Udp tester
Report

Steven Reid
A00929669
sreid97@my.bcit.ca
Submitted on: February 22, 2022

Purpose

The udp_tester program implements a simple client/server POSIX compliant program for testing and display of UDP packets.

Requirements

Task	Status
Report (Testing, User Guide, Design)	Fully Implemented
Video	Fully Implemented
Start immediately	Fully Implemented
Start at a specific time	Fully Implemented
# of packets to send	Fully Implemented
Port	Fully Implemented
Server Statistics	Fully Implemented
- received	Fully Implemented
- lost	Fully Implemented
- Out of order	Fully Implemented
- Min/Max lost	Fully Implemented
- Min/Maxc out of order	Fully Implemented

Platforms

udp_tester has been tested on:

- Ubuntu 21.10

Language

- ISO C11
- Compiles with gcc and clang

Design

Structures

Legend

User-entered program arguments
Other source

client → FSM

Field	Purpose
server	Contains IP address for server
port	Contains port used by client
start	Start time for packet transmission
packets	Amount of packets to send to server
packetSize	Size of packets end to server (in Bytes)
delay	Delay between packets sent to server
tcpSocketFD	File Descriptor for tcp socket
udpSocketFD	File Descriptor for udp socket
clientID	Unique ID assigned by server
serverAddress	Address information for server

logParser → Linked List Node → Assigned as typedef Link

Field	Purpose
clientID	4 character unique ID for each client
expectedNumberOfPackets	Expected Packets
receivedNumberOfPackets	Received Packets
packetSize	Size of each Packet
packetIDs	Array of u_int16_t values (dynamically allocated)
next	Pointer to next element in linked list

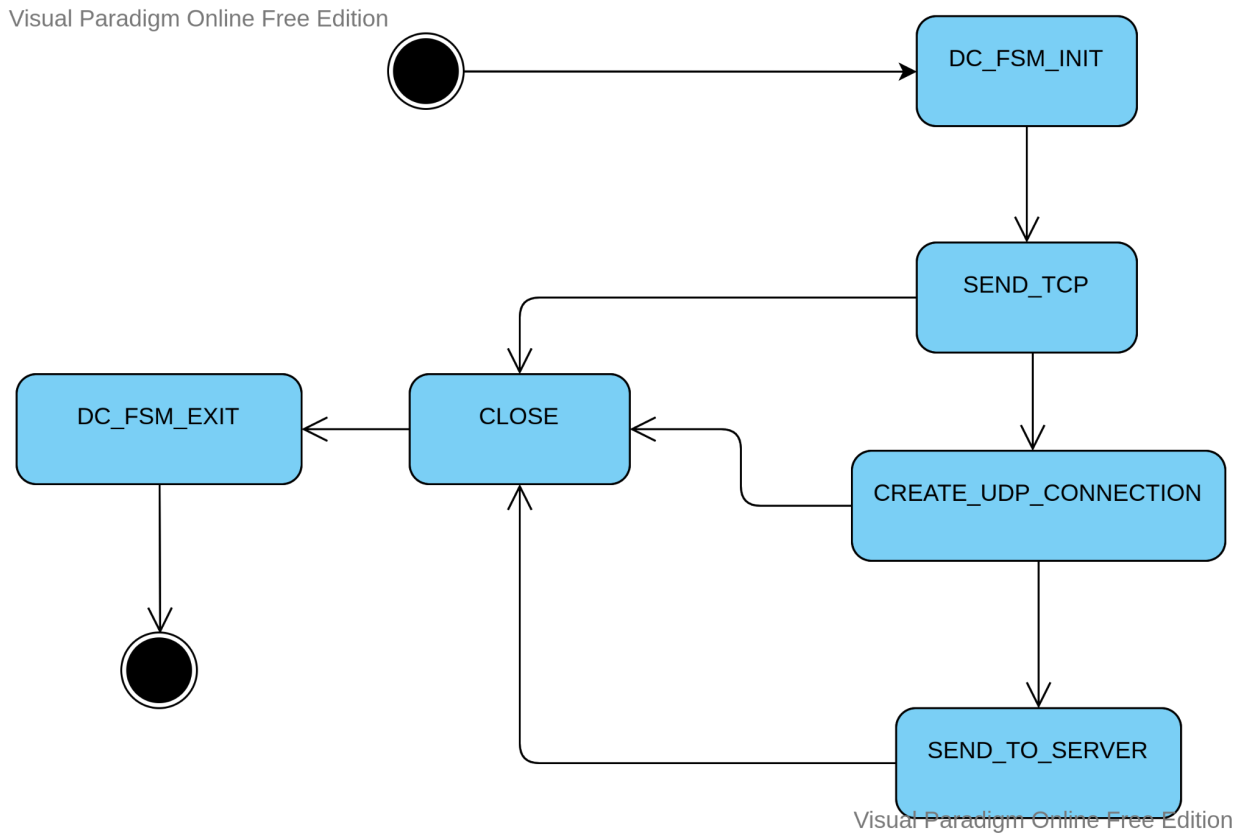
Finite State Machine

State Transition Table - client

From State	To State	Action
DC_FSM_INIT	SEND_TCP	sendTCPInformation
SEND_TCP	CREATE_UDP_CONNECTION	createSocket
SEND_TCP	CLOSE	closeConnection
CREATE_UDP_CONNECTION	SEND_TO_SERVER	sendToServer
CREATE_UDP_CONNECTION	CLOSE	closeConnection
SEND_TO_SERVER	CLOSE	closeConnection
CLOSE	DC_FSM_EXIT	—

State Transition Diagram

client



Functions

client

sendTCPInformation

Purpose

Establish a TCP connection with the server and receive a client ID.

Parameters

The state object.

Return

Type	Next State
Success	`CREATE_UDP_CONNECTION`
Failure	`CLOSE`

Pseudocode

- Establish TCP connection
    - Get Address information
    - Create TCP socket
    - Connect to TCP server
    - Write UDP packet information to server
    - Read Unique Client ID from server and store
- Go to next state to create UDP connection

createSocket

Purpose

Create a UDP socket that will be used in sending packets.

Parameters

The state object.

Return

Type	Next State
Success	`SEND_TO_SERVER`
Failure	`CLOSE`

Pseudocode

- Create UDP socket
- Initialize serverAddress struct with family, port and addr variables

getCurrentTime

Purpose

Get local time.

Parameters

Environment and Error structs only.

Return

Type	Value
char *	`Current time as readable string.`

Pseudocode

- Call time function to get current time
- Use ctime to parse into a readable string
- Iterate through string to separate “HH:MM” from the rest
- Return “HH:MM”

sendToServer

Purpose

Sending of UDP packets to the server as specified by program arguments.

Parameters

The state object.

Return

Type	Next State
Success	`CLOSE`
Failure	`CLOSE`

Pseudocode

- Call getCurrentTime
- While start does not equal current time or start does not equal “START”
    - Print a waiting message
    - Call getCurrentTime
    - Sleep for 30 seconds
- Pad the body of each packet with packetsize - size of ID string
- For loop (from 0 → size of packets)
    - Send packet
    - Delay using nanosleep for delay time

closeConnection

Purpose

Terminate TCP and UDP connections from the client.

Parameters

The state object.

Return

Type	Next State
Success	`CLOSE`
Failure	`CLOSE`

Pseudocode

- If tcp socket exists → write termination message and close
- If udp socket exists → close

server

max

Purpose

Find maximum value between two integers.

Parameters

Type	Value
int	x
int	y

Return

Type	Value
int	Maximum of two integers.

