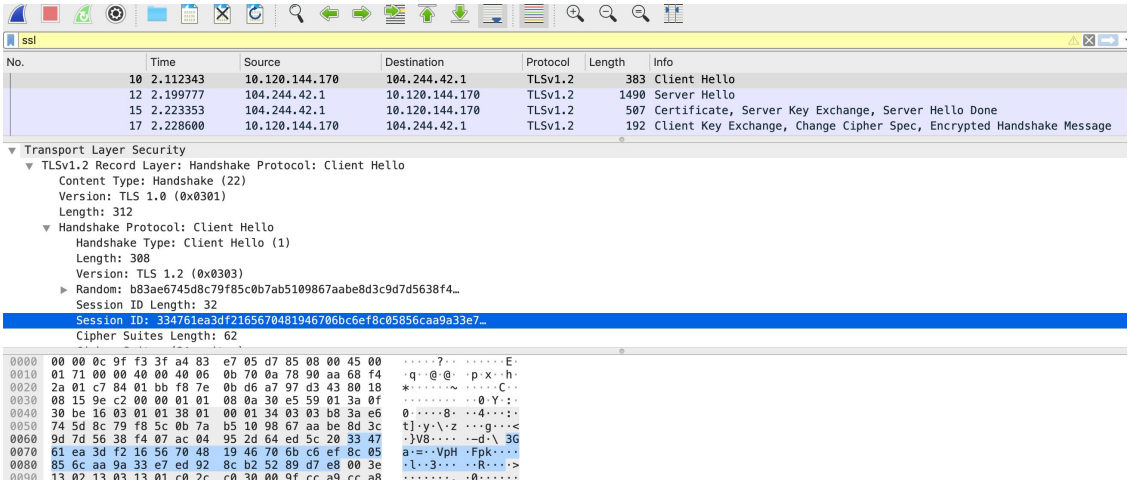
1. How long in bytes is the random data in the Hellos? Both the Client and Server include this random data (a nonce) to allow the establishment of session keys.

32 bytes.



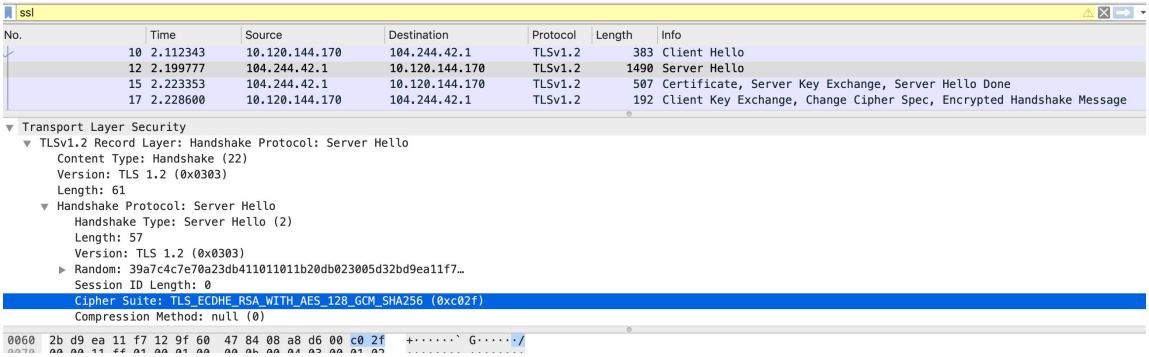
2. How long in bytes is the session identifier sent by the server? This identifier allows later resumption of the session with an abbreviated handshake when both the client and server indicate the same value. In our case, the client likely sent no session ID as there was nothing to resume.

32 bytes.



3. What Cipher method is chosen by the Server? Give its name and value. The Client will list the different cipher methods it supports, and the Server will pick one of these methods to use.

