1 Introduction to TCP connection management (Level 1)

1.1 Introduction

When an application process in a client host seeks a reliable data connection with a process in another host (say, server), the client-side TCP then proceeds to establish a TCP connection with the TCP at the server side. A TCP connection is a point-to-point, full-duplex logical connection with resources allocated only in the end hosts. The TCP connection between the client and the server is established in the following manner and is illustrated in Figure 1-1.

- 1. The TCP at the client side first sends a special TCP segment, called the SYN packet, to the TCP at the server side.
- Upon receiving the SYN packet, the server allocates TCP buffer and variables to the connection. Also, the server sends a connection-granted segment, called the SYN-ACK packet, to the TCP at the client side.
- 3. Upon receiving the SYN-ACK segment, the client also allocates buffers and variables to the connection. The client then acknowledges the server's connection granted segment with an ACK of its own.

This connection establishment procedure is often referred to as the three-way handshake. The special TCP segments can be identified by the values in the fields SYN, ACK and FIN in the TCP header (see Figure 1-2). We also note that the TCP connection is uniquely identified by the source and destination port numbers (see Figure 1-2) exchanged during TCP connection establishment and the source and destination IP addresses.

Once a TCP connection is established, the application processes can send data to each other. The TCP connection can be terminated by either of the two processes. Suppose that the client application process seeks to terminate the connection. Then, the following handshake ensures that the TCP connection is torn down.

- The TCP at the client side sends a special TCP segment, called the FIN packet, to the TCP at the server side.
- 2. When the server receives the FIN segment, it sends the client an acknowledgement segment in return and its own FIN segment to terminate the full-duplex connection.

3. Finally, the client acknowledges the FIN-ACK segment (from the server) with an ACK of its own. At this point, all the resources (i.e., buffers and variables) in the two hosts are deallocated.

During the life of a TCP connection, the TCP protocol running in each host makes transitions through various TCP states. Figure 1-1 illustrates the typical TCP states visited by the client and the server during connection establishment and data communication.

TCP is defined in RFCs 793, 1122, 7323 and, 2018. A recommended textbook reference for TCP is Chapter 3: Transport layer, of Computer Networking: A top-down approach, by James Kurose and Keith Ross (Pearson).



Figure 1-1: TCP connection establishment between a client and a server

◄ 32 bits													
Source port #									Destination port #				
Sequence number													
Acknowledgement number													
Header length	Unused	C W R	E C E	U R G	A C K	P S H	R S T	S Y N	F I N	Receive window			
	Inter	net	cheo	cksu	m	Urgent data pointer							
Options													
Data													

Figure 1-2: TCP Header

1.2 Network Setup

Open NetSim and click on **Experiments >Internetworks> TCP> Introduction to TCP connection management** then click on the tile in the middle panel to load the example as shown in below Figure 1-3.

NetSim Home NetSim Star Network Simulation/ Version 13.2.35 (64 B	n dard /Emulation Pla Bit)	atform					
			Sear	n Q 🛱			
New Simulation	Ctrl+N	Experiments	Introduction to TCP connection management	Results			
Your Work Examples Experiments	Ctrl+O	Internetion(s Network Performance Network Performance WirFil Routing and Switching Top Introduction to TCP connection management III Related data transfer with TCP DI Mathematical model of TCP throughput performance III TCP Congestion Control Algorithms III Advanced: TCP BIC Congestion control algorithm III Advanced: Routing JOT-VSN S SN R S SN R	When an application process in a client host seeks a reliable data connection with a process in another host, the client-side TCP then proceeds to establish a TCP connection. This connection establishment procedure is often referred to as the three- wy handhake. The special TCP segments used in the handhake can be identified by the values in the fields SVN, ACX and FIN in the TCP header. TCP connection establishment FTP Application with FTB application with FTB segment to be the second by the	1 App1,779 Wred bade,1 2 2 Vred bade,2 3 2 4 Router,3 Router,4			
License Settings Exit	Alt+F4	Cellular Networks Egacy Networks Legacy Networks Legar networking concepts through simulation experiments. Documentation come middle panel to load the simulation. Click on the book icon on the left (Experiment	es with objective, theory, set-up, results, and discussion. Expand and click on the file name to display exp (b) panel to view documentation (pdf).	Called The STY packet, Ster (L-7 at the street about 4. Upon recoving the STY packet, Ster server abouts STOP buffer and vaniables to the connection. Also, the sterver sends a connection-granited segment, called the STVA CKS pack to the TCTP at the client side. Upon recoving the STVA CKS segment, the client aboutances buffers and vaniables to the connection. The client then acknowledges the server's connection granted segment with an ACK of its own.			
Support		Learn	Documentation Conta	tt Us			
Answer/FAQ Contact Technical Su	pport	Videos Experiment Manual	User Manual Email Technology Libraries Phone	- +91 767 605 4321			
Email currentilitat			Source Code Help Websit	e : www.tetcos.com			

