Report

1DV701 Computer Networks - an introduction

Assignment 3: TFTP Server

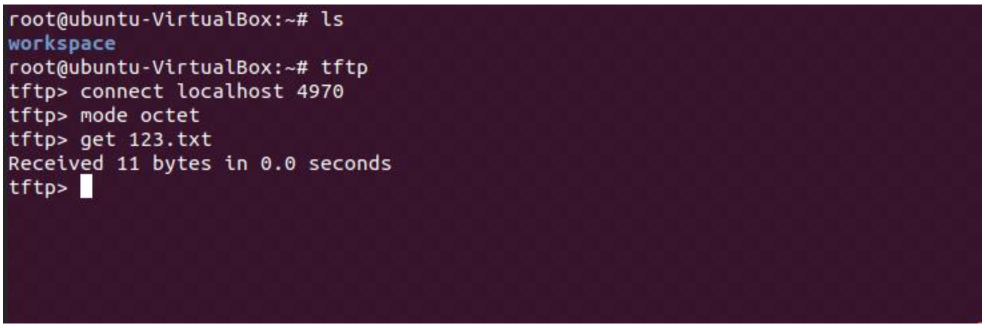
hz222bp

This report would analyze the TFTP server according to RFC1350 with the screenshot. The virtual machine with Linux system is used for running both server and client. After executing the command "build.sh" to compile, the server can run by the command "run.sh", and the client was installed by "sudo apt-get install tftp". Then the specific screenshot for each question are analyzed as following:

Problem 1

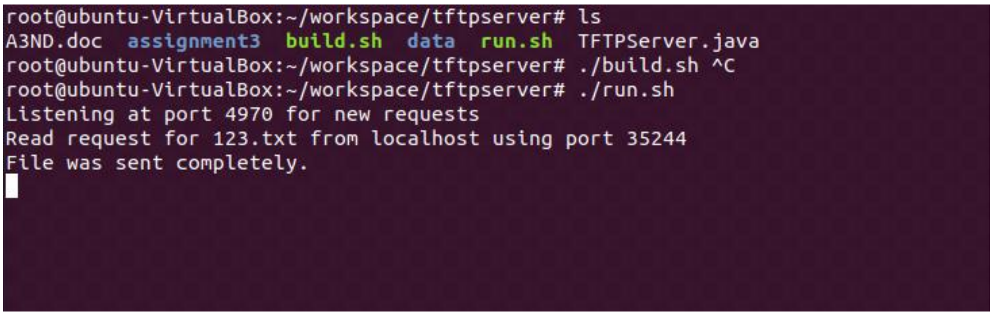
On the basis of the provided TFTPServer starter code and steps, the server listens to the actual port 35244. The server get the txt files from the dictionary /tmp for showing the function. As the screenshot shows, the client can send a request to read a small file that is shorter than 512 bytes.

Client requests small file:



```
root@ubuntu-VirtualBox:~# ls
workspace
root@ubuntu-VirtualBox:~# tftp
tftp> connect localhost 4970
tftp> mode octet
tftp> get 123.txt
Received 11 bytes in 0.0 seconds
tftp>
```

Server sends small file:



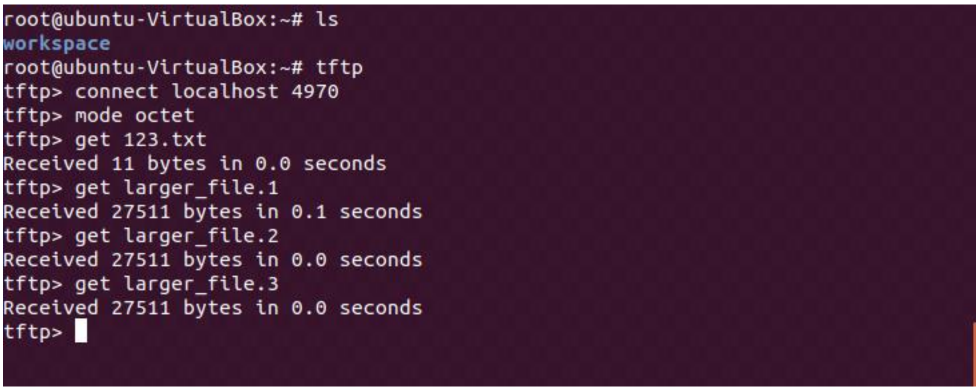
```
root@ubuntu-VirtualBox:~/workspace/tftpserver# ls
A3ND.doc  assignment3  build.sh  data  run.sh  TFTPServer.java
root@ubuntu-VirtualBox:~/workspace/tftpserver# ./build.sh ^C
root@ubuntu-VirtualBox:~/workspace/tftpserver# ./run.sh
Listening at port 4970 for new requests
Read request for 123.txt from localhost using port 35244
File was sent completely.
```

Although both socket and sendSocket can deal with OP_WRQ request, the socket can also deal OP_RRQ request, and sendSocket is used for dealing OP_ACK and OP_ERR requests. Thus the TFTPServer use both of them for implementing the comprehensive function.

