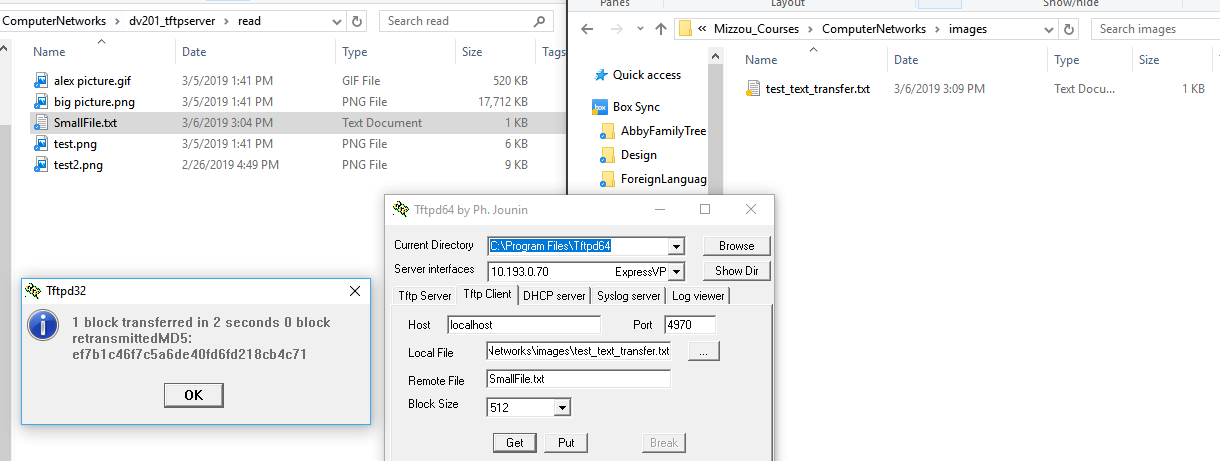
DV201 (Software Engineering) Assignment 3 (TFTP Server)

By: Alex and Fabian

# Problem 1 (TFTP Server):

**Read Request From Client with file less than 512 Bytes (Screenshot):**

In the following test the windows explorer window shows the contents of the “read” directory for the tftp server. This directory contains all the available files that the client user can get. The explorer window on the right shows the local directory for the client user where the resultant image was stored. The TFTP client at the bottom shows that we will connect to localhost on port 4970 and in this case get the file SmallFile.txt. Since the file is less that 512 bytes it only requires 1 block to transfer. When the “Get” button is pressed the transfer begins and the popup message indicates that the 1 block has been successfully transferred and the file on the local filesystem is shown on the left (test\_text\_transfer.txt)



**Socket and sendSocket:**

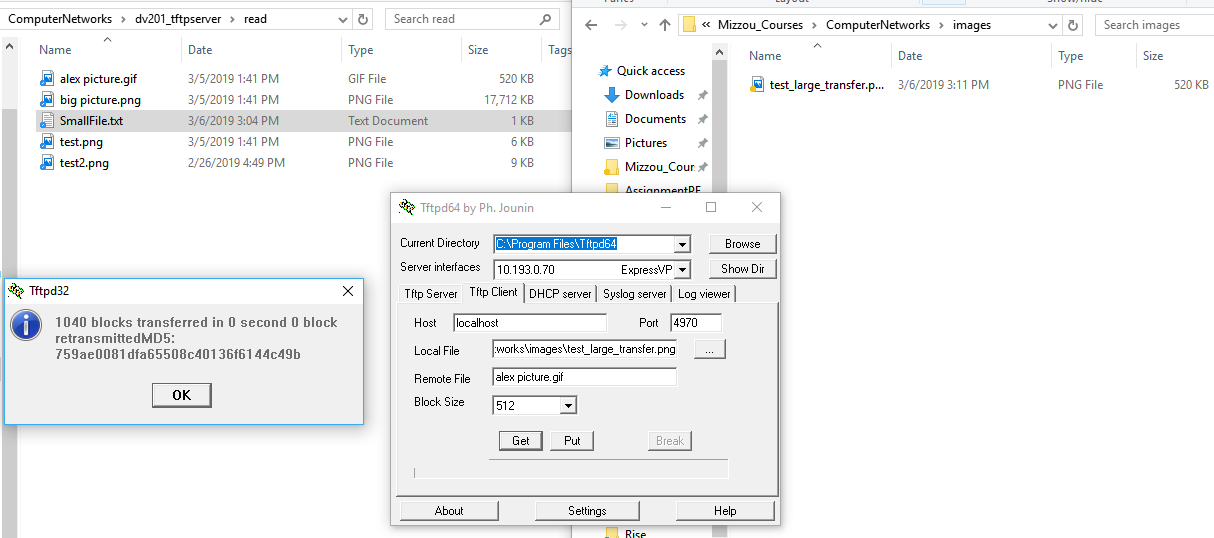
The reason that the server uses socket and sendSocket is that socket is used to start the server on a known port so incoming connections can be accepted. sendSocket is then used when a connection is accepted to open a new port so that the file/data can be transferred from the server to the client. This is done so that the known port the server runs on can be open to accept new incoming requests while the server is handling/transmitting files to clients on other ports. sendSocket picks any available port while socket uses a well known port so that users know which port to access the server on.

