Overview

The purpose of this Lab 3 Assignment is to create a TCP client that will submit transactions requests to the instructor's TCP server, where a response will be generated and returned to the client. The source of the client's request messages may be from a local file created by the student, or dynamically created. Request message and response message formats and contents are defined in later sections of this lab assignment.

Each student is to submit request messages to complete three separate testing scenarios:

1. Scenario 1:

1.1. Purpose:

Complete a function test of client connectivity and simple synchronous transaction processing with the instructor's server.

1.2. Requirements:

1.2.1. Transaction processing mode is to be synchronous.

1.2.2. Client is to transmit 100 request messages formatted and with data as defined in the *Request Message Format* section of this assignment.

1.2.3. Client is to maintain a single perpetual TCP session for the entire 100 request and response messages.

1.2.4. After receipt of the 100th response, client is to perform a half-session shutdown for each half of the TCP session, then close the socket. Status from the shutdowns and close are to be included in the trailer record of the log file and formatted in accordance with specifications defined in the *Trailer Record Format and Data Content* of this assignment.

1.2.5. Client is to populate the "*RequestID*" field in the request message with data of its discretion.

1.2.6. Client is to populate the *ResponseDelay* field in the request message with an ASCII value of zero.

1.2.7. Client is to populate the "ResponseType" field of the response message in accordance with the instructions in the Response Message Format section of this assignment.

1.2.8. Client is to impose a delay of no less than 50 milliseconds between receipt of a response message and the subsequent transmission of the next request message.

1.2.9. Client is to maintain a log file of the request and response messages that will be submitted to instructor. Log file organization is to be compliant with that defined in the *Log Files Organization* section of this assignment.

1.2.10. Log file name: **Lab3.Scenario1.[*last name*][*first name initial*].txt**

1.3. Grading:

1.3.1. 50 Points for successful TCP session establishment and shutdown

1.3.2. 10 points for successful completion of all 100 transactions.

2. Scenario 2:

2.1. Purpose:

To test the accuracy and completeness of matching response messages to request messages when operating in an asynchronous transaction mode where responses may be returned in a sequence different from the arrival sequence of request messages at the server.

2.2. Requirements:

2.2.1. Transaction processing mode is to be asynchronous

2.2.2. Client is to transmit 100 request messages formatted and with data as defined in the *Request Message Format and Contents* section of this assignment. Client is to maintain a single perpetual TCP session for the entire 100 request and response messages.

2.2.3. After receipt of the 100th response, client is to perform a half-session shutdown for each half of the TCP session, then close the socket. Status from the shutdowns and close are to be included in the trailer record of the log file and formatted in accordance with specifications defined in the *Trailer Record Format and Data Content* of this assignment.

2.2.4. Client is to populate the "*RequestID*" field in the request message with information that will enable the client to uniquely and exclusively match a received response to the request.

2.2.5. Client is to populate the *ResponseDelay* field in the request message with a millisecond value that will cause the server to return responses in a sequence other than the request arrival sequence. There must be a minimum of at least two occurrences of a response sequence difference from request submission sequence. Other than for purposes of forcing these response sequence differences, the *ResponseDelay* field in the rest of the request messages may be set to an ASCII zero. This populated value must be in ASCII and ≥ zero and ≤ 3000

2.2.6. Client is to impose a delay of no less than 50 milliseconds between the transmissions of each request message.

2.2.7. Client is to populate the "ResponseType" field of the response message in accordance with the instructions in the Response Message Format section of this assignment.

2.2.8. Client is to maintain a log file of the request and response messages that will be submitted to instructor. Log file organization is to be compliant with that defined in the *Log Files Organization* section of this assignment.

2.2.9. Log file name: **Lab3.Scenario2.[*last name*][*first name initial*].txt**

Grading:

25 points for successful completion of assignment. When there is an incomplete accuracy of matching responses to the correct request, the accuracy rate will be applied to the 25 point maximum.

3. **Scenario 3:**

3.1. **Purpose:**

To test the successful receipt and error recovery from a latent response. Once the client has timed-out while waiting for a response to a specific request message and performed its own stand-in response, the client still needs to expect a latent response from the server. The amount of time a client waits for a response after sending the request before issuing a stand-in response, and the amount of time the client continues to wait for a latent response after sending a stand-in response are usually established by business policies and practices.

For the purposes of this scenario, the client is to wait no longer than three seconds for a response, whereupon the client is to issue a stand-in response. After issuing the stand-in response, the client is to wait an additional twenty seconds for a possible latent response from the server before removing any presence of the pending request from its system.

3.2. Requirements:

3.2.1. Transaction processing mode is to be asynchronous

3.2.2. Client is to transmit 100 request messages formatted and with data as defined in the *Request Message Format and Contents* section of this assignment.