Problem 2

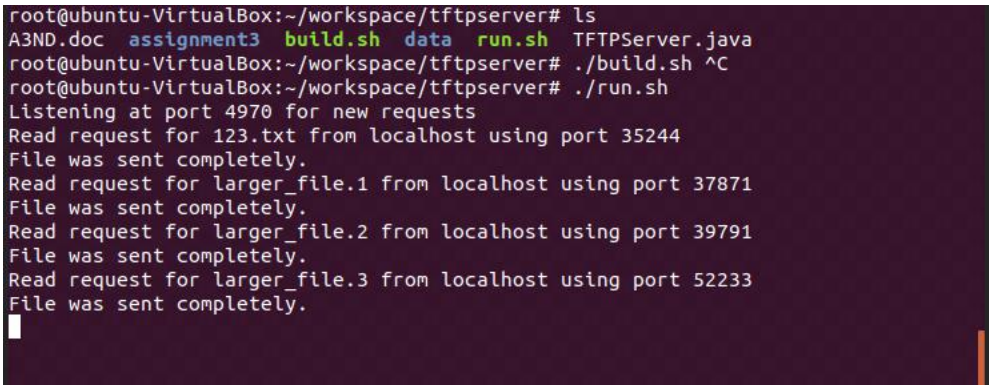
The client also can request to the multiple large files. The data transfer ports are chosen independently by the sender and receiver during the transfer initialization. The timeout functionality with proper re-transmissions are implemented, but I just basically test this function by killing the client during a transmission session, the sever would try to resend the packet five times before giving up.

Client requests multiple large files:



```
root@ubuntu-VirtualBox:~# ls
workspace
root@ubuntu-VirtualBox:~# tftp
tftp> connect localhost 4970
tftp> mode octet
tftp> get 123.txt
Received 11 bytes in 0.0 seconds
tftp> get larger_file.1
Received 27511 bytes in 0.1 seconds
tftp> get larger_file.2
Received 27511 bytes in 0.0 seconds
tftp> get larger_file.3
Received 27511 bytes in 0.0 seconds
tftp> 
```

Server sends multiple large files:



```
root@ubuntu-VirtualBox:~/workspace/tftpserver# ls
A3ND.doc  assignment3  build.sh  data  run.sh  TFTPServer.java
root@ubuntu-VirtualBox:~/workspace/tftpserver# ./build.sh ^C
root@ubuntu-VirtualBox:~/workspace/tftpserver# ./run.sh
Listening at port 4970 for new requests
Read request for 123.txt from localhost using port 35244
File was sent completely.
Read request for larger_file.1 from localhost using port 37871
File was sent completely.
Read request for larger_file.2 from localhost using port 39791
File was sent completely.
Read request for larger_file.3 from localhost using port 52233
File was sent completely.
```

VG-task 1

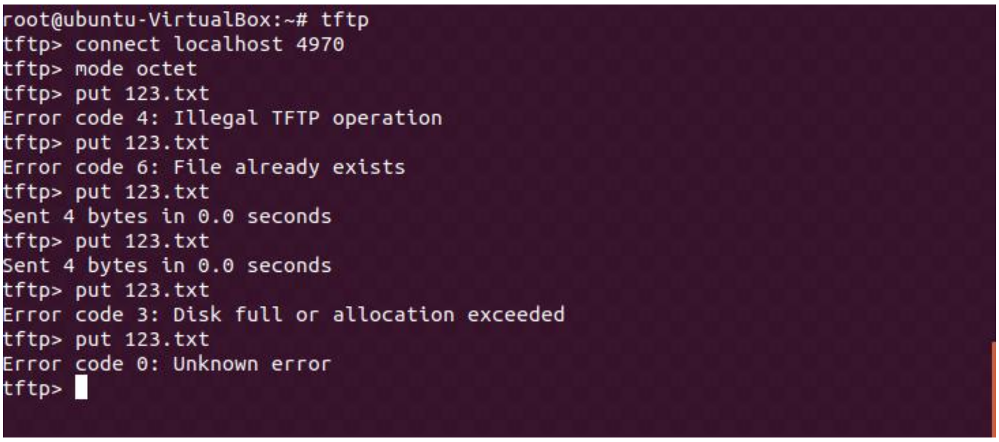
I also tried to obverse the traffic of the read request by Wireshark, however, my understanding is very limited, and my virtual machine has some problem to download Wireshark. Thus for this task, I just list the differences between read and write requests are listed in the following table:

Request	Write	Read
Opcode	2	1
Server	It sends ACK packet to client, then waits for DATA packet from client.	It sends DATA packet to client, then waits for ACK packet from client.
Method	PUT	GET

Problem 3 and VG-task 2

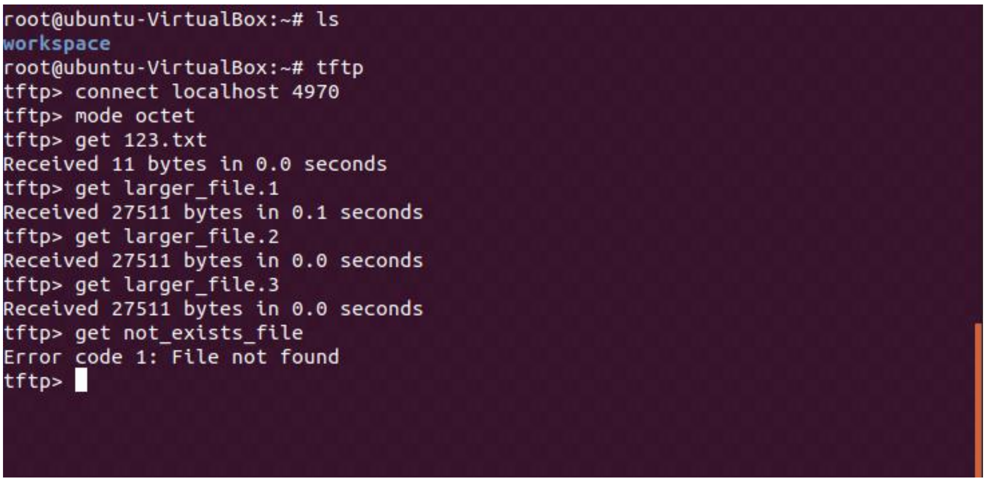
This chapter would discuss the TFTP error handling according to the RFC1350 specification. The error codes 0, 1, 2, 3, 4, 6 are illustrated with screenshot as well as 5 is explained with both the codes and screenshot.

Error code 0: Unknown error



```
root@ubuntu-VirtualBox:~# tftp
tftp> connect localhost 4970
tftp> mode octet
tftp> put 123.txt
Error code 4: Illegal TFTP operation
tftp> put 123.txt
Error code 6: File already exists
tftp> put 123.txt
Sent 4 bytes in 0.0 seconds
tftp> put 123.txt
Sent 4 bytes in 0.0 seconds
tftp> put 123.txt
Error code 3: Disk full or allocation exceeded
tftp> put 123.txt
Error code 0: Unknown error
tftp> 
```