# Problem 2:

**Multiple Read Requests from Client with files greater than 512 Bytes (Screenshot):**

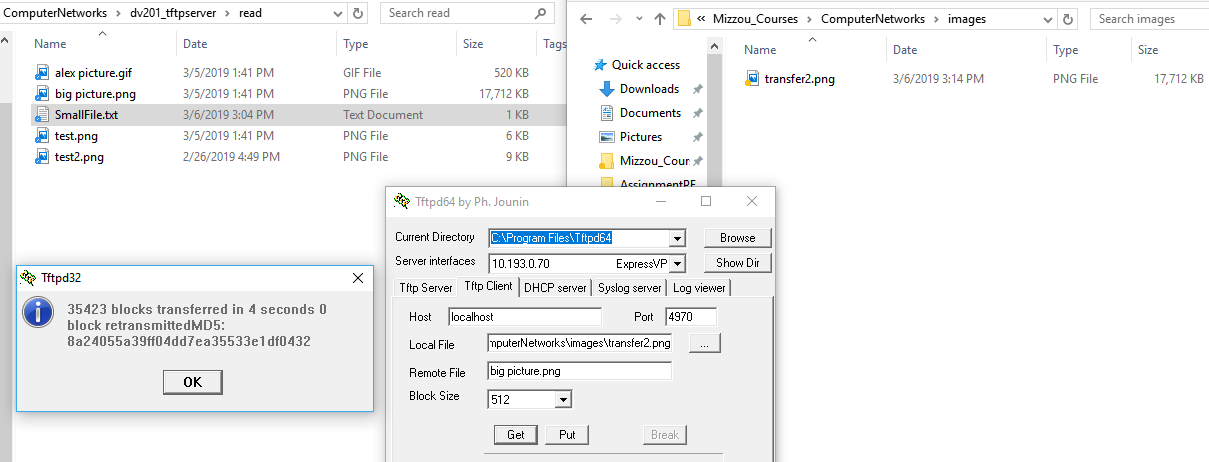
**Large Transfer 1:**

This test was preformed in the same way as the test for problem 1 but this time with larger files. As you can see in the confirmation screenshot, for the file “alex picture.gif”, which is 520 KB in size, there were 1040 blocks transferred. The completed transfer to the local system is shown in the right explorer window. You can see that the size of the file on the server and on the local filesystem is the same.