Figure 1-3: List of scenarios for the example of Introduction to TCP connection management

NetSim UI displays the configuration file corresponding to this experiment as shown below Figure 1-4.



Figure 1-4: Network set up for studying the Introduction to TCP connection management

1.3 Procedure

The following set of procedures were done to generate this sample.

Step 1: A network scenario is designed in NetSim GUI comprising of 2 Wired Nodes and 2 Routers in the "**Internetworks**" Network Library.

Step 2: In the General Properties of Wired Node 1 i.e., Source, Wireshark Capture is set to Online. Transport Layer properties Congestion plot is set to true.

NOTE: Accept default properties for Routers as well as the Links.

Step 3: Right-click the link ID (of a wired link) and select Properties to access the link's properties. Set Max Uplink Speed and Max Downlink Speed to **10** Mbps. Set Uplink BER and Downlink BER to **0**. Set Uplink Propagation Delay and Downlink Propagation Delay as **100** microseconds for the links 1 and 3 (between the Wired Node's and the routers). Set Uplink Propagation Delay and Downlink Propagation Delay as **50000** microseconds for the backbone link connecting the routers, i.e., 2.

Step 4: Right click on the Application Flow **App1 FTP** and select Properties or click on the Application icon present in the top ribbon/toolbar.

An FTP Application is generated from Wired Node 1 i.e., Source to Wired Node 2 i.e., Destination with File Size set to 14600 Bytes and File Inter Arrival Time set to 10 Seconds.

Step 5: Click on Display Settings > Device IP check box in the NetSim GUI to view the network topology along with the IP address.

Step 6: Enable the plots and click on Run simulation. The simulation time is set to 10 seconds.

1.4 Output

We have enabled Wireshark capture in Wired Node 1. The PCAP file is generated at the end of the simulation and is shown in Figure 1-5.