3.2.3. Client is to maintain a single perpetual TCP session for the entire 100 request and response messages.

3.2.4. After receipt of all (there will be more than 100) responses, client is to perform a half-session shutdown for each half of the TCP session, then close the socket. Status from the shutdowns and close are to be included in the trailer record of the log file and formatted in accordance with specifications defined in the *Trailer Record Format and Data Content* of this assignment.

3.2.5. Client is to populate the "*RequestID*" field with information that will enable the client to uniquely and exclusively match a received response to the request.

3.2.6. Client is to populate the "*ResponseDelay*" field of at least two request messages with a millisecond value of an ASCII 4000, to cause a latent response after it has issued an internal stand-in response.

3.2.7. Client must issue a stand-in response after a pending request has waited no less than three seconds and no longer than four seconds for a matching response. Client will log the request and stand-in response and note in the "*ResponseType*" field of the response message that this is a stand-in response.

3.2.8. When client detects a latent response, it must match it to the proper original request and record both to the log file, and note in the "*ResponseType*" field of the response message that this is a latent response.

3.2.9. In the event client receives an unsolicited response (e.g. a latent response after waiting the additional 20 seconds, or simply a spurious response from the server) client is to log the response (w/o its matching request) and note in the "*ResponseType*" field of the response message that this is an unsolicited response.

3.2.10. Client is to maintain a log file of all request and response messages that will be submitted to instructor. Log file organization is to be compliant with that defined in the *Log Files Organization* section of this assignment.

3.2.11. Log file name: **Lab3.Scenario3.[*last name*][*first name initial*].txt**

3.3. **Grading**

15 points for successful completion.

Completion:

Upon submission of all three log files to the instructor, this lab is complete. Please submit log files as they become available, i.e. do not wait to accumulate all three before submitting the first one.

Recommendation:

It will be most useful for you to write your code in modules so that you may re-use, rather than re-invent, them when developing the next lab. My suggestion is that you create a "receive" function separate from a "transmit" function. The receive function may be used by the client to receive a response and used by the server in Lab 4 to receive the request. Also, the transmit function may be used by the client to send a request and used by the server in Lab 4 to transmit a response.

Access To Instructor's Server:

[Note: This procedure may be changed, if changed you will be notified via email and Blackboard]

For those students using a Windows or MAC platform, you may download a remote VPN secure client from NCP Secure Client. Do not download this software until you are ready to being testing because you have only a 30-day period to "test" the software before the license will expire. You need to complete all three scenarios of Lab3 inside this 30-day period

"Please download the NCP Secure Entry Client for Juniper by using the following link http://www.ncp-e.com/en/downloads/software.html and install the client..

I will forward each student a configuration profile for the NCP Secure Client that will enable you to connect to the instructor's server for testing your client. The internal name of the server is "Hagar" and is located at the private IP address of **192.168.101.222**, and you are to use service port 2605. Hagar is available for test 24/7.

On my laptop I have a virtual server that students may connect to rather than using the NCP remote access VPN software. However that access is only available on site and in the class room. I do get to class early, around 3:00 PM, on Thursdays. Please make arrangements with me if you would like to test before class time on a Thursday.

Request Message Format:

Field Name:	TCPHeader
Data type	Binary (Network, Big Endian Bit Order)
Justification:	Right
Length:	2
Begin position:	0
End position	1
Description:	Binary value that represents the length of the request message, excluding these two bytes.

Field Name:	MessageType
Data type:	ASCII Alphanumeric fixed "**REQ**"
Justification:	Any
Length:	3
Begin position:	2
End position:	4
Description:	This is to be a fixed three-character sequence of "**REQ**", all in upper case.

Insert a vertical '|' character at position 5

Field Name:	msTimeStamp
Data type:	ASCII Numeric
Justification:	Right, space or zero filled to the left
Length:	Variable with a maximum length of 10
Begin position:	6

End position:	Variable
Description:	A millisecond time stamp value from a running elapsed system timer on the client

Insert a vertical '|' character in next position as a field separator

Field Name:	RequestID
Data type:	ASCII Alphanumeric
Justification:	Any
Length	Variable with a maximum length of 20
Begin position:	Variable
End position:	Variable
Description:	A unique value among all request messages that is used to exclusively match a response message to it respective request message. This field will be returned by the server in the response message and in the same positions.

Insert a vertical '|' character in next position as a field separator

Field Name:	StudentName
Data type:	ASCII Text
Justification:	Left, space filled to the right
Length:	Variable with a maximum length of 20
Begin position:	Variable
End position:	Variable
Description:	Student's last name followed by first letter of student's first name. for example: "FindleyT" [without quotation marks, upper case is optional].

