TCP
Explain referring to the monitoring results:

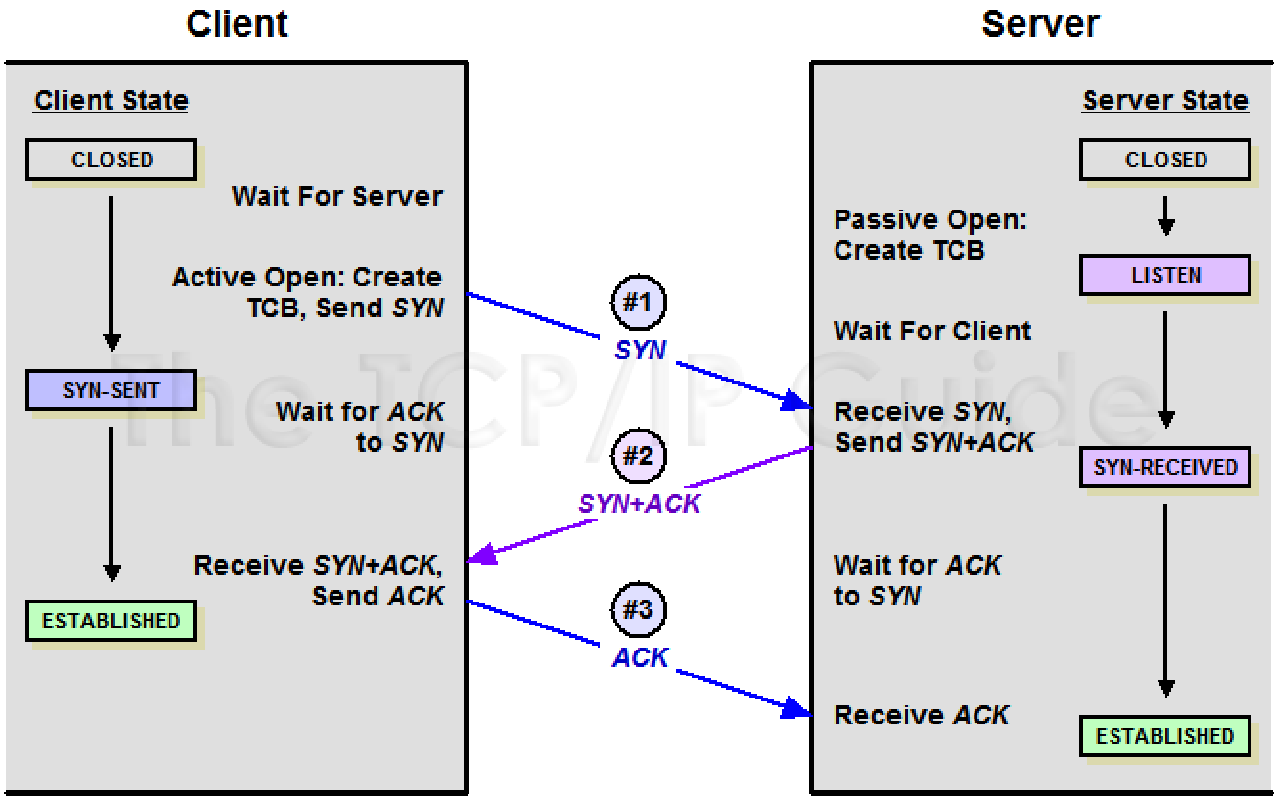
1. **How TCP connection is established?**

Ans: A three-way handshake is a method used in a TCP/IP network to create a connection between a local host/client and server. It is a three-step method that requires both the client and server to exchange SYN (synchronous) and ACK (acknowledgment) packets before actual data communication begins. A three-way handshake is also known as a TCP handshake.

The TCP level of the TCP/IP transport protocol is connection-oriented which means that, before any data can be transmitted, a reliable connection must be obtained and acknowledged. So, TCP must set up virtual connection between two hosts before any data are sent. This means the two hosts must agree on certain parameters, data flow, windowing, error detection, and options.

The host that initiates communication sends a synchronous (SYN) packet to the receiver. The receiver acknowledges this request by sending a SYN/ACK packet. This packet translates into, “I received your request and am ready to communicate with you.” The sending host acknowledges this with an acknowledgment (ACK) packet, which translates into, “I received your acknowledgement. Let’s start transmitting our data.” This completes the handshaking phase, after which a virtual connection is set up, and actual data can now be passed. The connection that has been set up at this point is considered full duplex, which means transmission in both directions is possible using the same transmission line.

