Contents
The TCP/IP Guide
9 TCP/IP Lower-Layer (Interface, Internet and Transport) Protocols (OSI Layers 2, 3 and 4)
9 TCP/IP Transport Layer Protocols
9 Transmission Control Protocol (TCP) and User Datagram Protocol (UDP)
9 TCP/IP Transmission Control Protocol (TCP)
9 TCP Basic Operation: Connection Establishment, Management and Termination

Prev. Topic TCP Basic Operation: Connection Establishment, Management and Termination

Prev. Page Pages 1 2 3 Next Page

TCP Connection Preparation: Transmission Control Blocks (TCBs) and Passive and Active Socket *OPENs* Next Topic

TCP Operational Overview and the TCP Finite State Machine (FSM)

(Page 2 of 3)

The Simplified TCP Finite State Machine

In the case of TCP, the finite state machine can be considered to describe the "life stages" of a connection. Each connection between one TCP device and another begins in a null state where there is no connection, and then proceeds through a series of states until a connection is established. It remains in that state until something occurs to cause the connection to be closed again, at which point it proceeds through another sequence of transitional states and returns to the closed state.

The full description of the states, events and transitions in a TCP connection is lengthy and complicated—not surprising, since that would cover much of the entire TCP standard. For our purposes, that level of detail would be a good cure for insomnia but not much else. However, a ***simplified*** look at the TCP FSM will help give us a nice overall feel for how TCP establishes connections and then functions when a connection has been created.

AdChoices

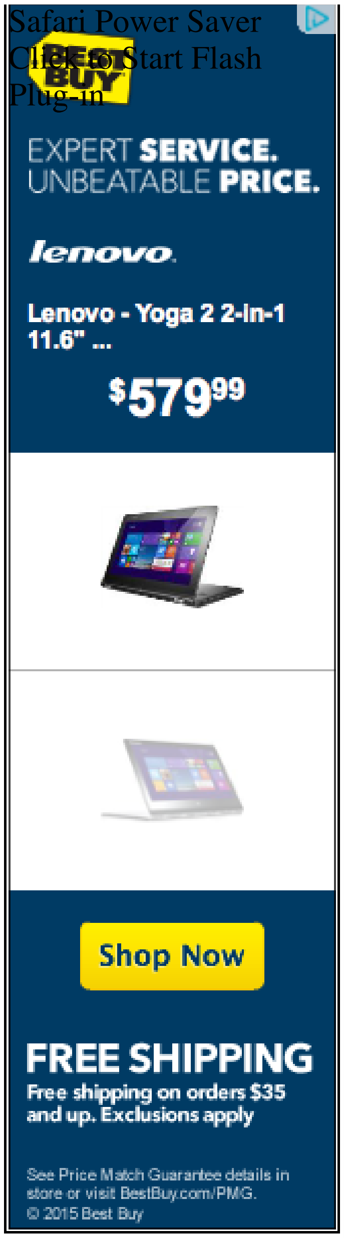
Table 151 briefly describes each of the TCP states in a TCP connection, and also describes the main events that occur in each state, and what actions and transitions occur as a result. For brevity, three abbreviations are used for three types of message that control transitions between states, which correspond to the TCP header flags that are set to indicate a message is serving that function. These are:

- ***SYN:*** A *synchronize* message, used to initiate and establish a connection. It is so named since one of its functions is to synchronizes sequence numbers between devices.
- ***FIN:*** A *finish* message, which is a TCP segment with the *FIN* bit set, indicating that a device wants to terminate the connection.
- ***ACK:*** An *acknowledgment*, indicating receipt of a message such as a *SYN* or a *FIN*.

Again, I have not shown every possible transition, just the ones normally followed in the life of a connection. Error conditions also cause transitions but including these would move us well beyond a "simplified" state machine. The FSM is also illustrated in Figure 210, which you may find easier for seeing how state transitions occur.

Table 151: TCP Finite State Machine (FSM) States, Events and Transitions

State	State Description	Event and Transition
CLOSED	This is the default state that each connection starts in before the process of establishing it begins. The state is called "fictional" in the standard. The reason is that this state represents the situation where there is no connection between devices—it either hasn't been created yet, or has just been destroyed. If that makes sense. J	**Passive Open:** A server begins the process of connection setup by doing a passive open on a TCP port. At the same time, it sets up the data structure (transmission control block or TCB) needed to manage the connection. It then transitions to the *LISTEN* state.
		Active Open, Send *SYN*: A client begins connection setup by sending a *SYN* message, and also sets up a TCB for this connection. It then transitions to the *SYN-SENT* state.
		Receive Client *SYN*, Send *SYN+ACK*: The server device receives a *SYN* from