Insert a vertical '|' character in next position as a field separator

Field Name:	StudentID
Data type:	ASCII Alphanumeric
Justification:	Any
Length:	Variable with a maximum length of 7
Begin position:	Variable
End position:	Variable
Description:	Student's "918" ID number in the form of "dd-dddd"

Field Name:	ResponseDelay
Data type:	ASCII Numeric

Insert a vertical '|' character in next position as a field separator

Justification:	Right, space or zero filled to the left
Length:	Variable with a maximum length of 5
Begin position:	Variable
End position:	Variable
Description:	A numeric value that represents the number of milliseconds the server is to delay before sending a response.

Insert a vertical '|' character in next position as a field separator

Field Name:	ClientIPAddress
Data type:	ASCII Alphanumeric

Justification:	Left, space filled to the right
Length:	Variable with a maximum length of 15
Begin position:	Variable
End position:	Variable
Description:	The IPv4 address of the client in dotted decimal notation

Insert a vertical '|' character in next position as a field separator

Field Name:	ClientServicePort
Data type:	ASCII Numeric
Justification:	Right, zero or space filled to the left
Length:	Variable with a maximum length of 5
Begin position:	Variable
End position:	Variable
Description:	The client's service port number that is being used in the TCP session with the instructor's server.

Insert a vertical '|' character in next position as a field separator

Field Name:	ClientSocketNumber
Data type:	ASCII Numeric
Justification:	Right, zero or space filled to the left
Length:	Variable with a maximum length of 5
Begin position:	Variable
End position:	Variable
Description:	The client's TCP socket number that is being used in the TCP session with the instructor's server.

Insert a vertical '|' character in next position as a field separator

Field Name:	ForeignHostIPAddress
Data type:	ASCII Alphanumeric
Justification:	Left, space filled to the right
Length:	Variable with a maximum length of 15
Begin position:	Variable
End position:	Variable
Description:	The IPv4 address of the foreign host (server) in dotted decimal notation

Insert a vertical '|' character in next position as a field separator

Field Name:	ForeignHostServicePort
Data type:	ASCII Numeric
Justification:	Right, zero or space filled to the left
Length:	Variable with a maximum length of 5
Begin position:	Variable
End position:	Variable
Description:	The foreign host's (server's) service port number that is being used in the TCP session with the student's client (should always be 2605).

Insert a vertical '|' character in next position as a field separator

Field Name:	StudentData
Data type	ASCII Alphanumeric
Justification:	Any

Length:	Variable with a maximum length of 20
Begin position:	Variable
End position:	Variable
Description:	Discretionary data defined by the student, may be all spaces. Will not be returned in the response.

Insert a vertical '|' character in next position as a field separator

Field Name:	ScenarioNo
Data type:	ASCII Numeric
Justification:	Any
Length:	1
Begin position:	Variable
End position:	Variable
Description:	A one digit number to indicate the scenario of this test.

Insert a vertical '|' character in next position as a field separator

Maximum length of request message146_{10}
Maximum binary value in TCP header field = 144_{10}, or HEX 0090

Response Message Format:

Field Name:	TCPHeader
Data type	Binary (Network, Big Endian, Byte Order)
Justification:	Right
Length:	2
Begin position:	0
End position	1
Description:	Binary value that represents the length of the response message, excluding these two bytes.

Field Name:	MessageType
Data type:	ASCII Alphanumeric fixed "**RSP**"
Justification:	Any
Length:	3
Begin position:	2
End position:	4
Description:	This will be a fixed three-character sequence of "**RSP**", all in upper case.

Insert a vertical '|' character at position 5

Field Name:	msTimeStamp
Data type:	ASCII Numeric
Justification:	Right, space or zero filled to the left
Length:	Variable with a maximum length of 10
Begin position:	6

End position:	Variable
Description:	A millisecond time stamp value from a running elapsed system timer on the server

Insert a vertical '|' character in next position as a field separator

Field Name:	RequestID
Data type:	ASCII Alphanumeric
Justification:	Any
Length	Variable with a maximum length of 20
Begin position:	Variable
End position:	Variable
Description:	Data from the *RequestID* field of the client's request message

Insert a vertical '|' character in next position as a field separator

Field Name:	StudentName
Data type:	ASCII Text
Justification:	Left, space filled to the right
Length:	Variable with a maximum length of 20
Begin position:	Variable
End position:	Variable
Description:	Data from the *StudentName* field of the client's request message

Insert a vertical '|' character in next position as a field separator

Field Name:	Student ID
Data type:	ASCII Aphanumeric
Justification:	Any
Length:	Variable with a maximum length of 7
Begin position:	Variable
End position:	Variable
Description:	Data from the *StudentID* field of the client's request message