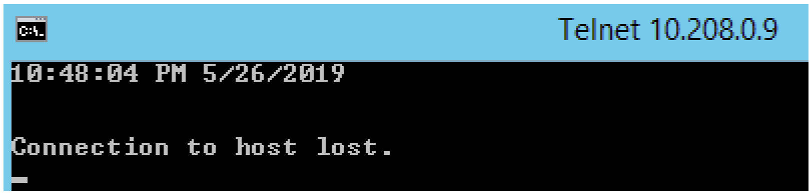
So, by three handshake connection TCP connection is established.
Although the three-way handshake only requires three packets to be transmitted over our networked media, the termination of this reliable connection will necessitate the transmission of four packets. Because a TCP connection is full duplex (that is, data can be flowing in each direction independent of the other), each direction must be terminated independently.
Here, this diagram demonstrate how TCP connection is established using 3-handshake connection.



2. Who opens the connection?

Ans: Windows Server 2 opens the connection whose IP is 10.208.0.20 as client in this case by sending synchronous (SYN) packet to the receiver.

In order to open the connection, a client send SYN (synchronous) data packet over an IP network to a server on the same or an external network in this case Windows Server 2 which is on same network. The objective of this packet is to ask/infer if the server is open for new connections.



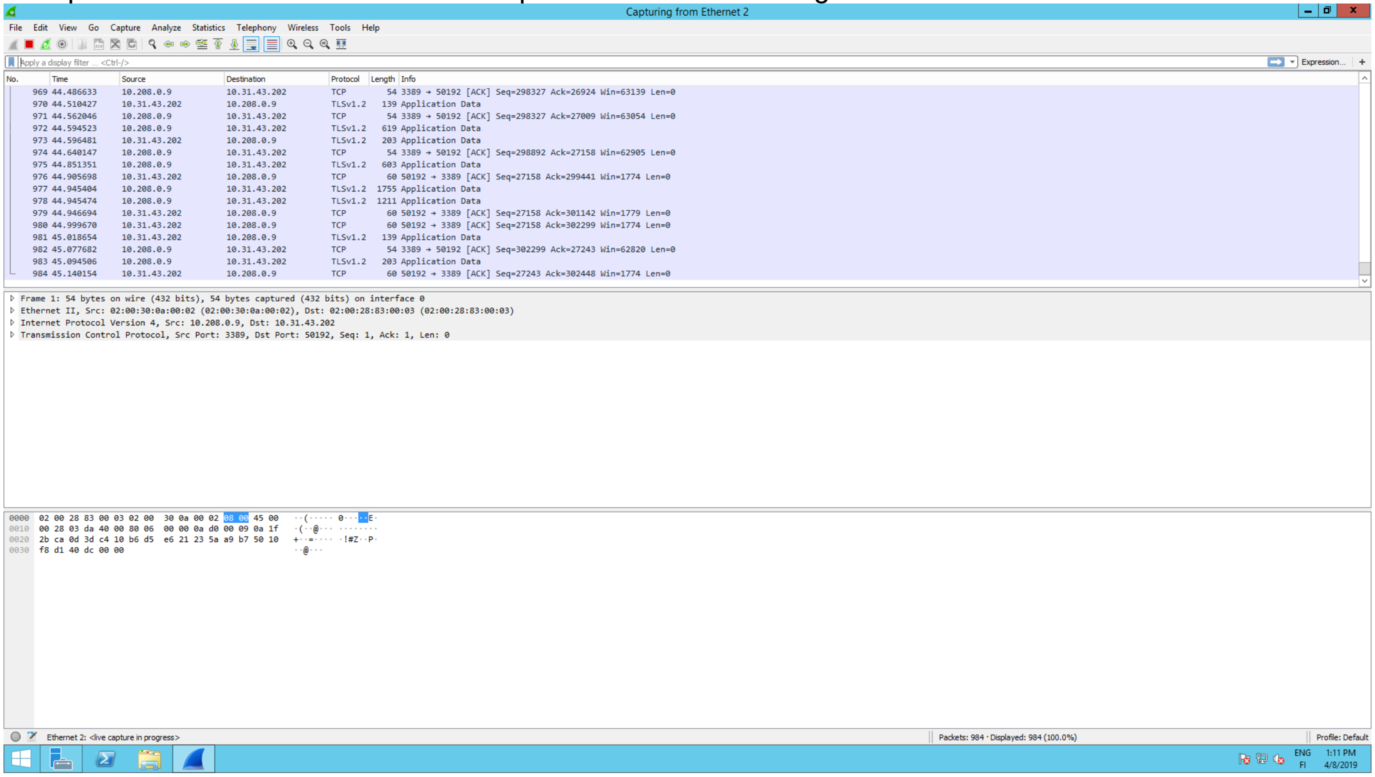
```
10:48:04 PM 5/26/2019

Connection to host lost.
```