Safari Power Saver
Click to Start Flash Plug-in
BEST BUY
EXPERT SERVICE. UNBEATABLE PRICE.
lenovo
Lenovo - Yoga 2 2-in-1 11.6" ...
$579.99
Shop Now
FREE SHIPPING
Free shipping on orders $35 and up. Exclusions apply
See Price Match Guarantee details in store or visit BestBuy.com/PMG.
© 2015 Best Buy

LISTEN	A device (normally a server) is waiting to receive a *synchronize* (*SYN*) message from a client. It has not yet sent its own *SYN* message.	a client. It sends back a message that contains its own *SYN* and also acknowledges the one it received. The server moves to the *SYN-RECEIVED* state.
SYN-SENT	The device (normally a client) has sent a *synchronize* (*SYN*) message and is waiting for a matching *SYN* from the other device (usually a server).	**Receive *SYN*, Send *ACK*:** If the device that has sent its *SYN* message receives a *SYN* from the other device but not an *ACK* for its own *SYN*, it acknowledges the *SYN* it receives and then transitions to *SYN-RECEIVED* to wait for the acknowledgment to its *SYN*.
		Receive *SYN+ACK*, Send *ACK*: If the device that sent the *SYN* receives both an acknowledgment to its *SYN* and also a *SYN* from the other device, it acknowledges the *SYN* received and then moves straight to the *ESTABLISHED* state.
SYN-RECEIVED	The device has both received a *SYN* (connection request) from its partner and sent its own *SYN*. It is now waiting for an *ACK* to its *SYN* to finish connection setup.	**Receive *ACK*:** When the device receives the *ACK* to the *SYN* it sent, it transitions to the *ESTABLISHED* state.
ESTABLISHED	The "steady state" of an open TCP connection. Data can be exchanged freely once both devices in the connection enter this state. This will continue until the connection is closed for one reason or another.	**Close, Send *FIN*:** A device can close the connection by sending a message with the *FIN* (*finish*) bit sent and transition to the *FIN-WAIT-1* state.
		Receive *FIN*: A device may receive a *FIN* message from its connection partner asking that the connection be closed. It will acknowledge this message and transition

		to the *CLOSE-WAIT* state.
CLOSE-WAIT	The device has received a close request (*FIN*) from the other device. It must now wait for the application on the local device to acknowledge this request and generate a matching request.	**Close, Send *FIN***: The application using TCP, having been informed the other process wants to shut down, sends a close request to the TCP layer on the machine upon which it is running. TCP then sends a *FIN* to the remote device that already asked to terminate the connection. This device now transitions to *LAST-ACK*.
LAST-ACK	A device that has already received a close request and acknowledged it, has sent its own *FIN* and is waiting for an *ACK* to this request.	**Receive *ACK* for *FIN***: The device receives an acknowledgment for its close request. We have now sent our *FIN* and had it acknowledged, and received the other device's *FIN* and acknowledged it, so we go straight to the *CLOSED* state.
FIN-WAIT-1	A device in this state is waiting for an *ACK* for a *FIN* it has sent, or is waiting for a connection termination request from the other device.	**Receive *ACK* for *FIN***: The device receives an acknowledgment for its close request. It transitions to the *FIN-WAIT-2* state.
		Receive *FIN*, Send *ACK*: The device does not receive an *ACK* for its own *FIN*, but receives a *FIN* from the other device. It acknowledges it, and moves to the *CLOSING* state.
FIN-WAIT-2	A device in this state has received an ACK for its request to terminate the connection and is now waiting for a matching *FIN* from the other device.	**Receive *FIN*, Send *ACK***: The device receives a *FIN* from the other device. It acknowledges it and moves to the *TIME-WAIT* state.
CLOSING	The device has received a *FIN* from the other device and sent an *ACK* for it, but	**Receive *ACK* for *FIN***: The device receives an acknowledgment for its close

	not yet received an *ACK* for its own *FIN* message.	request. It transitions to the *TIME-WAIT* state.
TIME-WAIT	The device has now received a *FIN* from the other device and acknowledged it, and sent its own *FIN* and received an *ACK* for it. We are done, except for waiting to ensure the ACK is received and prevent potential overlap with new connections. (See the topic describing connection termination for more details on this state.)	**Timer Expiration:** After a designated wait period, device transitions to the *CLOSED* state.

