MATRIX Academic International Online Journal of Engineering and Technology
ISSN: 2348–3326 © MAIOJET
Volume III Issue I, April 2015

Bespoke Timers in Improved Transmission Control Protocol Model

Ashwani Parashar [#1] Ankit Saxena[*2], Dr. Swapnesh Taterh [#3]

[1]Research Scholar, ashwani.parashar@gmail.com
[2]Research Scholar ankitprg@gmail.com
[3]Amity University, swapnesh@hotmail.com

***Abstract*—Transmission Control Protocol is a key protocol in TCP-IP Protocol Suite. Transmission Control Protocol is responsible for delivery of data to the application. The timers in Transmission Control Protocol ensure that the data is delivered and acknowledged to the nodes with in time. This paper describes bespoke Timers in Improved Transmission Control Protocol Model that ensure that the data is delivered efficiently among peers.**

Keywords—ITCP; Timers; Retransmission; Timeout;

I. INTRODUCTION

The Transmission Control Protocol Model ensures that data delivered to the receiver is acknowledged. The sequence number is associated with the data delivered to the node. The acknowledgement number, one more than sequence number ensures that the data identified by sequence number is delivered to the node. The data which is not received or not acknowledged with in time is sent again. The window is maintained by end hosts which contains details of packets sent and aacknowledged, packet sent but not yet acknowledged and packets yet to be sent (Figure 1).

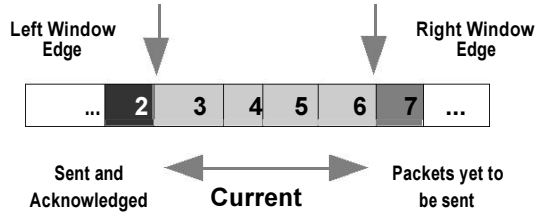

Figure 1:- The Sender's windows showing which packets are Sent and Acknowledged and which packets are sent and not acknowledged and which packets are not sent yet.

Each TCP header [1] contains the source and destination port number. The four tuples(source port, destination port, source address, destination address) uniquely identify each connection. The Sequence Number field identifies the segment. Acknowledgement Number field contains the Acknowledgement Number=Sequence number+1, Sequence number is that number that is received from the sender of the segment and Acknowledgement number is that number sent in the segment from receiver to sender acknowledging the receipt. The TCP Header Length field is 20 bytes in size. Eight flags bit field are defined in TCP header which are as follows (Figure 2):-

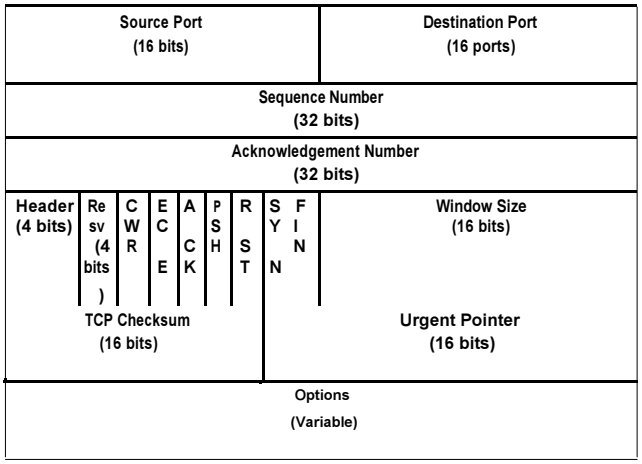

Figure 2 :- TCP Header. TCP Header fixed size is 20 bytes and TCP option maximum field size (variable) is 40 bytes.

CWR—Congestion Window Reduced (the sender reduced its sending rate);

ECE—ECN Echo (the sender received an earlier congestion notification);

URG—Urgent ;

ACK—Acknowledgment(ACK bit is set if current segment has to be acknowledged by its peer)

PSH—Push (the receiver should pass this data to the application as soon as possible);

RST—Reset the connection (connection abort);

SYN—Synchronize sequence numbers to initiate a connection;

FIN—The sender of the segment is finished sending data to its peer;

The Windows field (16 bits) specify that the sender of the segment is willing to accept. The checksum (16 bits) is the checksum of TCP header and options field. TCP Options (variable) maximum field size is 40 bytes. TCP option field may consists of single octet of option kind or octet of option kind , an octet of option-length, and the actual option-data octets. The option-length counts the one octet each of option-kind and option-length as well as the number of option-data octets. The Section II describes Improved Transmission Control Protocol Model (ITCP), section III provides details of bespoke timers of ITCP, Section IV discuss Analysis and section V provides conclusion.