3. How and when the data is transferred?

Ans: Once TCP connection is established between client and server them communication takes place between them and data transmissions process takes place.

This picture shows the data transferred process and their timing:

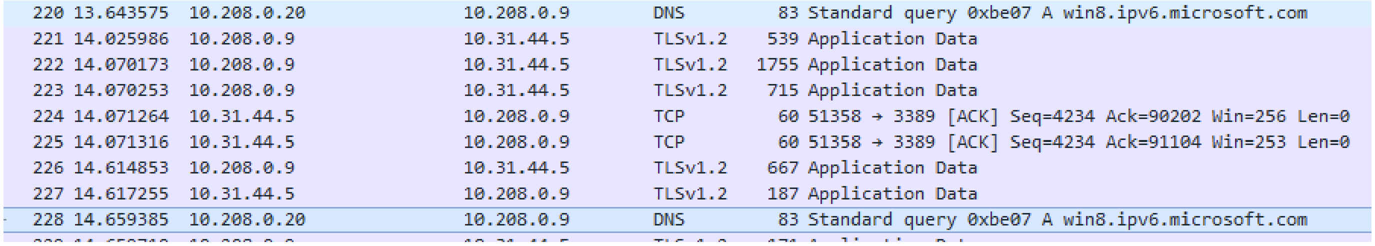

Capturing from Ethernet 2
File
Edit
View
Go
Capture
Analyze
Statistics
Telephony
Wireless
Tools
Help
Apply a display filter ... <Ctrl-/>
Expression...
Ethernet 2: <live capture in progress>
Packets: 984 · Displayed: 984 (100.0%)
Profile: Default
ENG
FI
1:11 PM
4/8/2019

```
No.  Time       Source         Destination    Protocol Length Info
 969 44.486633  10.208.0.9     10.31.43.202   TCP        54 3389 → 50192 [ACK] Seq=298327 Ack=26924 Win=63139 Len=0
 970 44.510427  10.31.43.202   10.208.0.9     TLSv1.2   139 Application Data
 971 44.562046  10.208.0.9     10.31.43.202   TCP        54 3389 → 50192 [ACK] Seq=298327 Ack=27009 Win=63054 Len=0
 972 44.594523  10.208.0.9     10.31.43.202   TLSv1.2   619 Application Data
 973 44.596481  10.31.43.202   10.208.0.9     TLSv1.2   203 Application Data
 974 44.640147  10.208.0.9     10.31.43.202   TCP        54 3389 → 50192 [ACK] Seq=298892 Ack=27158 Win=62905 Len=0
 975 44.851351  10.208.0.9     10.31.43.202   TLSv1.2   603 Application Data
 976 44.905698  10.31.43.202   10.208.0.9     TCP        60 50192 → 3389 [ACK] Seq=27158 Ack=299441 Win=1774 Len=0
 977 44.945404  10.208.0.9     10.31.43.202   TLSv1.2  1755 Application Data
 978 44.945474  10.208.0.9     10.31.43.202   TLSv1.2  1211 Application Data
 979 44.946694  10.31.43.202   10.208.0.9     TCP        60 50192 → 3389 [ACK] Seq=27158 Ack=301142 Win=1779 Len=0
 980 44.999670  10.31.43.202   10.208.0.9     TCP        60 50192 → 3389 [ACK] Seq=27158 Ack=302299 Win=1774 Len=0
 981 45.018654  10.31.43.202   10.208.0.9     TLSv1.2   139 Application Data
 982 45.077682  10.208.0.9     10.31.43.202   TCP        54 3389 → 50192 [ACK] Seq=302299 Ack=27243 Win=62820 Len=0
 983 45.094506  10.208.0.9     10.31.43.202   TLSv1.2   203 Application Data
 984 45.140154  10.31.43.202   10.208.0.9     TCP        60 50192 → 3389 [ACK] Seq=27243 Ack=302448 Win=1774 Len=0

▷ Frame 1: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface 0
▷ Ethernet II, Src: 02:00:30:0a:00:02 (02:00:30:0a:00:02), Dst: 02:00:28:83:00:03 (02:00:28:83:00:03)
▷ Internet Protocol Version 4, Src: 10.208.0.9, Dst: 10.31.43.202
▷ Transmission Control Protocol, Src Port: 3389, Dst Port: 50192, Seq: 1, Ack: 1, Len: 0

0000  02 00 28 83 00 03 02 00  30 0a 00 02 08 00 45 00   ··(·····0·····E·
0010  00 28 03 da 40 00 80 06  00 00 0a d0 00 09 0a 1f   ·(··@···········
0020  2b ca 0d 3d c4 10 b6 d5  e6 21 23 5a a9 b7 50 10   +··=·····!#Z··P·
0030  f8 d1 40 dc 00 00                                  ··@···
```