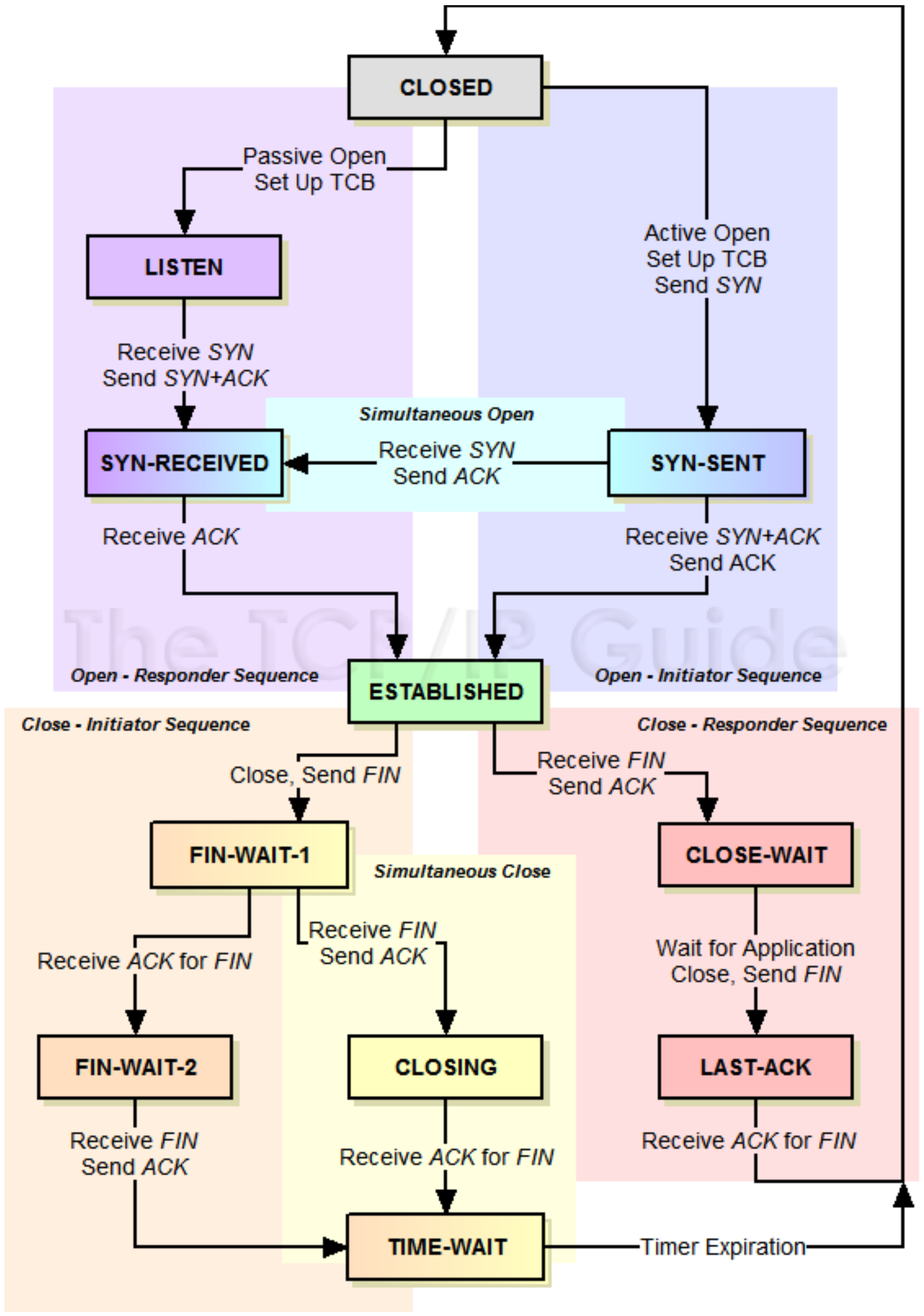


Figure 210: The TCP Finite State Machine (FSM)

This diagram illustrates the simplified TCP FSM. The color codings are not an official part of the definition of the FSM; I have added them to show more clearly the sequences taken by the two devices to open and close a link. For both establishment and termination there is a regular sequence, where the initiating and responding devices go through different states, and a *simultaneous* sequence where each uses the same sequence.

Tap tap… still awake? Okay, I guess even with serious simplification, that FSM isn't all that simple. It may seem a bit intimidating at first, but if you take a few minutes with it, you can get a good handle on how TCP works. The FSM will be of great use in making sense of the connection establishment and termination processes later in this section—and conversely, reading those sections will help you make sense of the FSM. So if your eyes have glazed over completely, just carry on and try coming back to this topic later.

Prev. Topic TCP Basic Operation: Connection Establishment, Management and Termination

Prev. Page Pages 1 2 3 Next Page

TCP Connection Preparation: Transmission Control Blocks (TCBs) and Passive and Active Socket *OPENs* Next Topic

Home - Table Of Contents - Contact Us