SYSC 3310

Group Project

TFTP Protocol w/ ERROR Check

Team 2

Loo Ben

Nguyen Andrew

Morrissette Eric

Palko Ben

Date: March 10th 2018

Table of Content

|  |  |
| --- | --- |
| Topics | Pages |
| ERROR Codes Deliverables  Timing Diagram ERROR Codes  Iteration 2  ERROR Packets Checking 1  ERROR Packets Checking 2  ERROR Packets Checking 3  ERROR Packets Checking 6  Iteration 4  ERROR Packets Checking 4  ERROR Packets Checking 5  Iteration 5  File Transfer between Different Computers  Breakdown Responsibilities  Diagrams  UCMs for Read/Write w/ Error Simulator  UML Class Diagram  How to Run and Troubleshoot  How to Run and Test Instructions  Test Table  Source Code |  |

Iteration Deliveries

Timing Diagram ERROR Codes

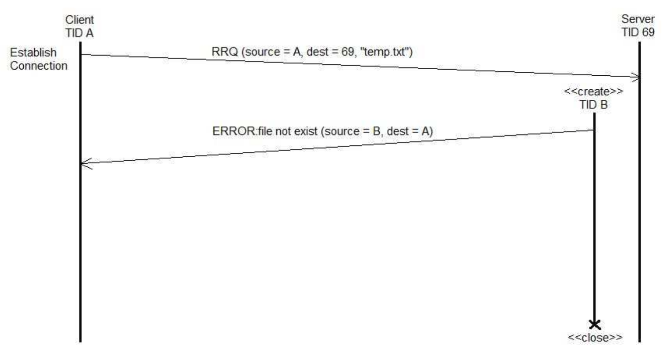
ERROR Codes

0 Not defined, see ERROR message (if any).  
1 File not found.  
2 Access violation.  
3 Disk full or allocation exceeded.  
4 Illegal TFTP operation.  
5 Unknown transfer ID.  
6 File already exists.  
7 No such user.

Iteration 2

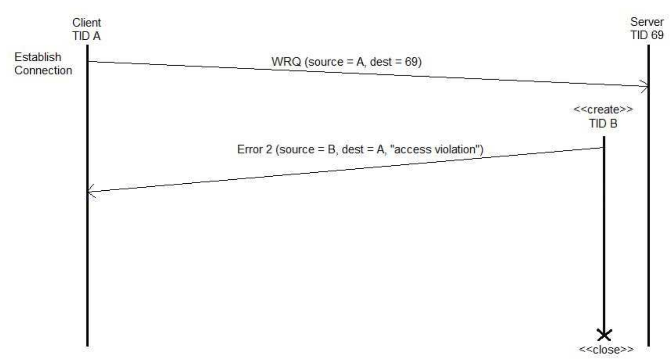
Timing Diagrams for ERROR Code (1, 2, 3, 6)

[ERROR Code 1] – File not found



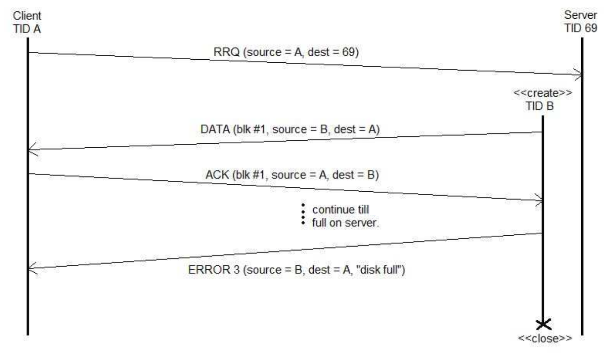
1. Client sends read request to destination 69 server
2. “temp.txt” file does not exist
3. Server sends an **ERROR (1)** packet to client (“ file not exist”)
4. Server closes socket
5. Client prints out **ERROR (1)** on server packet. Sent file “temp.txt” does not exist

[ERROR Code 2] – Access Violation



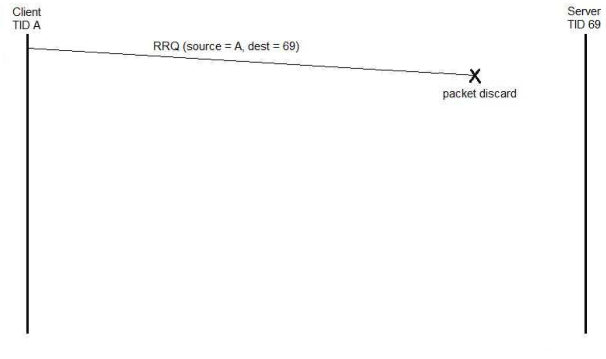
1. Client sends a write request to destination 69 server
2. Server traced client not having insufficient privileges to write.
3. Server sends an **ERROR (2)** packet to client (“access violation”)
4. Server closes packet
5. Client prints out **ERROR (2)** on server packet. Write was not successful, access violation