Also

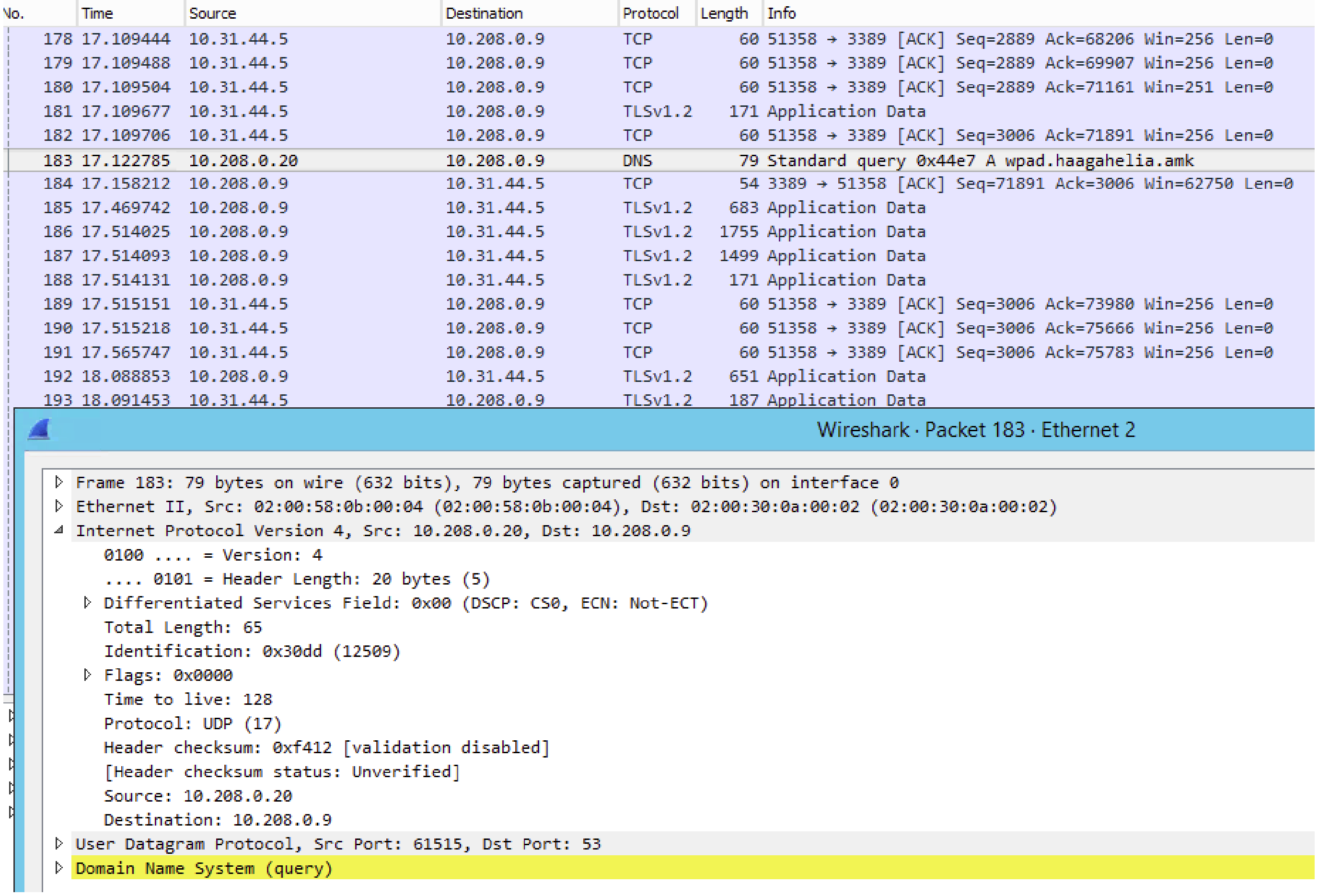


```
220 13.643575  10.208.0.20  10.208.0.9   DNS       83 Standard query 0xbe07 A win8.ipv6.microsoft.com
221 14.025986  10.208.0.9   10.31.44.5   TLSv1.2  539 Application Data
222 14.070173  10.208.0.9   10.31.44.5   TLSv1.2 1755 Application Data
223 14.070253  10.208.0.9   10.31.44.5   TLSv1.2  715 Application Data
224 14.071264  10.31.44.5   10.208.0.9   TCP       60 51358 → 3389 [ACK] Seq=4234 Ack=90202 Win=256 Len=0
225 14.071316  10.31.44.5   10.208.0.9   TCP       60 51358 → 3389 [ACK] Seq=4234 Ack=91104 Win=253 Len=0
226 14.614853  10.208.0.9   10.31.44.5   TLSv1.2  667 Application Data
227 14.617255  10.31.44.5   10.208.0.9   TLSv1.2  187 Application Data
228 14.659385  10.208.0.20  10.208.0.9   DNS       83 Standard query 0xbe07 A win8.ipv6.microsoft.com
```

4. Can you find out the amount of transferred data?

Ans Yes, I can find the amount of data transferred which 79 bytes.

When monitoring TCP connection between WinS1 and WinS2 (client).



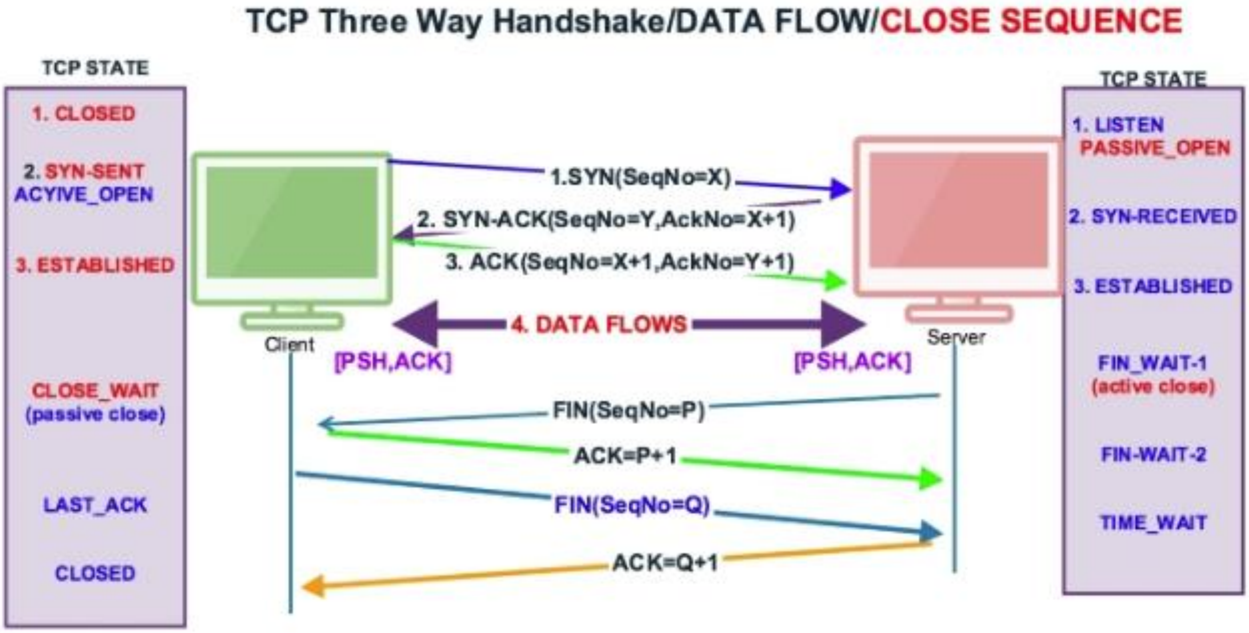
```
No.  Time       Source       Destination  Protocol  Length Info
 178 17.109444  10.31.44.5   10.208.0.9   TCP          60 51358 → 3389 [ACK] Seq=2889 Ack=68206 Win=256 Len=0
 179 17.109488  10.31.44.5   10.208.0.9   TCP          60 51358 → 3389 [ACK] Seq=2889 Ack=69907 Win=256 Len=0
 180 17.109504  10.31.44.5   10.208.0.9   TCP          60 51358 → 3389 [ACK] Seq=2889 Ack=71161 Win=251 Len=0
 181 17.109677  10.31.44.5   10.208.0.9   TLSv1.2     171 Application Data
 182 17.109706  10.31.44.5   10.208.0.9   TCP          60 51358 → 3389 [ACK] Seq=3006 Ack=71891 Win=256 Len=0
 183 17.122785  10.208.0.20  10.208.0.9   DNS          79 Standard query 0x44e7 A wpad.haagahelia.amk
 184 17.158212  10.208.0.9   10.31.44.5   TCP          54 3389 → 51358 [ACK] Seq=71891 Ack=3006 Win=62750 Len=0
 185 17.469742  10.208.0.9   10.31.44.5   TLSv1.2     683 Application Data
 186 17.514025  10.208.0.9   10.31.44.5   TLSv1.2    1755 Application Data
 187 17.514093  10.208.0.9   10.31.44.5   TLSv1.2    1499 Application Data
 188 17.514131  10.208.0.9   10.31.44.5   TLSv1.2     171 Application Data
 189 17.515151  10.31.44.5   10.208.0.9   TCP          60 51358 → 3389 [ACK] Seq=3006 Ack=73980 Win=256 Len=0
 190 17.515218  10.31.44.5   10.208.0.9   TCP          60 51358 → 3389 [ACK] Seq=3006 Ack=75666 Win=256 Len=0
 191 17.565747  10.31.44.5   10.208.0.9   TCP          60 51358 → 3389 [ACK] Seq=3006 Ack=75783 Win=256 Len=0
 192 18.088853  10.208.0.9   10.31.44.5   TLSv1.2     651 Application Data