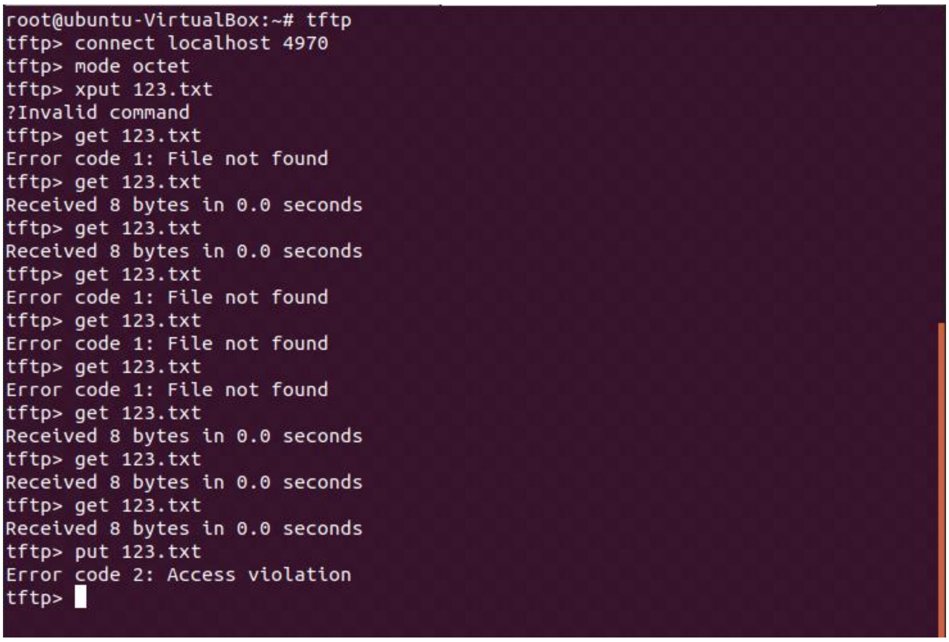
Error code 1: The client wants to read a file that does not exist.



```
root@ubuntu-VirtualBox:~# ls
workspace
root@ubuntu-VirtualBox:~# tftp
tftp> connect localhost 4970
tftp> mode octet
tftp> get 123.txt
Received 11 bytes in 0.0 seconds
tftp> get larger_file.1
Received 27511 bytes in 0.1 seconds
tftp> get larger_file.2
Received 27511 bytes in 0.0 seconds
tftp> get larger_file.3
Received 27511 bytes in 0.0 seconds
tftp> get not_exists_file
Error code 1: File not found
tftp> 
```

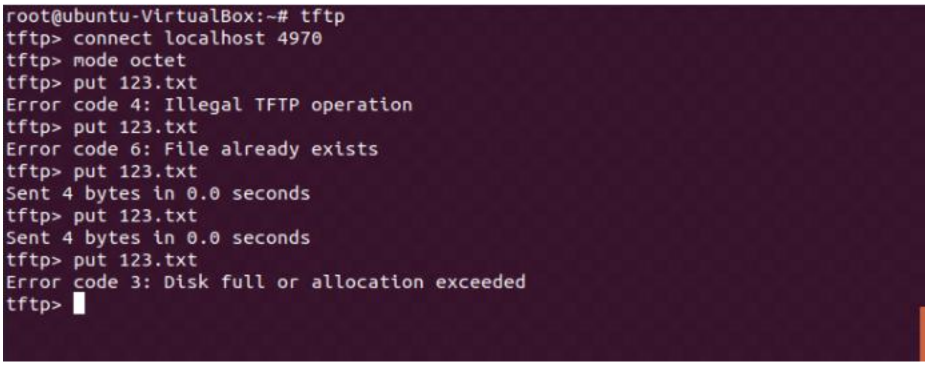
Error code 2:

Access violation



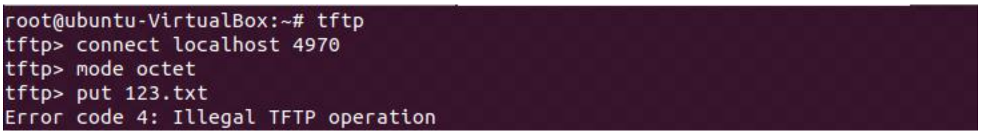
```
root@ubuntu-VirtualBox:~# tftp
tftp> connect localhost 4970
tftp> mode octet
tftp> xput 123.txt
?Invalid command
tftp> get 123.txt
Error code 1: File not found
tftp> get 123.txt
Received 8 bytes in 0.0 seconds
tftp> get 123.txt
Received 8 bytes in 0.0 seconds
tftp> get 123.txt
Error code 1: File not found
tftp> get 123.txt
Error code 1: File not found
tftp> get 123.txt
Error code 1: File not found
tftp> get 123.txt
Received 8 bytes in 0.0 seconds
tftp> get 123.txt
Received 8 bytes in 0.0 seconds
tftp> get 123.txt
Received 8 bytes in 0.0 seconds
tftp> put 123.txt
Error code 2: Access violation
tftp> 
```

Error code 3: Disk full or allocation exceeded



```
root@ubuntu-VirtualBox:~# tftp
tftp> connect localhost 4970
tftp> mode octet
tftp> put 123.txt
Error code 4: Illegal TFTP operation
tftp> put 123.txt
Error code 6: File already exists
tftp> put 123.txt
Sent 4 bytes in 0.0 seconds
tftp> put 123.txt
Sent 4 bytes in 0.0 seconds
tftp> put 123.txt
Error code 3: Disk full or allocation exceeded
tftp> 
```

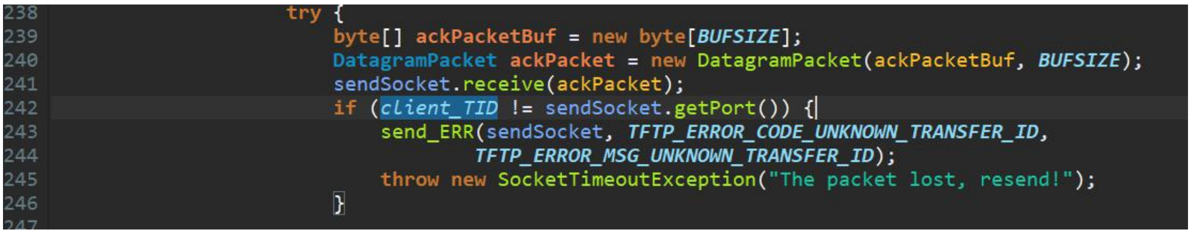
Error code 4: Illegal operation



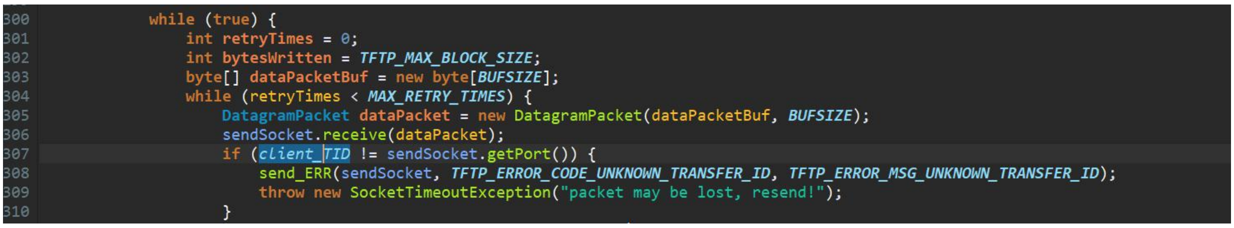
```
root@ubuntu-VirtualBox:~# tftp
tftp> connect localhost 4970
tftp> mode octet
tftp> put 123.txt
Error code 4: Illegal TFTP operation
```

Error code 5: Unknown transfer ID

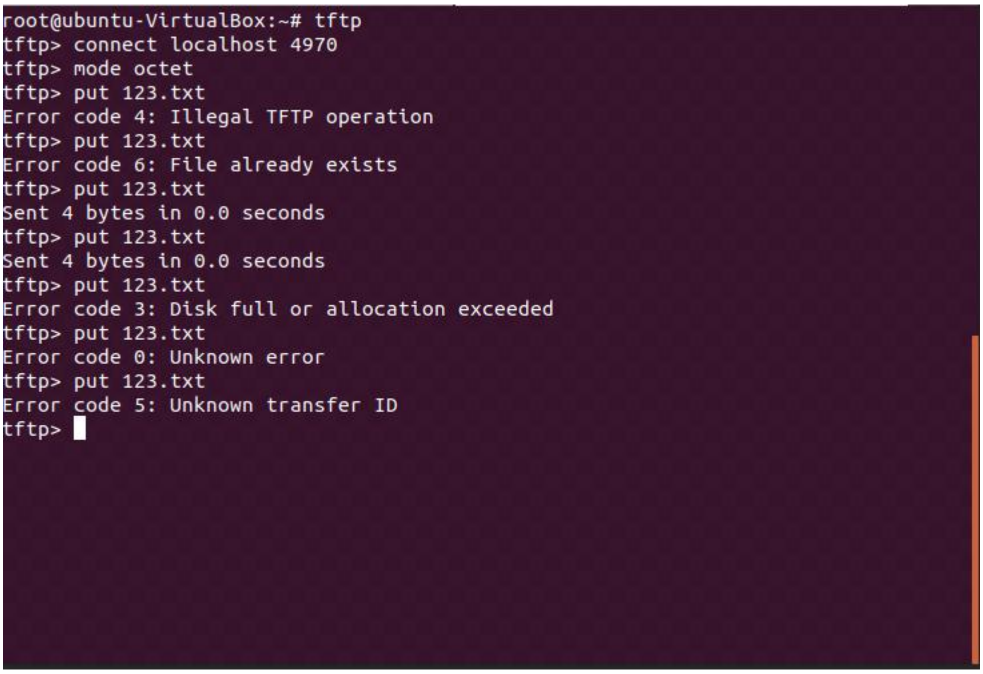
It is the codes part for generating the error code 5. In my implementation, there are two kinds of assumption for the unknown transfer ID.



```
try {
    byte[] ackPacketBuf = new byte[BUFSIZE];
    DatagramPacket ackPacket = new DatagramPacket(ackPacketBuf, BUFSIZE);
    sendSocket.receive(ackPacket);
    if (client_TID != sendSocket.getPort()) {
        send_ERR(sendSocket, TFTP_ERROR_CODE_UNKNOWN_TRANSFER_ID,
                TFTP_ERROR_MSG_UNKNOWN_TRANSFER_ID);
        throw new SocketTimeoutException("The packet lost, resend!");
    }
```



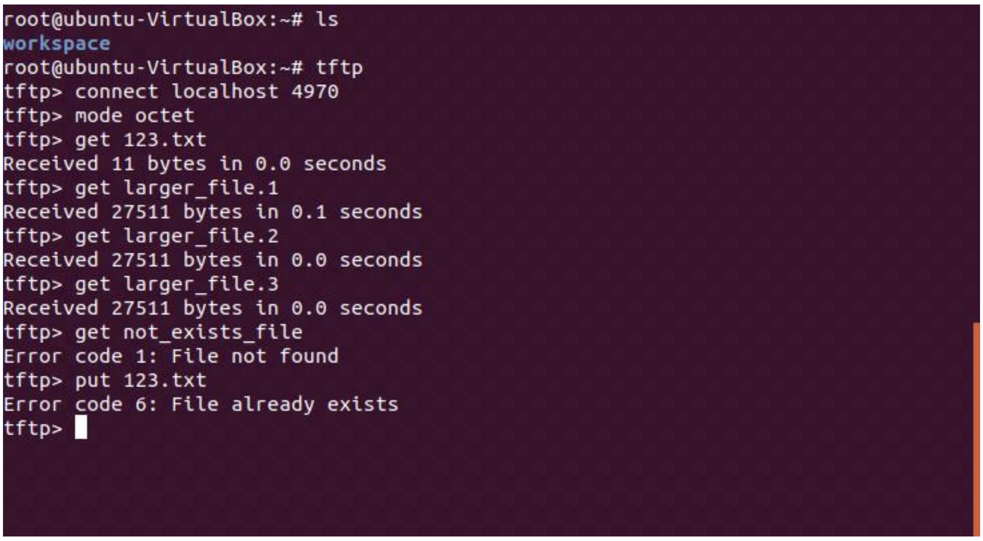
```
            while (true) {
                int retryTimes = 0;
                int bytesWritten = TFTP_MAX_BLOCK_SIZE;
                byte[] dataPacketBuf = new byte[BUFSIZE];
                while (retryTimes < MAX_RETRY_TIMES) {
                    DatagramPacket dataPacket = new DatagramPacket(dataPacketBuf, BUFSIZE);
                    sendSocket.receive(dataPacket);
                    if (client_TID != sendSocket.getPort()) {
                        send_ERR(sendSocket, TFTP_ERROR_CODE_UNKNOWN_TRANSFER_ID, TFTP_ERROR_MSG_UNKNOWN_TRANSFER_ID);
                        throw new SocketTimeoutException("packet may be lost, resend!");
                    }
```



```
root@ubuntu-VirtualBox:~# tftp
tftp> connect localhost 4970
tftp> mode octet
tftp> put 123.txt
Error code 4: Illegal TFTP operation
tftp> put 123.txt
Error code 6: File already exists
tftp> put 123.txt
Sent 4 bytes in 0.0 seconds
tftp> put 123.txt
Sent 4 bytes in 0.0 seconds
tftp> put 123.txt
Error code 3: Disk full or allocation exceeded
tftp> put 123.txt
Error code 0: Unknown error
tftp> put 123.txt
Error code 5: Unknown transfer ID
tftp> 
```

Error code 6:

The client wants to write to a file that already exists



```
root@ubuntu-VirtualBox:~# ls
workspace
root@ubuntu-VirtualBox:~# tftp
tftp> connect localhost 4970
tftp> mode octet
tftp> get 123.txt
Received 11 bytes in 0.0 seconds
tftp> get larger_file.1
Received 27511 bytes in 0.1 seconds
tftp> get larger_file.2
Received 27511 bytes in 0.0 seconds
tftp> get larger_file.3
Received 27511 bytes in 0.0 seconds
tftp> get not_exists_file
Error code 1: File not found
tftp> put 123.txt
Error code 6: File already exists
tftp> 
```