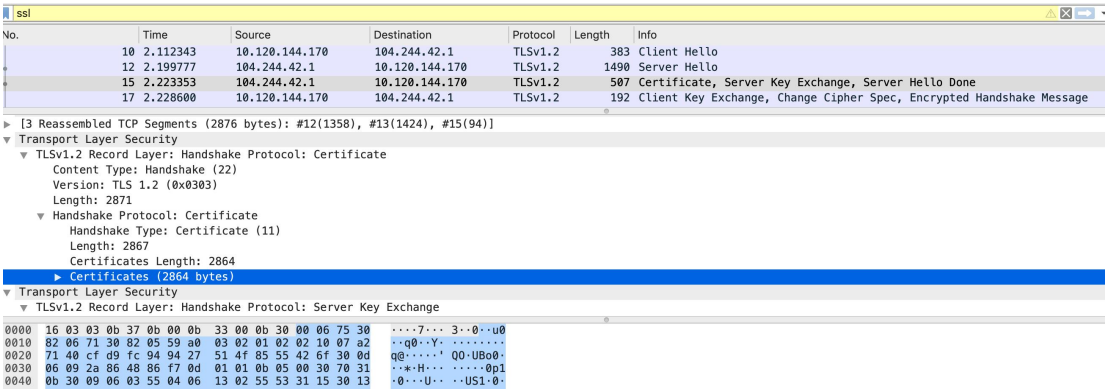
Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f).



Certificate Messages

4. Who sends the Certificate, the client, the server, or both? A certificate is sent by one party to let the other party authenticate that it is who it claims to be. Based on this usage, you should be able to guess who sends the certificate and check the messages in your trace.

The server.

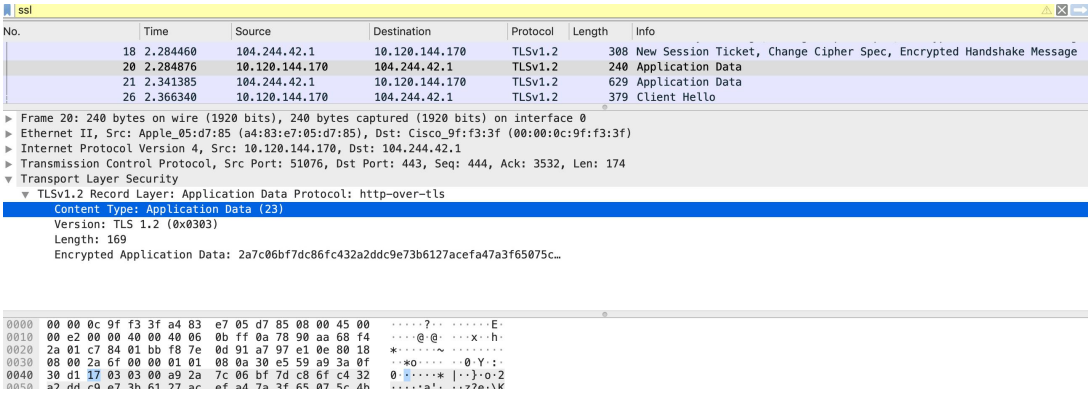


Client Key Exchange and Change Cipher Messages

5. At the Record Layer, what Content-Type values are used to indicate each of these messages? Say whether the values are the same or different than that used for the Hello and Certificate messages. Note that this question is asking you to look at the Record Layer and not an inner Handshake Protocol.

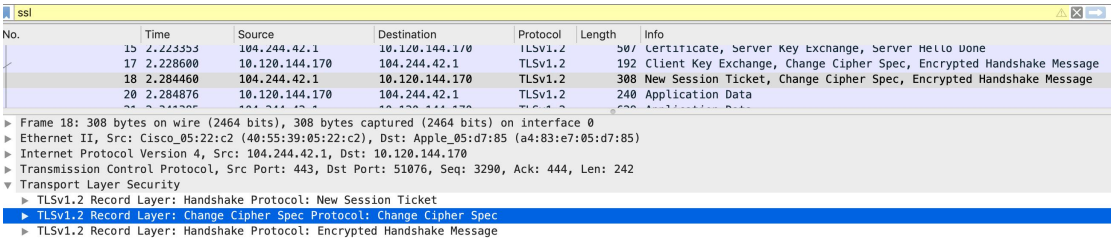
The values are represented as integers, and they are different.

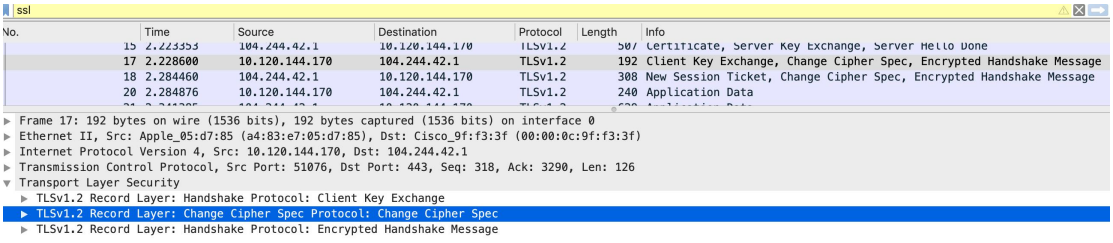
Application data type has a value of 23, while hello message is of Handshake data type, its value is 22.