 193 18.091453  10.31.44.5   10.208.0.9   TLSv1.2     187 Application Data
```

```
▷ Frame 183: 79 bytes on wire (632 bits), 79 bytes captured (632 bits) on interface 0
▷ Ethernet II, Src: 02:00:58:0b:00:04 (02:00:58:0b:00:04), Dst: 02:00:30:0a:00:02 (02:00:30:0a:00:02)
◢ Internet Protocol Version 4, Src: 10.208.0.20, Dst: 10.208.0.9
     0100 .... = Version: 4
     .... 0101 = Header Length: 20 bytes (5)
   ▷ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
     Total Length: 65
     Identification: 0x30dd (12509)
   ▷ Flags: 0x0000
     Time to live: 128
     Protocol: UDP (17)
     Header checksum: 0xf412 [validation disabled]
     [Header checksum status: Unverified]
     Source: 10.208.0.20
     Destination: 10.208.0.9
▷ User Datagram Protocol, Src Port: 61515, Dst Port: 53
▷ Domain Name System (query)
```

5. How the connection is closed?

Ans: TCP connection is normally close appears when the client or server decides that all data has been sent to the receiver and we can close the connection. There are three ways a TCP connection is closed:

1. The client initiates closing the connection by sending a FIN packet to the server.
2. The server initiates closing the connection by sending a FIN packet to the client.
3. Both client and server initiates closing the connection termination independently as TCP being duplex.


TCP Three Way Handshake/DATA FLOW/CLOSE SEQUENCE
TCP STATE
1. CLOSED
2. SYN-SENT
ACYIVE_OPEN
3. ESTABLISHED
CLOSE_WAIT
(passive close)
LAST_ACK
CLOSED
Client
1.SYN(SeqNo=X)
2. SYN-ACK(SeqNo=Y,AckNo=X+1)
3. ACK(SeqNo=X+1,AckNo=Y+1)
4. DATA FLOWS
[PSH,ACK]
[PSH,ACK]
FIN(SeqNo=P)
ACK=P+1
FIN(SeqNo=Q)
ACK=Q+1
Server
TCP STATE
1. LISTEN
PASSIVE_OPEN
2. SYN-RECEIVED
3. ESTABLISHED
FIN_WAIT-1
(active close)
FIN-WAIT-2
TIME_WAIT