Insert a vertical '|' character in next position as a field separator

Field Name:	ResponseDelay
Data type:	ASCII Numeric
Justification:	Right, space or zero filled to the left
Length:	Variable with a maximum length of 5
Begin position:	Variable
End position:	Variable

Description:	Data from the ResponseDelay field of the client's request message

Insert a vertical '|' character in next position as a field separator

Field Name:	ForeignHostIPAddress
Data type:	ASCII Alphanumeric
Justification:	Left, space filled to the right
Length:	Variable with a maximum length of 15
Begin position:	Variable
End position:	Variable
Description:	The IPv4 address of the server's foreign host in dotted decimal notation

Insert a vertical '|' character in next position as a field separator

Field Name:	ForeignHostServicePort
Data type:	ASCII Numeric
Justification:	Right, zero or space filled to the left
Length:	Variable with a maximum length of 5
Begin position:	Variable
End position:	Variable
Description:	The server's foreign host's service port number, should always be 2605.

Insert a vertical '|' character in next position as a field separator

Field Name:	ServerSocketNumber
Data type:	ASCII Numeric
Justification:	Right, zero or space filled to the left
Length:	Variable with a maximum length of 5
Begin position:	Variable
End position:	Variable
Description:	The server's TCP socket number.

Insert a vertical '|' character in next position as a field separator

Field Name:	ServerIPAddress
Data type:	ASCII Numeric
Justification:	Right, zero or space filled to the left
Length:	Variable with a maximum length of 15
Begin position:	Variable

End position:	Variable
Description:	The IPv4 address of the server in dotted decimal notation.

Insert a vertical '|' character in next position as a field separator

Field Name:	ServerServicePort
Data type:	ASCII Numeric
Justification:	Right, zero or space filled to the left
Length:	Variable with a maximum length of 5
Begin position:	Variable
End position:	Variable
Description:	The server's service port number, should be 2605.

Insert a vertical '|' character in next position as a field separator

Field Name:	ResponseID
Data type:	ASCII Alphanumeric
Justification:	Any
Length:	Variable with a maximum length of 20
Begin position:	Variable
End position:	Variable
Description:	A unique value generated by the server to exclusively identify responses.

Insert a vertical '|' character in next position as a field separator

Field Name:	ResponseType
Data type:	ASCII Numeric
Justification:	Any
Length:	1
Begin position:	Variable
End position	Variable
Description:	This response field is to be populated by the client to indicate the following: '1'➔Normal response received from server '2'➔Stand-in response generated by client '3'➔Latent response received from server '4'➔Spurious or unsolicited response received from server

Insert a vertical '|' character in next position as a field separator
Maximum response length = 146_{10}
Maximum binary value in TCP header field = 144_{10}, or HEX 0090

Log File Organization:

To receive credit for this lab assignment, students are to submit their log files to the instructor. Please submit the logs as soon as you are comfortable with the results. Do not wait to accumulate log files from all three scenarios before submitting for the first scenario. There is to be only one log file for each scenario organized as follows:

1. Each line item in the log file represents either a request message or response message, except for the last line item which is a log trailer record.

2. Each line item is to be separated with a <CR><LF> character sequence.

3. Request line items are to be ordered in the sequence the request messages were issued. Hence, the request *msTimeStamp* field should continuously increment.

4. Response line items are to be ordered in the sequence they were received. Hence, the response *msTimeStamp* field should continuously increment.

5. The last line item is to be a log trailer record formatted in accordance with the *Trailer Record Format and Content* section of this assignment.

Trailer Record Format and Content:

The last line item of each log file is to be formatted with the following items of data:

Field Name:	Date
Data type:	ASCII, in the form of MMDDYYYY
Justification:	Any
Length:	8
Begin position:	0

End position: 7

Description: Date of completion of lab scenario

Insert a vertical '|' character at position 8

Field Name: Time

Data type: ASCII, in the form of HHMMSS [24 hr. notation]

Justification: Any

Length: 6

Begin position: 9

End position: 14

Description: Time of completion of lab scenario

Insert a vertical '|' character at position 15

Field Name: RcvShutdownStatus

Data type: ASCII Numeric

Justification: Right, space or zero filled to the left

Length: 5

Begin position: 16

End position: 20

Description: Return status from issuing a shutdown of the receive half session.

Insert a vertical '|' character at position 21

Field Name:	XmtShutdownStatus
Data type:	ASCII Numeric
Justification:	Right, space or zero filled to the left
Length:	5
Begin position:	22
End position:	26
Description:	Return status from issuing a shutdown of the transmit half session.

Insert a vertical '|' character at position 27

Field Name:	CloseStatus
Data type:	ASCII Numeric
Justification:	Right, space or zero filled to the left
Length:	5
Begin position:	28
End position:	32
Description:	Return status from issuing a socket close command