[ERROR Code 3] - Disk full or allocation exceeded



1. Client sends a read request to destination 69 server
2. Server successfully received the request and sends DATA packet (blk #1)
3. Client successfully received the DATA and sends an ACK (blk #1)
4. Step 2 and 3 occurs till the disk is full on server
5. Server sends an **ERROR (3)** packet to client (“disk full”)
6. Server closes packet
7. Client deletes the file that was not successful in completion
8. Client prints out **ERROR (3)** on server packet. Server disk full or allocation exceeded

[ERROR Code 6] – File already exists

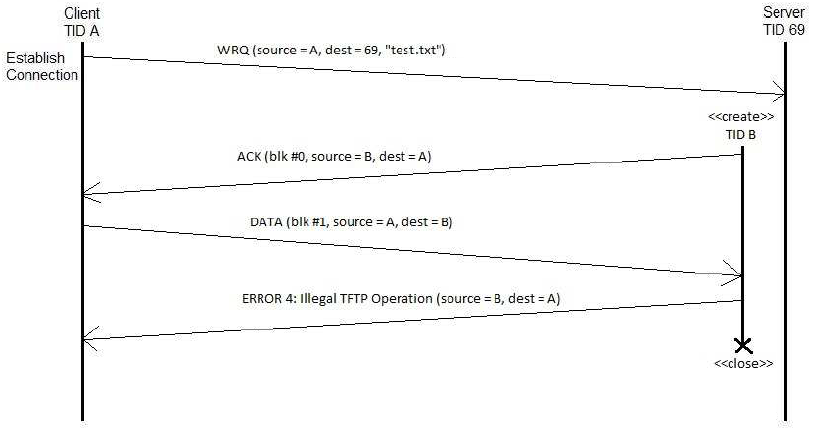


1. Client sends a read request to destination 69 server
2. Client finds existent DATA file
3. Client discards the packet transmission
4. Client prints out **ERROR (6)**. File already exist

Iteration 4

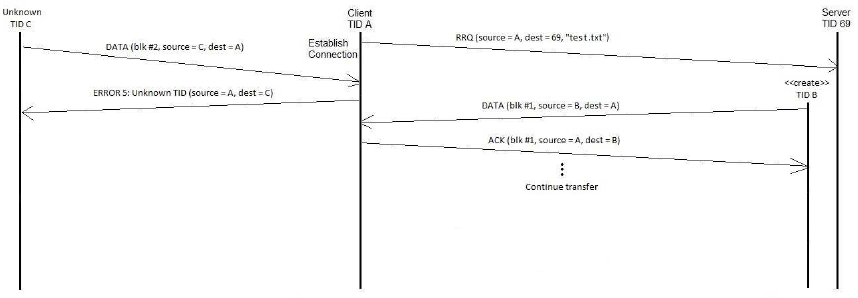
Timing Diagrams for ERROR Code (4, 5)

[ERROR Code 4] – Illegal TFTP operation



1. Client sends a write request to destination 69 server
2. Server receives the write request from Client
3. Server creates a new thread and send an ACK packet to client
4. Client receives the ACK packet
5. Client sends DATA packet but is incorrect
6. Server handled the incorrect DATA and sends an **ERROR(4)** to client. (“Illegal TFTP operation”)
7. Client prints out **ERROR(4)**. Illegal TFTP operation

[ERROR Code 5] – Unknown transfer ID.



1. Client sends a read request to destination 69 server
2. Server receives the read request from Client
3. An Unknown Thread sends a DATA packet to client
4. Client detects the unknown thread and send an **ERROR(5)** to unknown thread. (“Unknown transfer ID”)
5. Unknown prints out **ERROR(5)**. Unknown transfer ID
6. Server creates a new thread and send DATA packet to client
7. Client receives the DATA packet
8. Client sends ACK packet to server
9. Process continues the transfer