Pseudocode

- If x > y → return x

- Otherwise, return y

createServer

Purpose

Establish TCP and UDP connections that are listened to in a loop. Writes received information to log files.

Parameters

Type	Value
const struct dc_posix env	env
struct dc_error	err
u_int16_t	tcpPort passed from program argument

Return

Type	Value
void	N/A

Pseudocode

- Create listening TCP socket
- Bind TCP socket
- Create listening UDP socket
- Bind UDP socket
- Listen for connections on both sockets with select
- for loop(;;)
    - If connection on TCP
        - Accept
        - Fork
        - Write TCP information to tcpLog.txt in logs folder
        - Wait for TCP child process to complete
    - If connection on UDP
        - Receive packets from client
        - Write packets to udpLog.txt in logs folder
- For loop will continue until server is terminated in order to listen uninterrupted

logParser

createNode

Purpose

Create a single node for linked lists.

Parameters

Type	Value
void	N/A

Return

Type	Value
Link	Typedef struct Node*

Pseudocode

- Allocate memory for a new Link
- Set link->next to NULL
- Return link

createNodeWithNextNode

Purpose

Create a single node for linked lists and assign the next pointer to another node.

Parameters

Type	Value
Link	Pointer to a linked list node

Return

Type	Value
Link	Typedef struct Node*

Pseudocode

- Call createNode()
- Set link->next to link parameters
- Return link

getTail

Purpose

Get a pointer to the tail of a linked list.

Parameters

Type	Value
Link	Head of a linked list

Return

Type	Value
Link	Typedef struct Node*

Pseudocode

- If no head
    - return NULL
- If no head→next
    - Return head parameter
- Return getTail(head->next) → Recursive

addLast

Purpose

Add a node to the end of a linked list.

Parameters

Type	Value
Link *	Pointer to head of a linked list

Return

Type	Value
Link	Typedef struct Node*

Pseudocode

- Call createNode() to create newTail Link
- If no head
    - head = newTail

- Call getTail and assign return to currentTail Link
- Set currentTail→next to newTail
- Return newTail

deleteList

Purpose

Free memory from a linked list.

Parameters

Type	Value
Link	Head of a linked list

Return

Type	Value
Link	Typedef struct Node*

Pseudocode

- While (current node is not NULL)
    - Free current node
    - Set current to next node
- Set head to NULL

printMissingPackets

Purpose

Print missing UDP packets to STDOUT.

Parameters

Type	Value
const struct dc_posix env	env
struct dc_error	err
u_int16_t	packetIDs
size_t	expectedPackets
size_t	receivedPackets

Return

Type	Value
size_t	lostPacketsCounter

Pseudocode

- Create boolean array of size expectedPackets + 1
- Map all received packetIDs from input parameter to boolean array
- If any boolean values are still false after mapping, print to STDOUT, as they have been missed

calculateMissingPacketsInSequence

Purpose

Calculates the min and max packets missing in sequence.

Parameters

Type	Value
const struct dc_posix env	env
struct dc_error	err
u_int16_t	packetIDs
size_t	expectedPackets
size_t	receivedPackets
const char*	function to return

Return

Type	Value
size_t	minLost or maxLost value

Pseudocode

- Create boolean array of size expectedPackets + 1
- Map all received packetIDs from input parameter to boolean array
- If any boolean values are still false after mapping, store to lostPackets array and increment lostPacketsCounter
- For all values of lostPacketsCounter

        - If packets are in sequence, increment max sequence counter
        - Else, continue
- Return min or max lost counter

printOutOfOrderPackets

Purpose

Print out of order UDP packets to STDOUT.

Parameters

Type	Value
const struct dc_posix env	env
struct dc_error	err
u_int16_t	packetIDs
size_t	receivedPackets

Return

Type	Value
void	N/A

Pseudocode

- If packets are not in order
    - Increment out of order counter
- Print number of packets out of order to STDOUT
- For all received packets
    - If packets are out of order
    - Print packet ID to STDOUT

calculateOutOfOrderInSequence

Purpose

Calculate the min and max packets out of order in sequence

Parameters

Type	Value

const struct dc_posix env	env
struct dc_error	err
u_int16_t	packetIDs
size_t	expectedPackets
const char*	function to return

Return

Type	Value
size_t	minOrder or maxOrder value

Pseudocode

- If packets are not in order
    - Increment out of order counter
- For all received packets
    - If packets are out of order
    - Store to outOfOrderPackets array
- For all outOfOrderPackets in array
    - If in sequence, increment maxOrder counter

parseLogStatistics

Purpose

Parse TCP and UDP logs for analysis and display.

Parameters

Type	Value
const struct dc_posix env	env
struct dc_error	err

Return

Type	Value
void	N/A

Pseudocode

- Open tcpLog.txt

- While dc_getline > 0
    - Get line of tcpLog
    - Create head of client linked list
    - Create linked list node for client
    - Extract details from log line and store to client node
- While clientHead of linked list exists (is not NULL)
    - Open udpLog.txt
    - While client ID matches the ID of current linked list Node
        - Store Packet IDs to array in client Node
        - Increment packet counter
    - Write to screen:
        - Expected packets
        - Received packets
        - Lost packets
    - Call printMissingPackets to print missing packets and update the total of lost packets
    - Call calculateMissingPacketsInSequence to calculate min and max lost packets in sequence
    - Call calculateOutOfOrderInSequence to calculate min and max out of order packets in sequence
    - Call printOutOfOrderPackets to print out of order packets
- Print to screen:
    - Minimum Lost Packets
    - Maximum Lost Packets
    - Minimum Out Of Order Packets
    - Maximum Out Of Order Packets
    - Average Number Of Lost Packets
- Free linked list

Testing

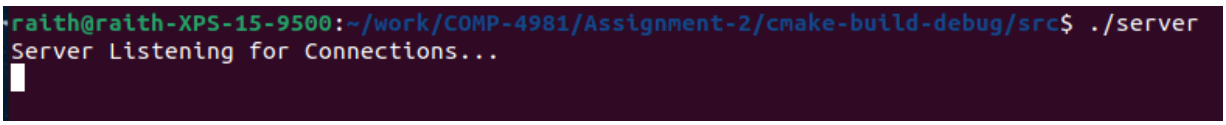
Test Results

Command	Description	Status	Example
./server	Start server with default parameters	Passed	Test 1
./client	Start client with default parameters and connect to server (starts immediately)	Passed	Test 2
N/A	Verify creation of log files in logs directory	Passed	Test 3

./client –start 21:02	Start client at a specific time	Passed	Test 4
N/A	Verify creation of new client at specified time in log files	Passed	Test 5
./client –packets 50	Start client with a specified number of packets	Passed	Test 6
N/A	Verify creation of 50 packets in log file	Passed	Test 7
./server –port 4999	Start server on specified port	Passed	Test 8
./client –port 4999	Start client on specified port	Passed	Test 9
N/A	Verify creation of 100 packets in log file	Passed	Test 10
./logParser	Run logParser to generate statistics	Passed	Test 11
./logParser	Change udpLog.txt - Removed one entry - Rearranged one entry Run logParser again.	Passed	Test 12
./logParser	Testing of example with packets ordered as such: 1, 2, 4, 5, 8	Passed	Test 13
./logParser	Testing of example with packets ordered as such: 1, 3, 2, 4, 7, 6, 5	Passed	Test 14
./client	Testing running multiple clients in separate terminals (3 clients)	Passed	Test 15
N/A	Verify creation of 3 clients worth of packets in log files	Passed	Test 16

Examples

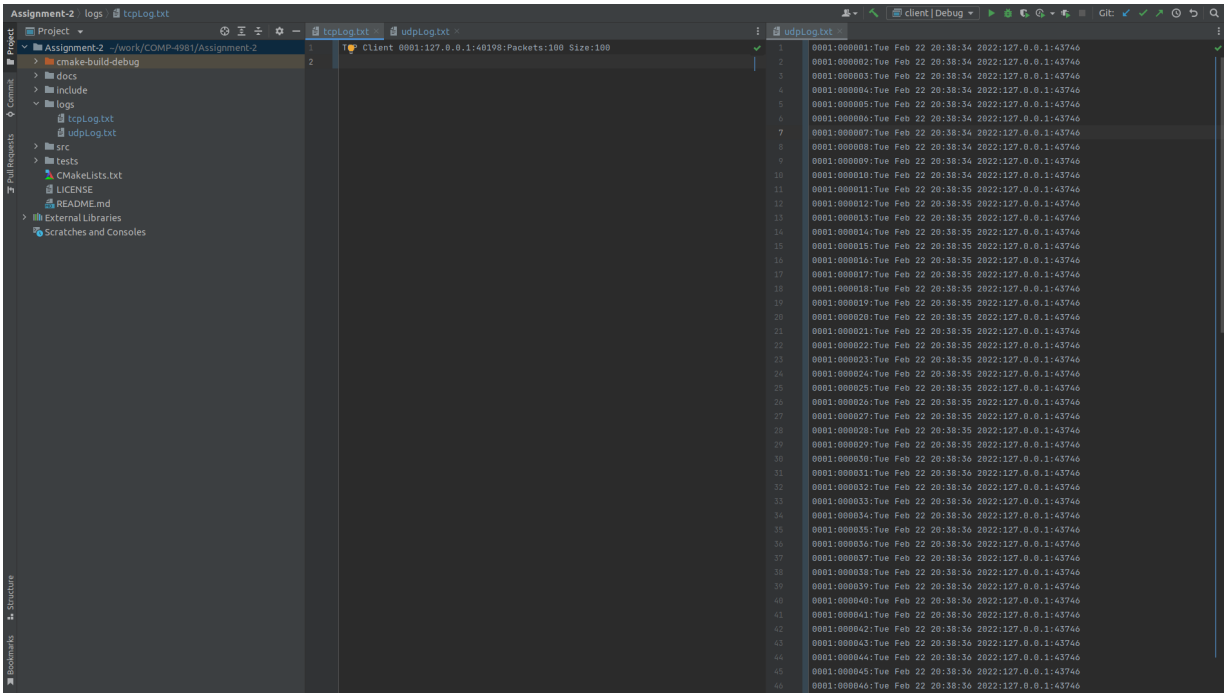
Test 1



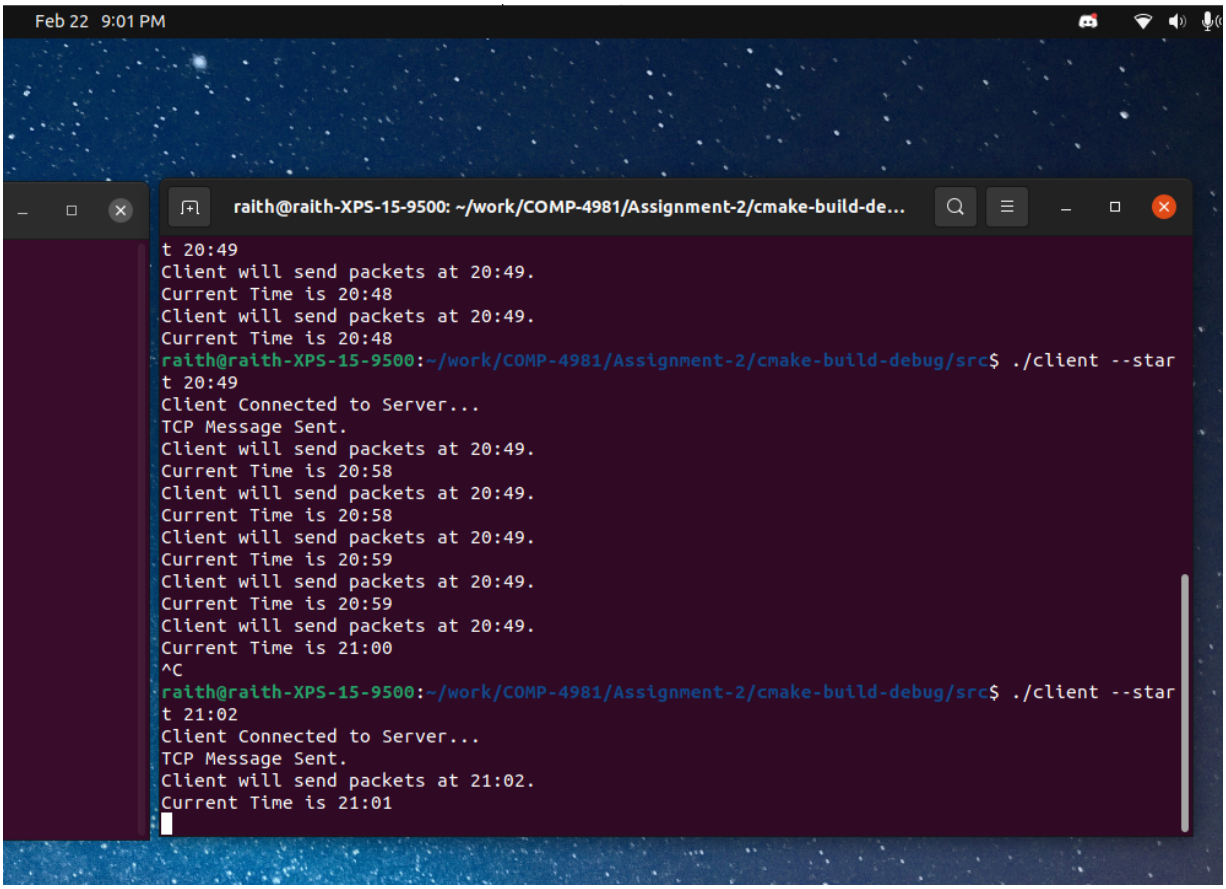
Test 2

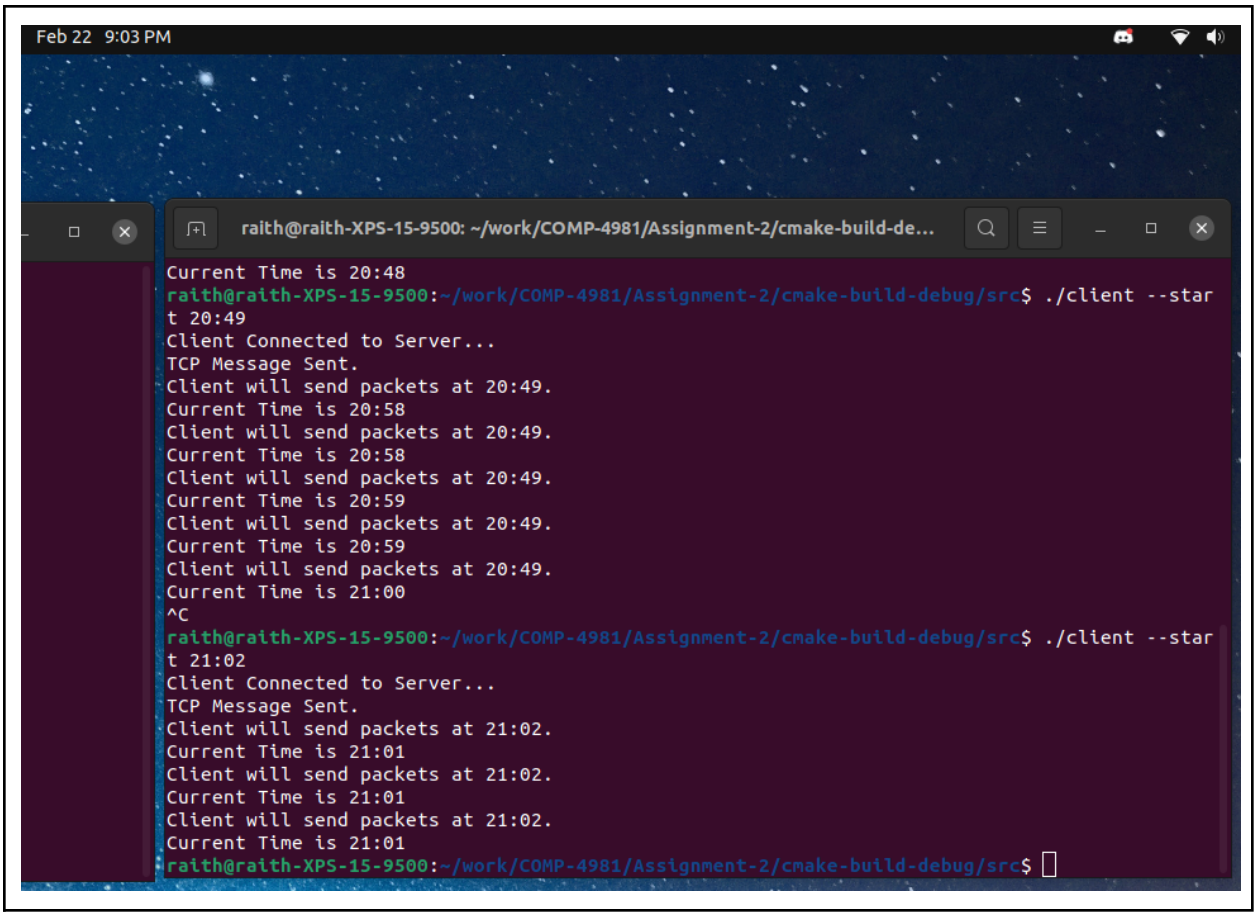
```
raith@raith-XPS-15-9500:~/work/COMP-4981/Assignment-2/cmake-build-debug/src$ ./client
Client Connected to Server...
TCP Message Sent.
raith@raith-XPS-15-9500:~/work/COMP-4981/Assignment-2/cmake-build-debug/src$
```

Test 3

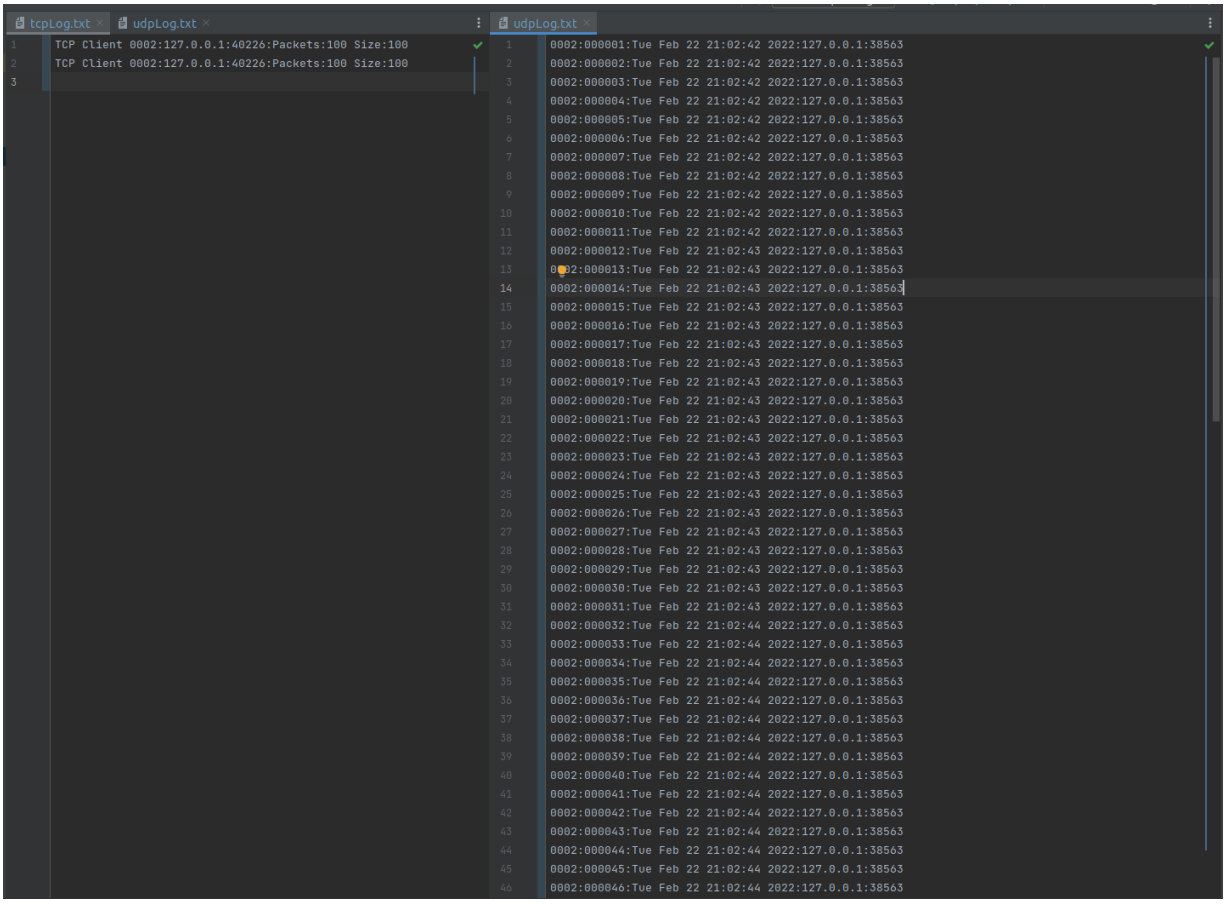


Test 4


Feb 22 9:01 PM
raith@raith-XPS-15-9500: ~/work/COMP-4981/Assignment-2/cmake-build-de...
t 20:49
Client will send packets at 20:49.
Current Time is 20:48
Client will send packets at 20:49.
Current Time is 20:48
raith@raith-XPS-15-9500:~/work/COMP-4981/Assignment-2/cmake-build-debug/src$ ./client --star
t 20:49
Client Connected to Server...
TCP Message Sent.
Client will send packets at 20:49.
Current Time is 20:58
Client will send packets at 20:49.
Current Time is 20:58
Client will send packets at 20:49.
Current Time is 20:59
Client will send packets at 20:49.
Current Time is 20:59
Client will send packets at 20:49.
Current Time is 21:00
^C
raith@raith-XPS-15-9500:~/work/COMP-4981/Assignment-2/cmake-build-debug/src$ ./client --star
t 21:02
Client Connected to Server...
TCP Message Sent.
Client will send packets at 21:02.
Current Time is 21:01


Feb 22 9:03 PM
raith@raith-XPS-15-9500: ~/work/COMP-4981/Assignment-2/cmake-build-de...
Current Time is 20:48
raith@raith-XPS-15-9500:~/work/COMP-4981/Assignment-2/cmake-build-debug/src$ ./client --star
t 20:49
Client Connected to Server...
TCP Message Sent.
Client will send packets at 20:49.
Current Time is 20:58
Client will send packets at 20:49.
Current Time is 20:58
Client will send packets at 20:49.
Current Time is 20:59
Client will send packets at 20:49.
Current Time is 20:59
Client will send packets at 20:49.
Current Time is 21:00
^C
raith@raith-XPS-15-9500:~/work/COMP-4981/Assignment-2/cmake-build-debug/src$ ./client --star
t 21:02
Client Connected to Server...
TCP Message Sent.
Client will send packets at 21:02.
Current Time is 21:01
Client will send packets at 21:02.
Current Time is 21:01
Client will send packets at 21:02.
Current Time is 21:01
raith@raith-XPS-15-9500:~/work/COMP-4981/Assignment-2/cmake-build-debug/src$

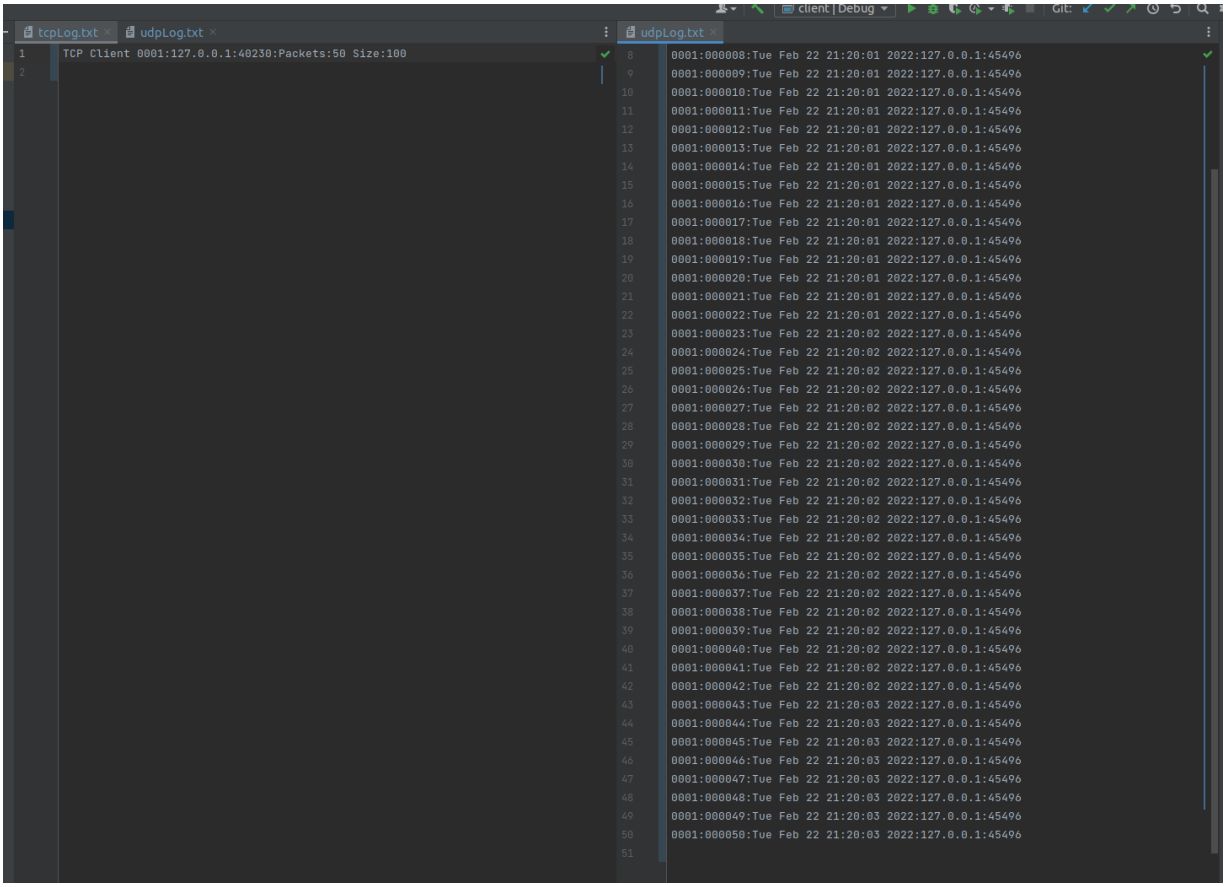
Test 5



Test 6

```
raith@raith-XPS-15-9500:~/work/COMP-4981/Assignment-2/cmake-build-debug/src$ ./client --packets 50
Client Connected to Server...
TCP Message Sent.
raith@raith-XPS-15-9500:~/work/COMP-4981/Assignment-2/cmake-build-debug/src$
```

Test 7



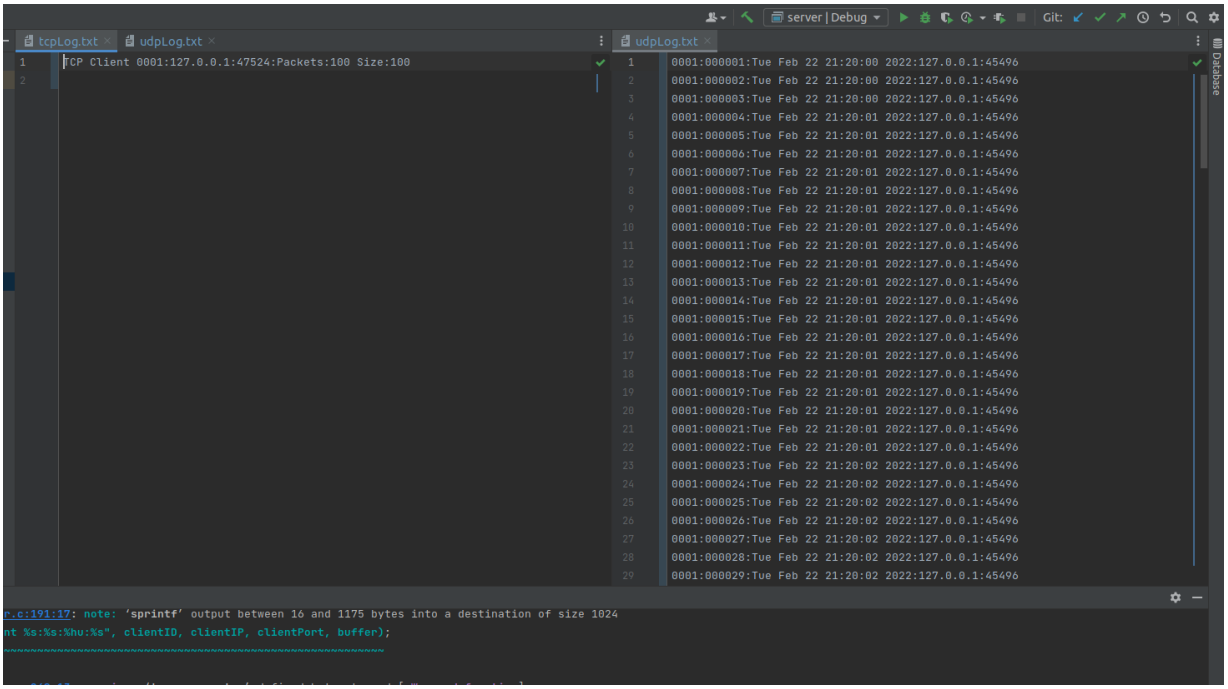
Test 8

```
raith@raith-XPS-15-9500:~/work/COMP-4981/Assignment-2/cmake-build-debug/src$ ./server --port 4999
Server Listening for Connections...
```

Test 9

```
raith@raith-XPS-15-9500:~/work/COMP-4981/Assignment-2/cmake-build-debug/src$ ./client --port 4999
Client Connected to Server...
TCP Message Sent.
raith@raith-XPS-15-9500:~/work/COMP-4981/Assignment-2/cmake-build-debug/src$
```

Test 10



Test 11

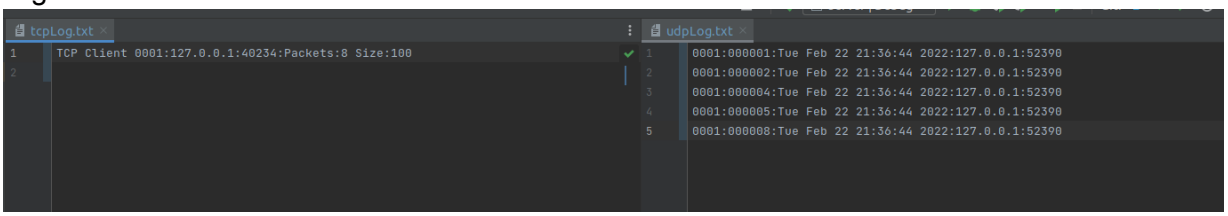
```
raith@raith-XPS-15-9500:~/work/COMP-4981/Assignment-2/cmake-build-debug/src$ ./logParser
Client 0001:
Packets Expected = 100
Packets Received = 100
Packets Lost = 0
---------------------------------------
Packet Numbers Lost:
---------------------------------------
Number of Packets Out Of Order: 0
---------------------------------------
Packet Numbers Out Of Order:
---------------------------------------
Minimum Lost Packets: 0
Maximum Lost Packets: 0
---------------------------------------
Minimum Out Of Order Packets: 0
Maximum Out Of Order Packets: 0
---------------------------------------
Average Number Of Lost Packets: 0.000000
raith@raith-XPS-15-9500:~/work/COMP-4981/Assignment-2/cmake-build-debug/src$
```

Test 12

```
raith@raith-XPS-15-9500:~/work/COMP-4981/Assignment-2/cmake-build-debug/src$ ./logParser
Client 0001:
Packets Expected = 100
Packets Received = 99
Packets Lost = 1
---------------------------------------
Packet Numbers Lost:
000011
---------------------------------------
Number of Packets Out Of Order: 1
---------------------------------------
Packet Numbers Out Of Order:
000024
---------------------------------------
Minimum Lost Packets: 1
Maximum Lost Packets: 1
---------------------------------------
Minimum Out Of Order Packets: 1
Maximum Out Of Order Packets: 1
---------------------------------------
Average Number Of Lost Packets: 1.000000
raith@raith-XPS-15-9500:~/work/COMP-4981/Assignment-2/cmake-build-debug/src$
```

Test 13

Log file looks like this:

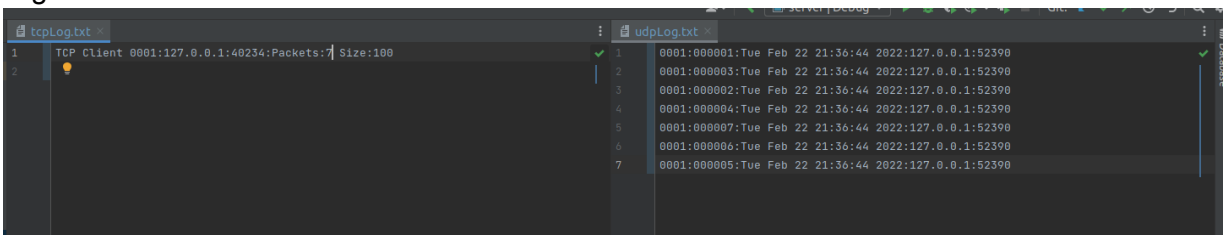


logParser run:

```
raith@raith-XPS-15-9500:~/work/COMP-4981/Assignment-2/cmake-build-debug/src$ ./logParser
Client 0001:
Packets Expected = 8
Packets Received = 5
Packets Lost = 3
---------------------------------------
Packet Numbers Lost:
000003
000006
000007
---------------------------------------
Number of Packets Out Of Order: 0
---------------------------------------
Packet Numbers Out Of Order:
---------------------------------------
Minimum Lost Packets: 1
Maximum Lost Packets: 2
---------------------------------------
Minimum Out Of Order Packets: 0
Maximum Out Of Order Packets: 0
---------------------------------------
Average Number Of Lost Packets: 3.000000
raith@raith-XPS-15-9500:~/work/COMP-4981/Assignment-2/cmake-build-debug/src$
```

Test 14

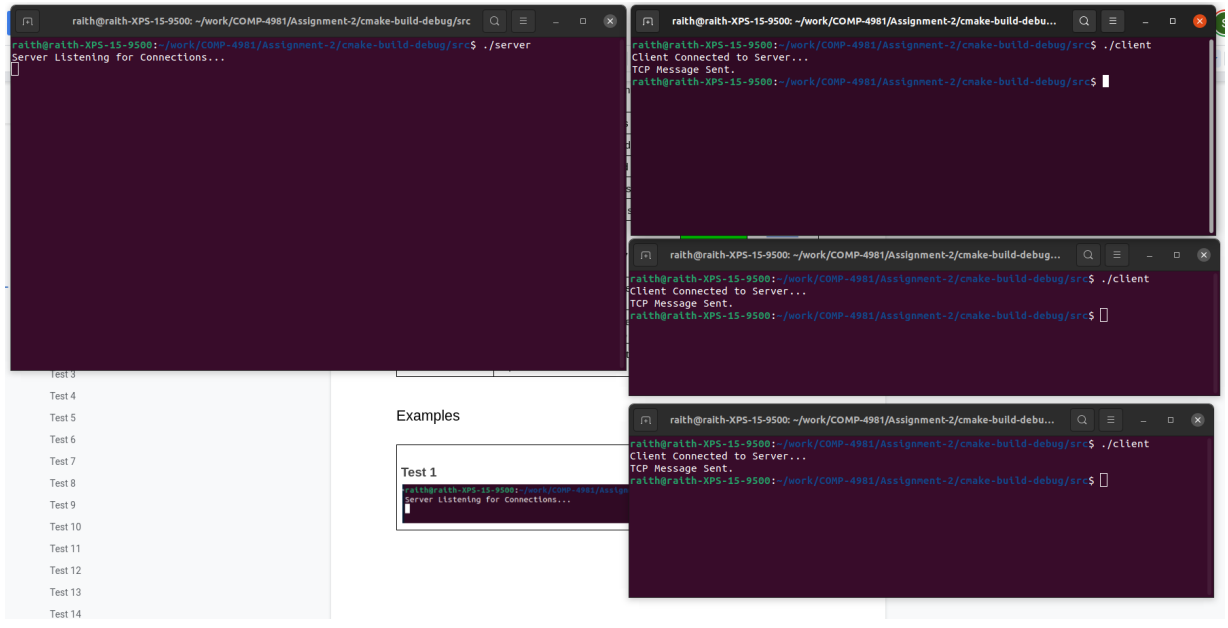
Log file looks like this:



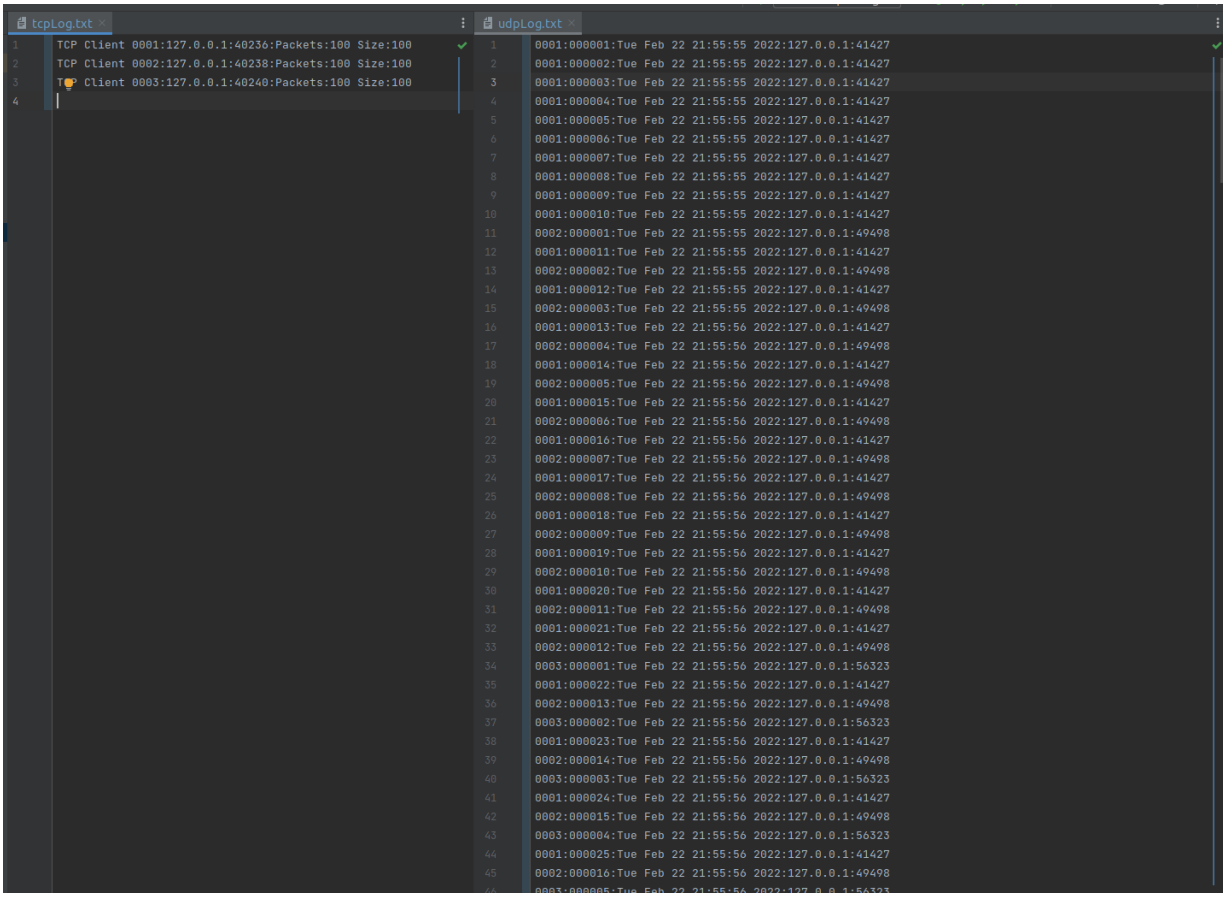
logParser run:

```
raith@raith-XPS-15-9500:~/work/COMP-4981/Assignment-2/cmake-build-debug/src$ ./logParser
Client 0001:
Packets Expected = 7
Packets Received = 7
Packets Lost = 0
---------------------------------------
Packet Numbers Lost:
---------------------------------------
Number of Packets Out Of Order: 3
---------------------------------------
Packet Numbers Out Of Order:
000002
000006
000005
---------------------------------------
Minimum Lost Packets: 0
Maximum Lost Packets: 0
---------------------------------------
Minimum Out Of Order Packets: 1
Maximum Out Of Order Packets: 2
---------------------------------------
Average Number Of Lost Packets: 0.000000
raith@raith-XPS-15-9500:~/work/COMP-4981/Assignment-2/cmake-build-debug/src$
```

Test 15



Test 16

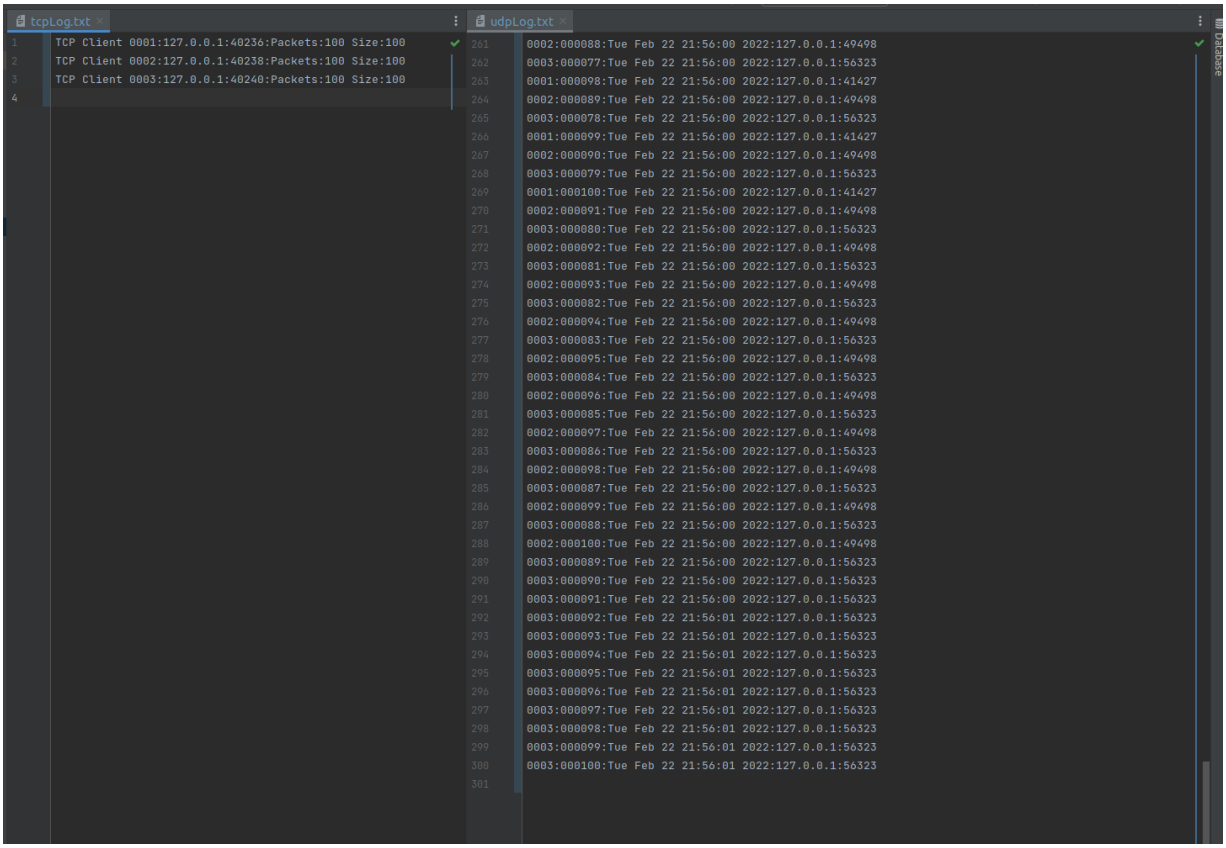


tcpLog.txt

```
TCP Client 0001:127.0.0.1:40236:Packets:100 Size:100
TCP Client 0002:127.0.0.1:40238:Packets:100 Size:100
TCP Client 0003:127.0.0.1:40240:Packets:100 Size:100
```

udpLog.txt

```
0001:000001:Tue Feb 22 21:55:55 2022:127.0.0.1:41427
0001:000002:Tue Feb 22 21:55:55 2022:127.0.0.1:41427
0001:000003:Tue Feb 22 21:55:55 2022:127.0.0.1:41427
0001:000004:Tue Feb 22 21:55:55 2022:127.0.0.1:41427
0001:000005:Tue Feb 22 21:55:55 2022:127.0.0.1:41427
0001:000006:Tue Feb 22 21:55:55 2022:127.0.0.1:41427
0001:000007:Tue Feb 22 21:55:55 2022:127.0.0.1:41427
0001:000008:Tue Feb 22 21:55:55 2022:127.0.0.1:41427
0001:000009:Tue Feb 22 21:55:55 2022:127.0.0.1:41427
0001:000010:Tue Feb 22 21:55:55 2022:127.0.0.1:41427
0002:000001:Tue Feb 22 21:55:55 2022:127.0.0.1:49498
0001:000011:Tue Feb 22 21:55:55 2022:127.0.0.1:41427
0002:000002:Tue Feb 22 21:55:55 2022:127.0.0.1:49498
0001:000012:Tue Feb 22 21:55:55 2022:127.0.0.1:41427
0002:000003:Tue Feb 22 21:55:55 2022:127.0.0.1:49498
0001:000013:Tue Feb 22 21:55:56 2022:127.0.0.1:41427
0002:000004:Tue Feb 22 21:55:56 2022:127.0.0.1:49498
0001:000014:Tue Feb 22 21:55:56 2022:127.0.0.1:41427
0002:000005:Tue Feb 22 21:55:56 2022:127.0.0.1:49498
0001:000015:Tue Feb 22 21:55:56 2022:127.0.0.1:41427
0002:000006:Tue Feb 22 21:55:56 2022:127.0.0.1:49498
0001:000016:Tue Feb 22 21:55:56 2022:127.0.0.1:41427
0002:000007:Tue Feb 22 21:55:56 2022:127.0.0.1:49498
0001:000017:Tue Feb 22 21:55:56 2022:127.0.0.1:41427
0002:000008:Tue Feb 22 21:55:56 2022:127.0.0.1:49498
0001:000018:Tue Feb 22 21:55:56 2022:127.0.0.1:41427
0002:000009:Tue Feb 22 21:55:56 2022:127.0.0.1:49498
0001:000019:Tue Feb 22 21:55:56 2022:127.0.0.1:41427
0002:000010:Tue Feb 22 21:55:56 2022:127.0.0.1:49498
0001:000020:Tue Feb 22 21:55:56 2022:127.0.0.1:41427
0002:000011:Tue Feb 22 21:55:56 2022:127.0.0.1:49498
0001:000021:Tue Feb 22 21:55:56 2022:127.0.0.1:41427
0002:000012:Tue Feb 22 21:55:56 2022:127.0.0.1:49498
0003:000001:Tue Feb 22 21:55:56 2022:127.0.0.1:56323
0001:000022:Tue Feb 22 21:55:56 2022:127.0.0.1:41427
0002:000013:Tue Feb 22 21:55:56 2022:127.0.0.1:49498
0003:000002:Tue Feb 22 21:55:56 2022:127.0.0.1:56323
0001:000023:Tue Feb 22 21:55:56 2022:127.0.0.1:41427
0002:000014:Tue Feb 22 21:55:56 2022:127.0.0.1:49498
0003:000003:Tue Feb 22 21:55:56 2022:127.0.0.1:56323
0001:000024:Tue Feb 22 21:55:56 2022:127.0.0.1:41427
0002:000015:Tue Feb 22 21:55:56 2022:127.0.0.1:49498
0003:000004:Tue Feb 22 21:55:56 2022:127.0.0.1:56323
0001:000025:Tue Feb 22 21:55:56 2022:127.0.0.1:41427
0002:000016:Tue Feb 22 21:55:56 2022:127.0.0.1:49498
```



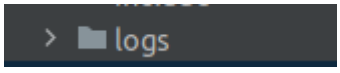
Test 17

User Guide

Installing

Obtaining

Download package from Learning Hub and install as Assignment 2 root folder.

Ensure that 

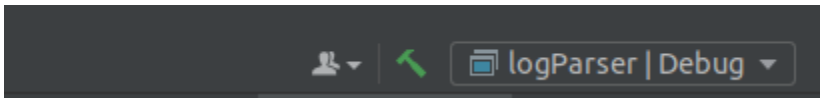
is present in the root folder.
If not, then

```
raith@raith-XPS-15-9500:~/work/COMP-4981/Assignment-2$ mkdir logs
```

In the project root folder.

Building

CLion → Build



OR

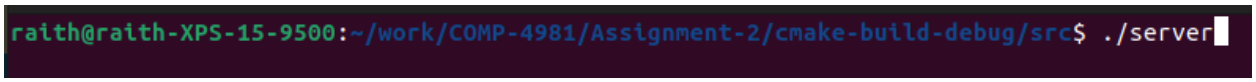
mkdir Assignment-2/cmake-build-debug
cmake -S udp_tester -B Assignment-2/cmake-build-debug
cmake --build Assignment-2/cmake-build-debug/

Installing

sudo cmake --install Assignment-2/cmake-build-debug

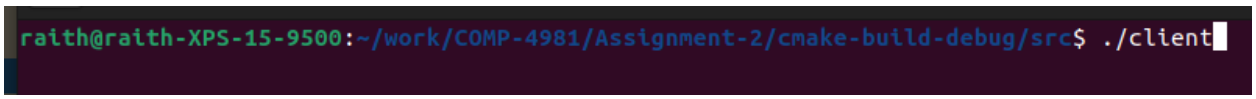
Running

Server



- Start the server first on the host computer or terminal.

Client



- Run client with any combination of program arguments
- View created log files in /logs folder for evidence of completion

Log Parser

- Run at any time to provide analysis of log files created by server

Program Arguments

Client

`./client --server`	Set ip address of server Default is 127.0.0.1

`./client --port`	Set port of tcp connection Default is 4981
`./client --start`	Set start time of server in format “HH:MM” in 24 hour format Default is “START” → starts immediately
`./client --packets`	Set amount of packets to send to server Default is 100
`./client --pSize`	Set packet size Default is 100 B
`./client --delay`	Set delay between packets Default is 50 ms

Server

`./client --port`	Set port of tcp connection Default is 4981

Log Parser

None	N/A

Features

- Server with ability to host on a chosen port
- Client with the ability to test UDP packet sending with a variety of combinations
- Log Parser that can be run separately to do an analysis of created log files

Limitations

- Log Parser must be run separately in order to keep server running
- Errors in log files can cause Log Parser to fail (corruption, damaged memory)
- Interface limited to console out at this time