II. IMPROVED TRANSMISSION CONTROL PROTOCOL MODEL (ITCP)

TCP connection consists of four tuple comprising source address, source port, destination address & destination port. TCP connection consists of set up, data setup and tear down. phase. The host that sent first SYN,ACK packet performs active open (Client) and the other host that receives the SYN packet performs passive open. This is different from simultaneous open where both host can send SYN,ACK packets and perform as active open.

The connection setup consists of three steps (Figure 3) which are described as follows:-

1. The client (Active opener) sends the first segment SYN,ACK (SYN,ACK bit set) with Sequence number Seq=ISN(c). Here, SYN,ACK bits will be set because client is expecting Acknowledgement from the Server(Passive opener).

2. The Server (Passive Opener) sends the second segment SYN,ACK (SYN,ACK bit set) with new Sequence number Seq=ISN(s) and Acknowledgement Number ACK=ISN(c)+1. Here, SYN,ACK bit is set because Server (Passive Opener) is expecting Acknowledgement from the Client for the second segment.

3. The client sends the third segment with new Sequence Number Seq=ISN(c)+1 and Acknowledgement Number ACK=ISN(s)+1. This completes the connection setup phase. This is also called Three way Handshake.

In Data Transfer phase, Data transfer takes place.

In Tear down phase, the TCP connection closes in three phases. Either side (Client or Server) can initiate the closing operation. Here, the client is initiating closing operation, hence performs as active closer and server performs as passive closer. It can also happen that only one side (either client or server) closes the connection, and the other side continues to transfer the data , then the connection performs the operation called half close. Another half close from other side closes the connections completely. The three phases in closing the connection in tear down phase are as follows:

1. The client (Active Closer) sends FIN,ACK segment (FIN and ACK bits are set). FIN,ACK bits is set because client is expecting acknowledgement of FIN,ACK segment from server. Seq=K, ACK=L are sent with FIN,ACK segment from Client to Server.

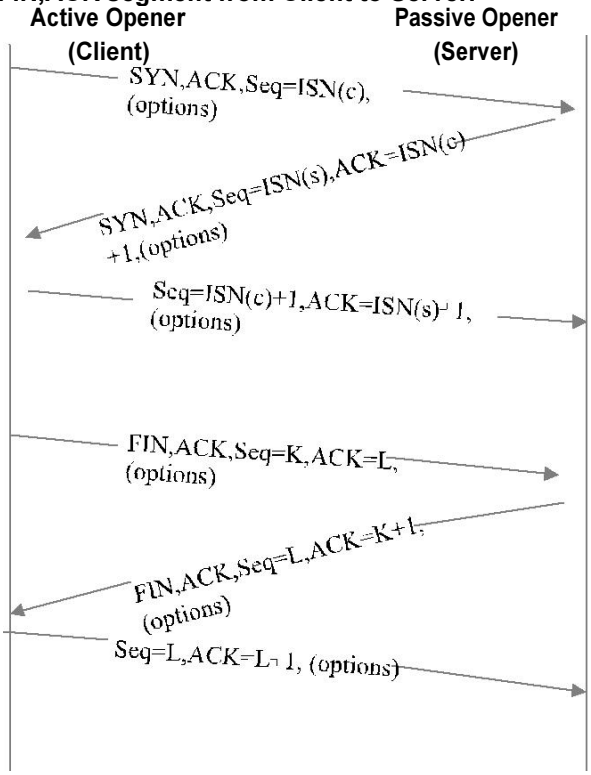

Figure 3. A TCP connection establishment and termination. The client initiates a three-way handshake to exchange initial sequence numbers for the client and server. The connection terminates after each side has sent a FIN and received an acknowledgment for it.

2. The Server (Passive Closer) sends FIN,ACK segment (FIN and ACK bits are set). FIN,ACK bits is set because server is expecting acknowledgement of

FIN,ACK segment sent from Server (Passive Closer) to Client(Active Closer). New Sequence Number Seq=L is given to segment and FIN,ACK segment from client to server is acknowledged with Acknowledgement ACK=K+1.

3. **New Sequence Number Seq=L is given for third segment ACK from client (Active Closer) to Server (Passive Closer) and acknowledgement number ACK=L+1 is set to acknowledge the second segment from Passive Closer (Server) to Active Closer (Client). The connection is closed completely when Server(Passive Closer) receives this acknowledgement.**

This completes the tear down phase.

III. Bespoke TIMERS OF IMPROVED TRANSMISSION CONTROL PROTOCOL MODEL

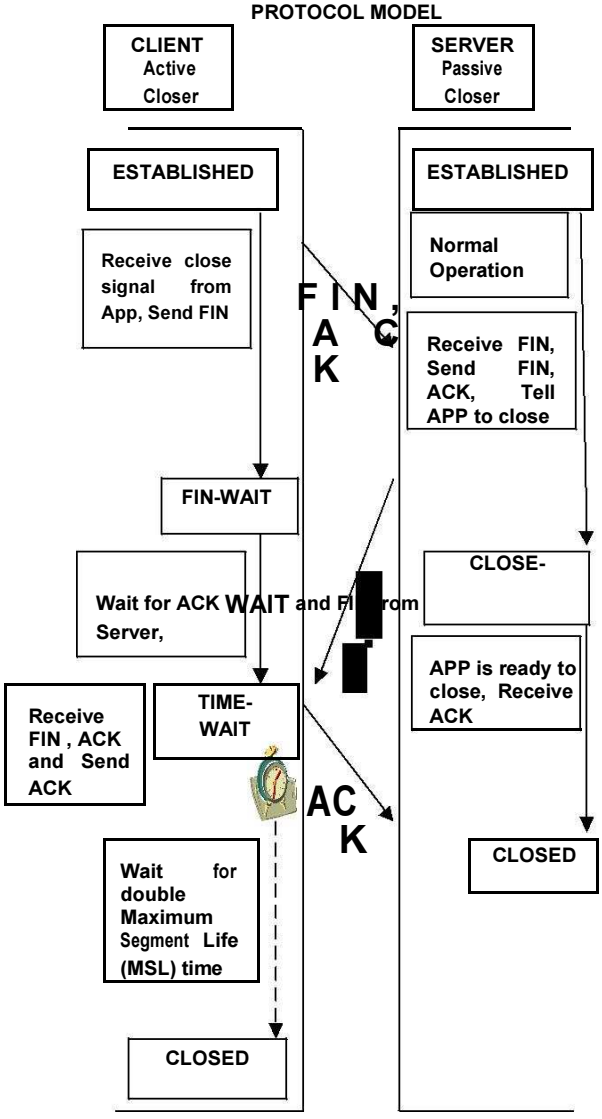

Figure 4:- TCP connection termination flow diagram where connection is terminated by client. Client states 'ESTABLISHED', 'FIN_ WAIT', 'TIME_WAIT' and 'CLOSED' state are shown. Server States ESTABLISHED', 'CLOSE-WAIT' and 'CLOSED' are shown.

FIN_WAIT timer

In Existing Transmission control Protocol Model [2], FIN_WAIT1 and FIN_WAIT2 timer exists. In Improved Transmission Control Protocol Model FIN_WAIT2 timer does not exists. FIN_WAIT timer in Improved Transmission Control Protocol Model waits for FIN segment of Server to Client and acknowledgement of FIN segment sent from client (Active Closer) to server (Passive Closer) (Figure. 4).

TIME_WAIT timer

FIN segment sent from Server(Passive Closer) to Client(Active Closer) is received and ACK of FIN segment is sent from Client(Active Closer) to Server(Passive Closer). Then, connection enters the TIME-WAIT state and a Time-wait timer is started. This is to allow all the segments in transit to be removed from the network. On expiration of the timer, the connection is terminated. (Figure 4).

CLOSE-WAIT timer and CLOSED State

Before CLOSE-WAIT timer is started, Server(Passive Closer) receives FIN segment from Client (Active Closer) with ACK bit set and Send FIN+ACK segment to client. After CLOSE-WAIT timer is started, Connection is CLOSED once the ACK from client(Passive Closer) is received. (Figure 4).

Retransmission Timer

The retransmission timer [3] is initialized to approximately three seconds when a TCP connection is established. However, it is adjusted on the fly to match the characteristics of the connection by using characteristics of the client and server. For example, In general, the CPU is quad core and the RAM of the Server is 10 GB and and client CPU is dual core, the RAM of the client is 2 GB. Hence, the retransmission timer is worked out as the ratio of [(Factor of Server CPU + Factor of Server RAM+Factor of Server Bandwidth)/(Factor of Client CPU + Factor Client RAM+Factor of Client Bandwidth)]. The reason for adjusting the retransmission timer in terms of ratio of Server and Client is that when the client gets connected to the server, the resources allocated to the server is more than the resources allocated for the connection on the client. Therefore, retransmission timer which is related to sending the segment again if the segment is dropped or the acknowledgement of the segment does not arrives before the user time out expires. In the present, Transmission Control Protocol Model , the timers are allocated with no consideration of factors like CPU and the Random Access Memory of the client and server which have key role when deciding for allocation of the resources during the setup of the connection. The timer for a given segment which is set for retransmitted segment is doubled after expiry.

USER TIMEOUT

There are two types of user timeout [4]. One is TCP timeout and second is user application timeout. If user application does not mention its timeout , then TCP timeout is considered. These timeouts are exchange during Open and Send system calls to end node. The default user time out is 5 minutes. The host may also choose timeout by setting thresholds R1 and R2. The TCP informs the user application once the retransmission threshold R1 (3 retransmissions) and threshold R2(100 seconds) is reached. TCP maintains four per-connection state variables to control the operation of the UTO option, three of which are as follows:-

USER_TIMEOUT :- TCP's USER TIMEOUT parameter

ADV_UTO :- UTO option advertised to the remote TCP peer. This is an application-specified value, and may be specified on a system-wide basis. If unspecified, it defaults to the default system-wide USER TIMEOUT.

ENABLED (Boolean) :- Flag that controls whether the UTO option is enabled for a connection. This flag applies to both sending and receiving. Defaults to false.

CHANGEABLE (Boolean) :- Flag that controls whether USER_TIMEOUT (TCP's USER TIMEOUT parameter) may be changed based on an UTO option received from the other end of the connection. Defaults to true and becomes false when an application explicitly sets USER_TIMEOUT. The UTO options that are changed between nodes are not binding and user application may choose to implement other node suggestion if ENABLED is true and CHANGEABLE is true. when CHANGEABLE is true, each end SHOULD compute the local USER TIMEOUT for a connection according to this formula:

USER_TIMEOUT = min(U_ LIMIT, max(ADV_UTO, REMOTE_UTO, L_LIMIT))

USER_TIMEOUT :- USER TIMEOUT value to be adopted by the local TCP for this connection.

U_LIMIT :- Current upper limit imposed on the user timeout of a connection by the local host.

ADV_UTO :- User timeout advertised to the remote TCP peer in a TCP User Timeout Option.

REMOTE_UTO :- Last user timeout value received from the other end in a TCP User Timeout Option.

L_LIMIT:- **Current lower limit imposed on the user timeout of a connection by the local host.**

The user timeout chosen in this way selects the maximum of the advertise value between nodes. The purpose of choosing the user timeout in this way is to choose the maximum utilization time of node. In existing Transmission Control Protocol Model, the user timeout is chosen based on the declared user time out option of the node. The default TCP user timeout is 5 minutes. There is wide gap between resources available between server and client. In the proposed Transmission Control Protocol Model, the default TCP user timeout is calculated based on the available CPU and memory resource on the node between which connection is to be established. The two levels L1 and L2 may be established where 'L1' denotes the best available CPU and memory resources and 'L2 ' denotes the threshold available cpu and memory resources during that last 10 minutes instance of time. The user time out related to the application may be continued to be set for client and server application requirement considering client and server resources available. Thus, rationalizing the TCP user time out and application user time out, the data can be exchanged in best optimized time.

IV. Analysis

TCP connection establishment in Improved Transmission Control Protocol model (Figure 3), ACK bits must be set in first SYN segment so that the acknowledgement of SYN segment from client to server is ensured by server to client. There is unnecessary flow of segment ACK from Server to client in TCP connection termination in existing Transmission Control Protocol model, which increase the traffic flow of data from server to client which can be piggybacked with FIN segment from Server to client. In Improved Transmission Control Protocol Model FIN_WAIT2 timer does not exists. FIN _WAIT timer in Improved Transmission Control Protocol Model waits for FIN segment of Server to Client and acknowledgement of FIN segment sent from client to server. TCP user time out and application user time out must be decided based on client and server CPU and RAM resources and the bandwidth available at client and server for the exchange of data.

V. Conclusion

In Improved Transmission Control Protocol Model, there is no additional flow of segment during connection termination as shown in figure 3. TCP Timers have been improved by considering the actual CPU and RAM resources and bandwidth at client and server for exchange of data in best optimized time.

References

1. **[RFC0793] Postel, J.,"Transmission Control Protocol," Internet RFC 0793, September 1981.**
2. **[RFC1122] Braden, R., "Requirements for Internet Hosts—Communication Layers," Internet RFC 1122, October 1989.**
3. **[RFC6298] Paxson V., Allman M., Chu J., Sargent M., "Computing TCP's Retransmission Timer", June 2011.**
4. **[RFC5482] Eggert L., Gont F., "TCP User Time out option," Internet RFC 5482, March 2009.**
5. **Swapnesh Taterh and Rakesh Jangid. "Security Measurement in Secure Smart**

Card." Published in IEEE International Conference on Advanced Research in Engineering & Technologies. K.L University, Vaddeswaram, Guntur, A.P. Feb 08-09, 2013.

6. **Swapnesh Taterh, Dr. K.P Yadav, Dr. S.K Sharma. "Security Requirement Elicitation Phase of Secure Software Development Life Cycle" Published in International Journal of Research Review in Engineering, Science and Technology. ISSN 2278-6643. Volume 2, Issue 1, March 2013.**