Iteration 5

File Transfer between Different Computers

Breakdown Responsibilities

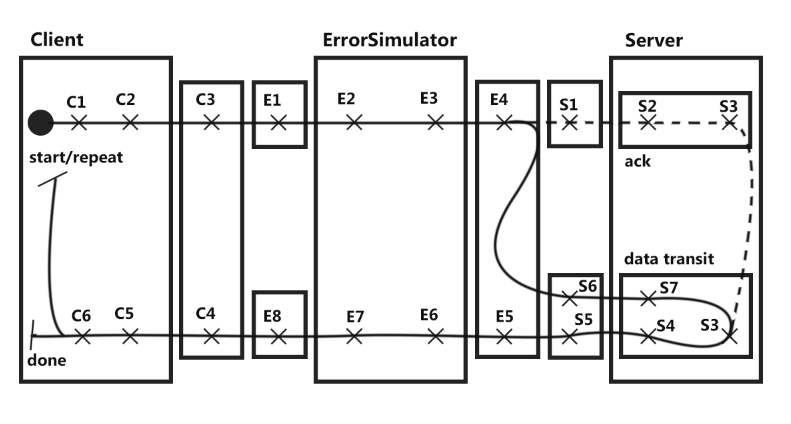
Anything with "\*" is their main contribution of the listed tasks.

We all did our best to contribute what we can and everyone made an impact in most classes.

|  |  |  |  |
| --- | --- | --- | --- |
| Ben Loo | Andrew Nguyen | Eric Morrissette | Ben Palko |
| Client\*  Server\*  TFTPConnection\*  ErrorSimulator  TFTPPacket  ThreadedConnection | FinalProject Doc\*  Diagrams\*  READ\_ME.txt\*  Client  TFTPConnection  UserInterface | Client\*  ErrorSimulator\*  Trello\*  TFTPConnection  TFTPPacket  ThreadedConnection | UserInterface\*  ThreadedConnection\*  Client  Server  TFTPConnection  Trello |

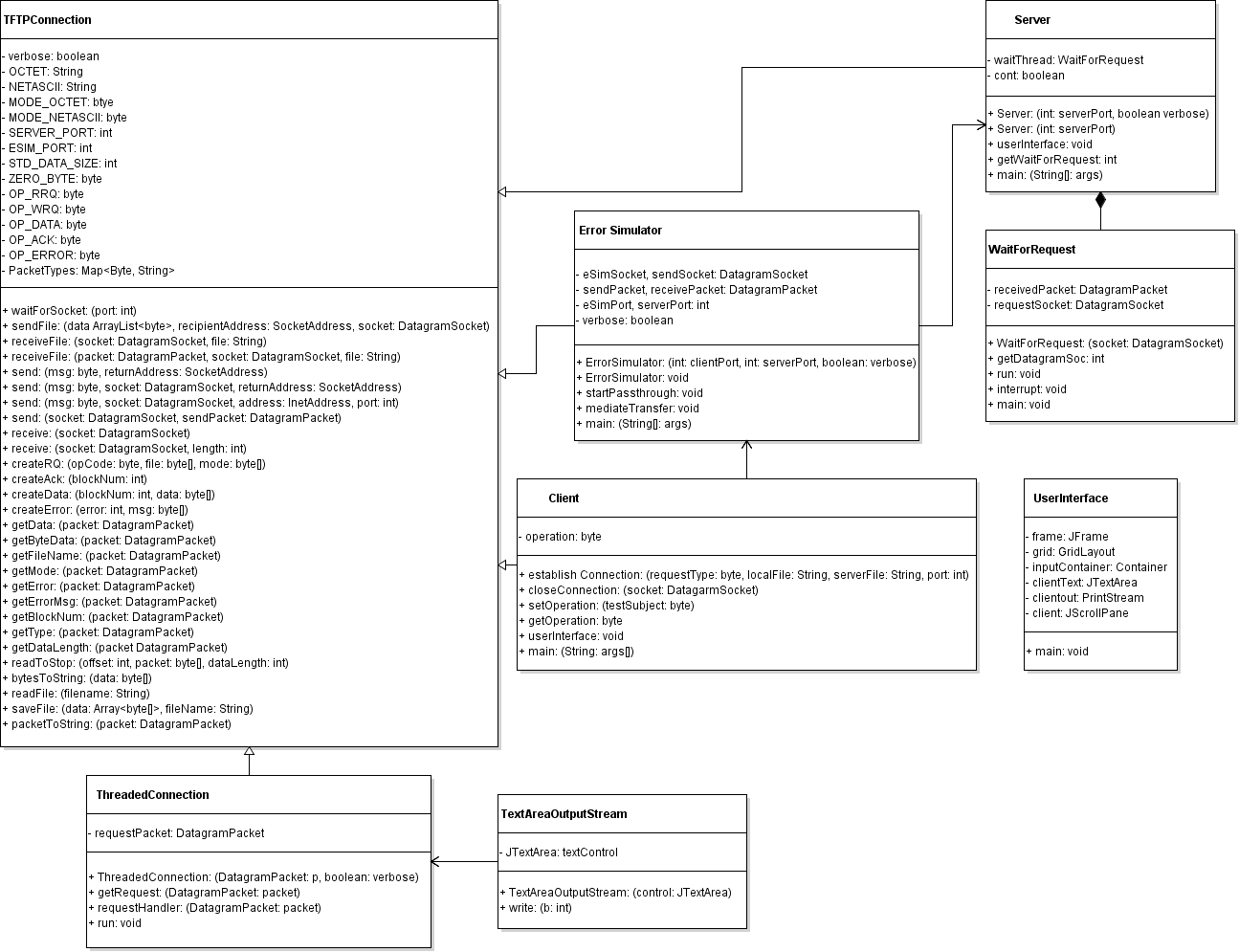
Diagrams

UCMs for Read/Write w/ Error Simulator



|  |  |  |
| --- | --- | --- |
| C1: Form Message (Input Selections w/ DATA or ACK)  C2: Create DatagramPacket w/ TFTP DATA Packet  C3: Send DatagramPacket  C4: Wait and Receive  C5: Extract Message  C6: OutputMessage  Repeat/Done | E1: Wait and Receive DatagramPacket  E2: Extract Message (Handle Errors)  E3: Form DatagramPacket  E4: Send DatagramPacket  E5: Wait and Receive  E6: Extract Message (Handle Errors)  E7: Form DatagramPacket  E8: Send DatagramPacket | S1: Receive DatagramPacket  S2: Extract Message  S3: Handle Request (ACK/DATA)  S4: Form DatagramPacket  S5: Send DatagramPacket  S6: Receive DatagramPacket  S7: Extract Message |

UML Class Diagram



How to Run and Troubleshoot

How to Run and Test Instructions

Running the application:

1. Run the UserInterface class (You don't have to run the classes separately any more).  
 - A new window should open up  
 - It'll ask for input commands

2. Select one of the four input commands (RRQ(1), WRQ(2), settings(3), quit(4)).  
 - If (1) or (2) is selected (continue on to "Step 3").  
 - If (3) is selected (jump to "Step 6").  
 - If (4) is selected, the application is done ("Jump to Step 9").

3. Input file to Read/Write on client (eg: "src\\test.txt").  
 - Press Enter.

4. Input file destination on server (eg: "Misc\\test.txt").  
 - Press Enter.  
 - Packet should be transferring back and fourth between client and server.  
 - Information will be shown on each class tab in User Interface.  
 - If it's Read, ACK will be sent and server will send DATA back.  
 - If it's Write. Data will be sent and server will send ACK back.

5. Once it is done, you can start again (Go back to "Step 2").

6. Setting(3) - User Interface will ask for an input for Verbose (eg: "true/false").  
 - If (true), User Interface will show and trace the packets being sent back and fourth. - If (false), file transfer will be hidden.

7. User Interface will ask for an input for Testing (eg: "true/false").  
 - If (true), the packet will transmit with ErrorSimulator.  
 7.1: Instructions will prompt and ask for what error handling the user would like to see.   
 Please follow the instruction it is shown in the interface.  
 (eg: Verbose > TypeOfError > Blk# > Type > Speed > TypeOfData > (Go back to “Step 2))   
 - If (false), the packet will ignore ErrorSimulator. Client will send and receive directly to/from Server.

8. User Interface will jump back to the main four input commands (Go back to "Step 2").

9. Close the User Interface Window.

END

Test Table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test ID | Test Name/Description | Test Steps | Expected Results | Actual Results | Status |
| 0001 | UIPrintOut | Initialize toString then compare with given string | RRQ(1), WRQ(2), settings(3), quit(4): | RRQ(1), WRQ(2), settings(3), quit(4): | Good |
| 0002 | createRQ | Input opcode, file, mode type. Call function createRQ and see if the format is correct | 0opCodefilename  0mode0 | 0opCodefilename  0mode0 | Good |
| 0003 | Check if verbose works | Select true in initial set up. See if detail transmit appear. | Sending:  Type: RRQ  File name: Misc\\test.txt  Mode: octet | Sending:  Type: RRQ  File name: Misc\\test.txt  Mode: octet | Good |
| 0004 | Check if verbose works 2 | Select false in initial set up. See if detail transmit appear. | N/A | N/A | Good |
| 0005 | Get Block# at end transit. Client to Server | Create packet and send to server from client. Getblock# and compare. | 1649 | 1649 | Good |

Source Code