	(*\\.\pij	pe\WIRED	_NODE_1_1													_		×
Fi	ile Edit	View	Go Capture	e Analyze	Statistics	Telephony	Wireless	Tools	Help									
			. 🔝 🗙 🖾	Q 🦛 🖻	. 👳 🕢		⊕ ⊝ ∈											
				• •	- = •			.									_	٦.
	Apply a	display filte	er <ctrl-></ctrl->															<u> </u>
No		Time	Sour	ce	1	Destination		Protoco	l Length	Info								^
	11	0.20615	0 11.	3.1.2		11.1.1.2		TCP	40	36934	+ → 82	[ACK]	Seq=1	Ack=	2921	Win=4	381 L	
	12	0.20615	io 11.	1.1.2	:	11.1.1.1		TCP	1500	82 →	36934	[<non< th=""><th>e>] Se</th><th>q=730</th><th>)1 Wir</th><th>1=7300</th><th>Len=</th><th></th></non<>	e>] Se	q=730)1 Wir	1=7300	Len=	
	13	0.20615	io 11.	1.1.2	:	11.1.1.1		TCP	1500	82 →	36934	[<non< th=""><th>e>] Se</th><th>q=876</th><th>51 Wir</th><th>1=7300</th><th>Len=</th><th></th></non<>	e>] Se	q=876	51 Wir	1=7300	Len=	
	14	0.20737	1 11.	3.1.2		11.1.1.2		TCP	40	36934	1 → 82	[ACK]	Seq=1	Ack=	4381	Win=4	381 L	
	15	0.20737	1 11.	1.1.2	:	11.1.1.1		TCP	1500	82 →	36934	[<non< th=""><th>e>] Se</th><th>q=102</th><th>21 Wi</th><th>n=876</th><th>0 Len</th><th></th></non<>	e>] Se	q=102	21 Wi	n=876	0 Len	
	16	0.20737	1 11.	1.1.2	:	11.1.1.1		TCP	1500	82 →	36934	[<non< th=""><th>e>] Se</th><th>q=116</th><th>681 Wi</th><th>n=876</th><th>0 Len</th><th></th></non<>	e>] Se	q=116	681 Wi	n=876	0 Len	
	17	0.30910	07 11.	3.1.2	:	11.1.1.2		TCP	40	36934	1 → 82	[ACK]	Seq=1	Ack=	5841	Win=4	381 L	
	18	0.30910	07 11.	1.1.2	:	11.1.1.1		TCP	1500	82 →	36934	[<non< th=""><th>e>] Se</th><th>q=131</th><th>.41 Wi</th><th>n=102</th><th>20 Le</th><th></th></non<>	e>] Se	q=131	.41 Wi	n=102	20 Le	
	19	0.31032	.9 11.	3.1.2	:	11.1.1.2		TCP	40	36934	1 → 82	[ACK]	Seq=1	Ack=	:7301	Win=4	381 L	
	20	0.31155	11.	3.1.2	:	11.1.1.2		TCP	40	36934	+ → 82	[ACK]	Seq=1	Ack=	8761	Win=4	381 L	
	21	0.31277	⁷ 2 11.	3.1.2	:	11.1.1.2		TCP	40	36934	+ → 82	[ACK]	Seq=1	Ack=	10221	Win=	4381	
	22	0.31399	11.	3.1.2	:	11.1.1.2		TCP	40	36934	+ → 82	[ACK]	Seq=1	Ack=	11681	Win=	4381	
	23	0.31521	.6 11.	3.1.2	:	11.1.1.2		TCP	40	36934	+ → 82	[ACK]	Seq=1	Ack=	:13141	Win=	4381	
	24	0.41328	6 11.	3.1.2		11.1.1.2		TCP	40	36934	+ → 82	[ACK]	Seq=1	Ack=	:14601	Win=	4381	
	25	0.41328	36 11.	1.1.2		11.1.1.1		TCP	40	82 →	36934	[FIN]	Seq=1	4601	Win=1	8980	Len=0	
	26	0.51396	1 11.	3.1.2		11.1.1.2		TCP	40	36934	+ → 82	[FIN,	ACK]	Seq=1	. Ack=	:14601	Win=	
ш	27	0.51396	51 11.	1.1.2		11.1.1.1		TCP	40	LTCP	Retrar	ismiss	ion] 8	2 → 3	36934	[FIN]	Seq=	
	28	0.51401	.5 11.	3.1.2		11.1.1.2		тср	40	36934	+ → 82	[FIN]	Seq=2	Win=	4381	Len=0		
Ľ	29	0.51401	.5 11.	1.1.2		11.1.1.1		TCP	40	82 →	36934	[ACK]	Seq=1	4602	Ack=:	Win=	18980	×
<																	>	
>	Frame	27: 40	bytes on wi	re (320 b	its), 40	bytes capt	ured (32	0000	45 00 00	28 0	0 00 0	0 00	ff 06	a3 c	b Øb	01 01	02	E · · (· ·
	Raw pa	icket da	ta					0010	0b 01 01	L 01 0	0 52 9	0 46	00 00	39 0	a 00	00 00	00	
>	Intern	et Prot	ocol Versio	on 4, Src:	11.1.1.2	2, Dst: 11.	1.1.1	0020	50 01 4a	a 24 8	4 18 0	00 00						P•J\$•
>	Transm	ission	Control Pro	otocol, Sr	c Port: 8	32, Dst Por	t: 36934											

Figure 1-5: Wireshark Packet capture at Wired_Node_1

- 1. The 3-way handshake of TCP connection establishment and TCP connection termination is observed in the packet capture (Figure 1-5).
- 2. Data is transferred only after the TCP connection is established.
- 3. We can access the packet header details of the TCP segments (SYN, SYN-ACK, FIN, FINACK) in Wireshark.