6. Who sends the Change Cipher Spec message, the client, the server, or both?

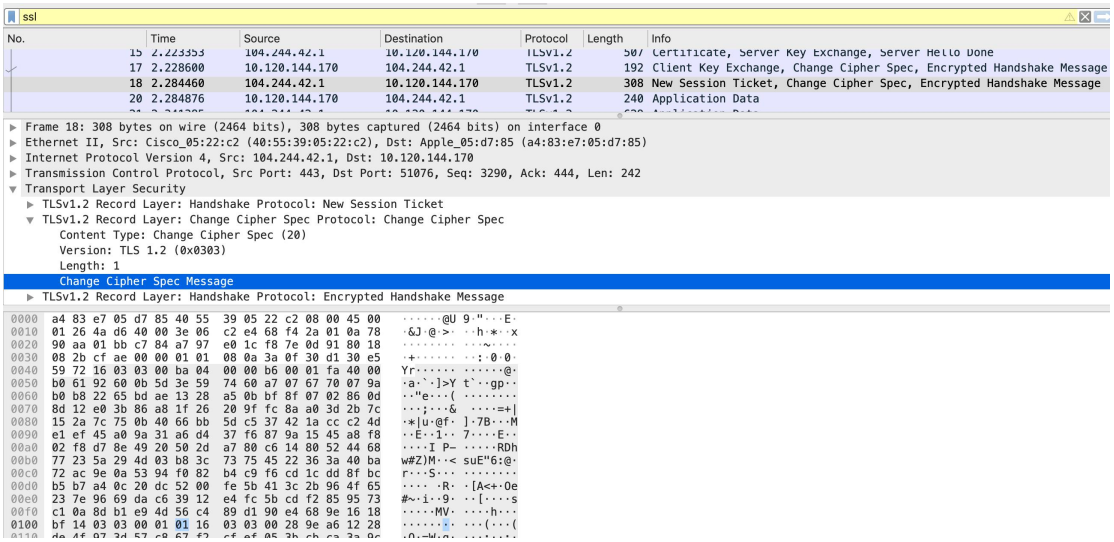
Both the server and the client.





7. What are the contents carried inside the Change Cipher Spec message? Look past the ContentType and other headers to see the message itself.

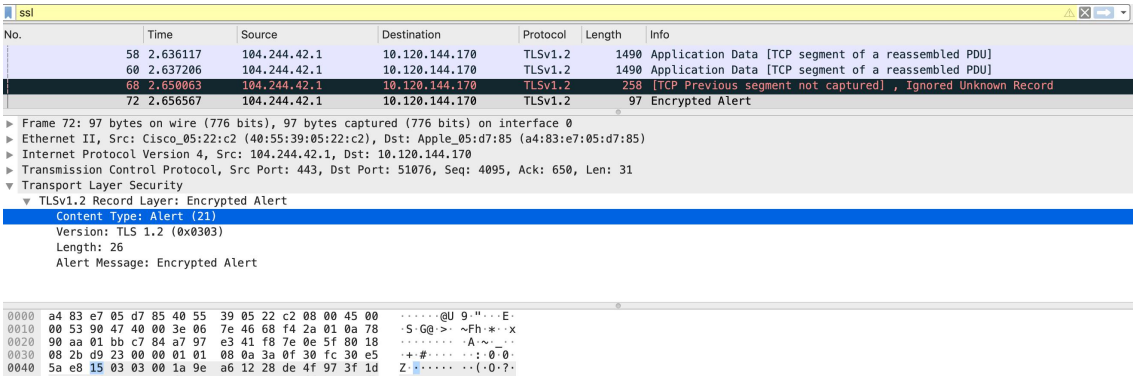
Change Cipher Spec Message: 01



Alert Messages

8. At the Record Layer, what Content-Type value is used to signal an alert?

Alert (21)



9. Tell us whether the contents of the alert are encrypted or sent in the clear? To check this, see whether you can read the contents of the alert to see what kind of alert has been sent.

Encrypted.