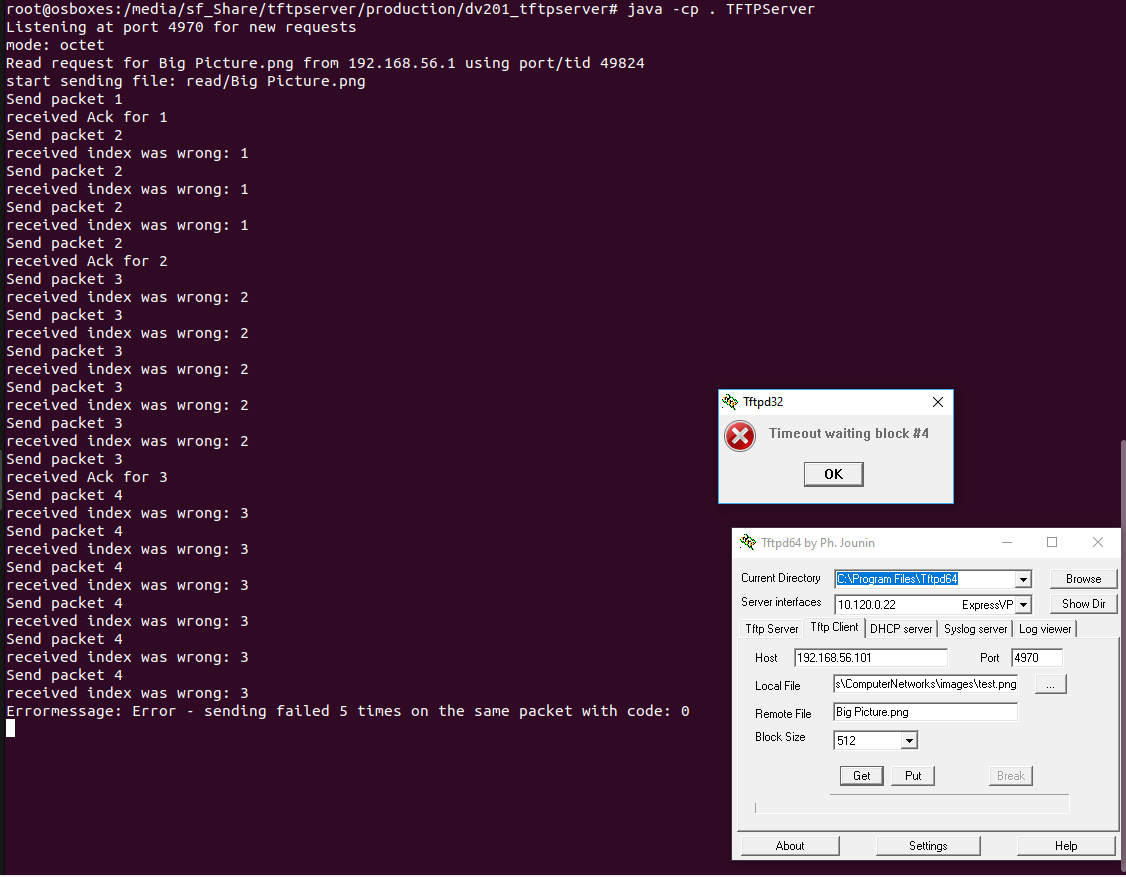


**Large Transfer 2:**

As you can see in the confirmation screenshot, for the file “big picture.png”, which is 17,712 KB in size, there were 35423 blocks transferred. This transfer took 4 seconds because of how large the image was but the server was still capable of handling the transfer. The completed transfer to the local system is shown in the right explorer window. You can see that the size of the file on the server and on the local filesystem is the same.



**Timeouts and Retransmissions:** To test the timeouts and retransmissions we added an artificial delay to all requests coming into the server using the following UNIX command ( tc qdisc add dev eth0 root netem delay 10000ms). This command adds a 10 second delay to all inbound communication on the selected port. This means that it will not receive the ACK (acknowledgement) that the request was successful from the client until at least 10 seconds after the packet was sent. The server will attempt to retransmit a packet 5 times before it will not longer transmit anything more and the client application has a 3 second timeout in its settings. The screenshot below shows the terminal of the VM where the server is running. There is also the local TFTP client in the bottom right corner. The server tries to retransmit a packet if it does not receive the correct ACK. Since there is a delay in the network the ACK packets are all very late which causes many retransmissions until eventually on block 4 the timeout is too long and the server stops trying to retransmit and sends a timeout error.



**VG-Task 1:**

# Problem 3 (Error Handling):

In the following screenshots the terminal window is the server running on a virtual machine

## Code 0:

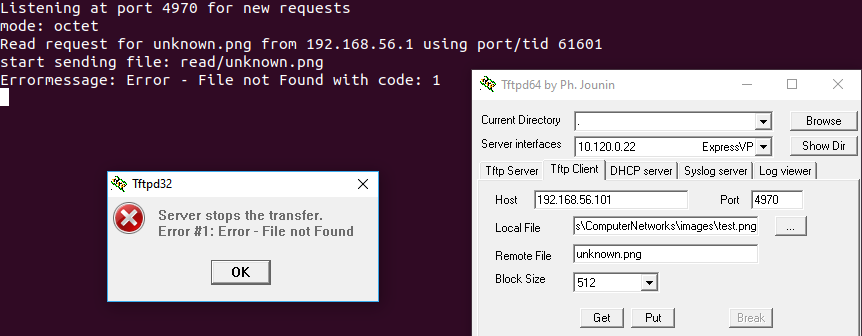
According to the RFC 1350 spec the Code 0 Error code indicates an undefined error and you should check the error message if there is any. Our server can send a Code 0 for a few conditions. In the screenshot below the server attempted to retransmit the same packet 5 times without receiving an acknowledgement and after not receiving an ACK message the 5th time it sent the Code 0 error message. Our program will also send a code 0 on the following conditions:

* Attempting to using a mode other than the octet (“binary”) mode
* Sending or Receiving a file larger than 32 MB
* Unknown Errors/Failures
* Timeout waiting Errors
* Errors receiving incoming data

## 

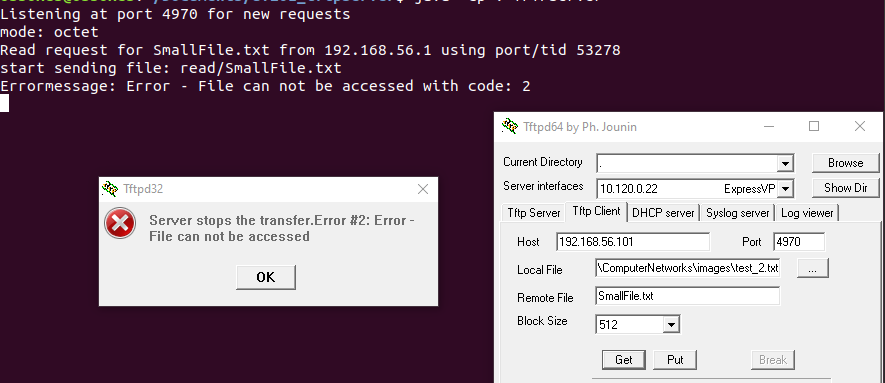
## Code 1:

According to the RFC 1350 spec the Code 1 Error code indicates that the File is not Found. Our server will send this error code if the client user requests a file that does not exist on the server. For this test the client user requested “unknown.png”. The server could not find the file and send the Code 1: File not found error to the client.



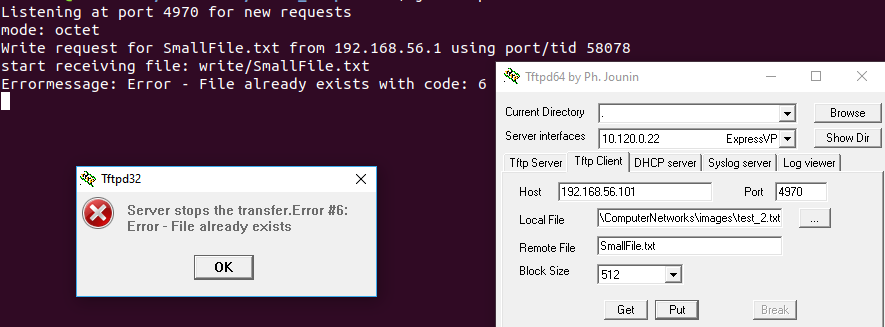
## Code 2:

According to the RFC 1350 spec the Code 1 Error code indicates that there is an access violation. For this test I changed the permissions of the file “SmallFile.txt” using “chmod a-r” which makes it so that no one user can read the file. This means when the server attempts to access the file it will cause an access violation and send a Code 2 error as shown by the error message window.



## Code 6:

According to the RFC 1350 spec the Code 1 Error code indicates that the File Already Exists. Our server will send this error code if the file that the client user is trying to “Put” on the server already exists. In this case the file “SmallFile.txt” already exists on the server so the server will not allow it to be overwritten and the server sends a code 6 error message.



**VG-task 2:**

Code 3:

Code 4: