# Basic Settings:

Server: Port is 4950

Parser Usage: ./Parser old\_file\_name new\_file\_name

Filenames will be prefixed by the current directory of the Server, so just type the relative path will work.

To run the tests, put Server and Parser and files used to send in the same folder.

Run ‘tftp 0.0.0.0 4950’ on the same folder, too.

For the given 8 test cases, we divided them into 3 test suites. The first suite tests cases 1 through 4. And the second tests the rest. Finally, the last suite tests the BONUS feature.

There’re four test files in total, namely **1\_file2047 2\_file2048 3\_file2CR** and **4\_pycharm.dmg**. The fourth file is about 200MB.

# Test Suite 1:

We transferred first 2 files in binary mode, and the last 2 files in ascii mode.

We named the transferred file with prefix ‘c’, which indicates ‘client’.

After transferring process, we use ‘diff’ command to compare files. Figure 1 shows the process and result, from which two files are the same.

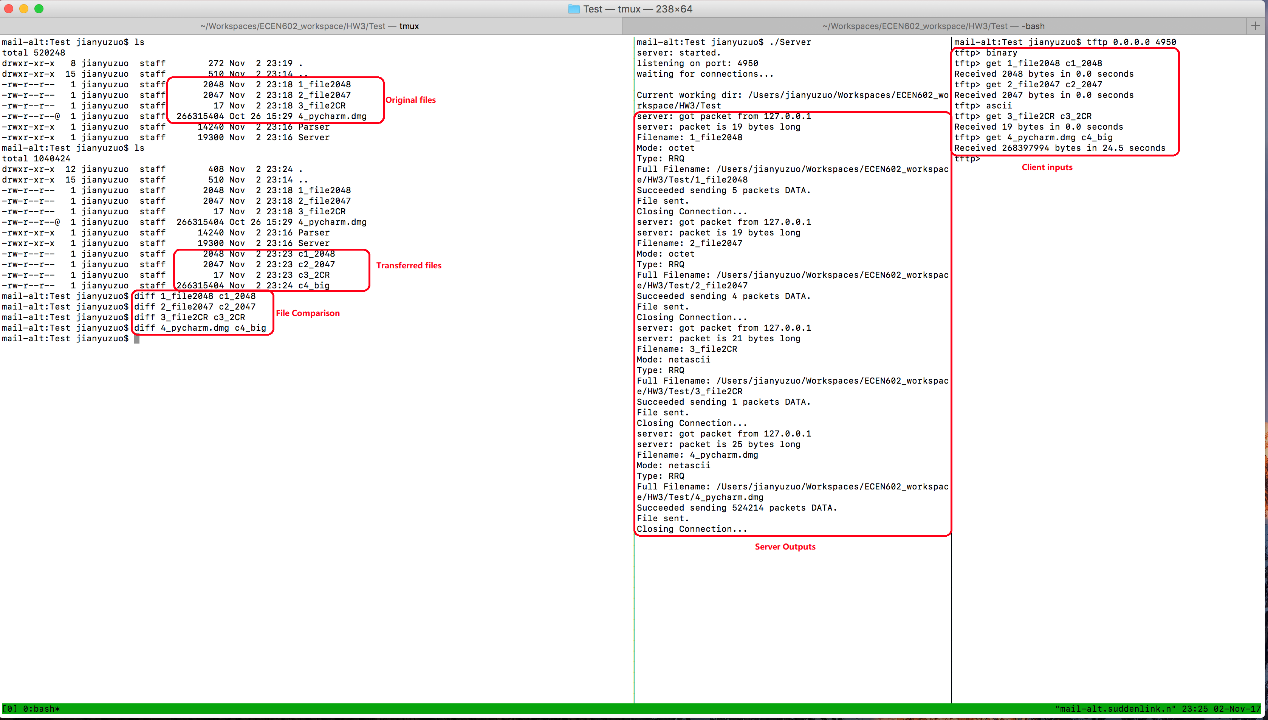


Figure 1

# Test Suite 2:

In this test suite, we will test for multi-connections, error message and timeouts.

Figure 2 shows the result that 4 connections to the server requesting for the same file, which works fine.

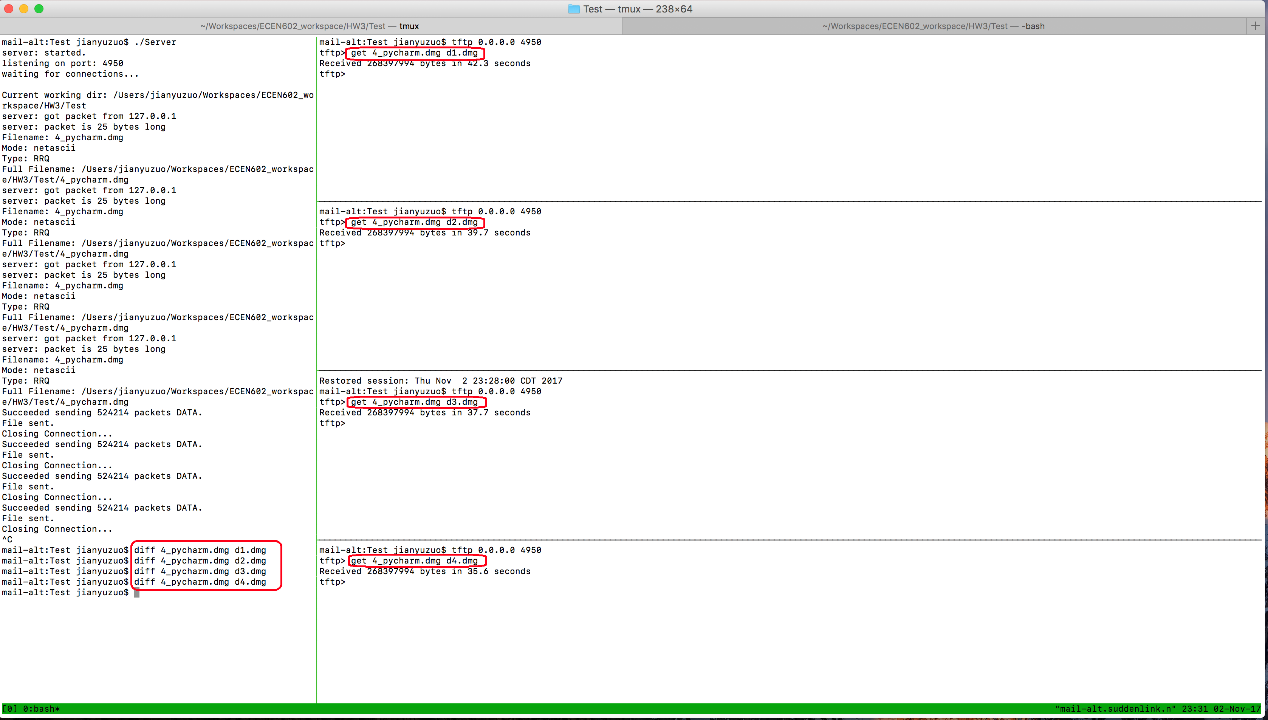


Figure 2

Figure 3 shows the timeout figure.

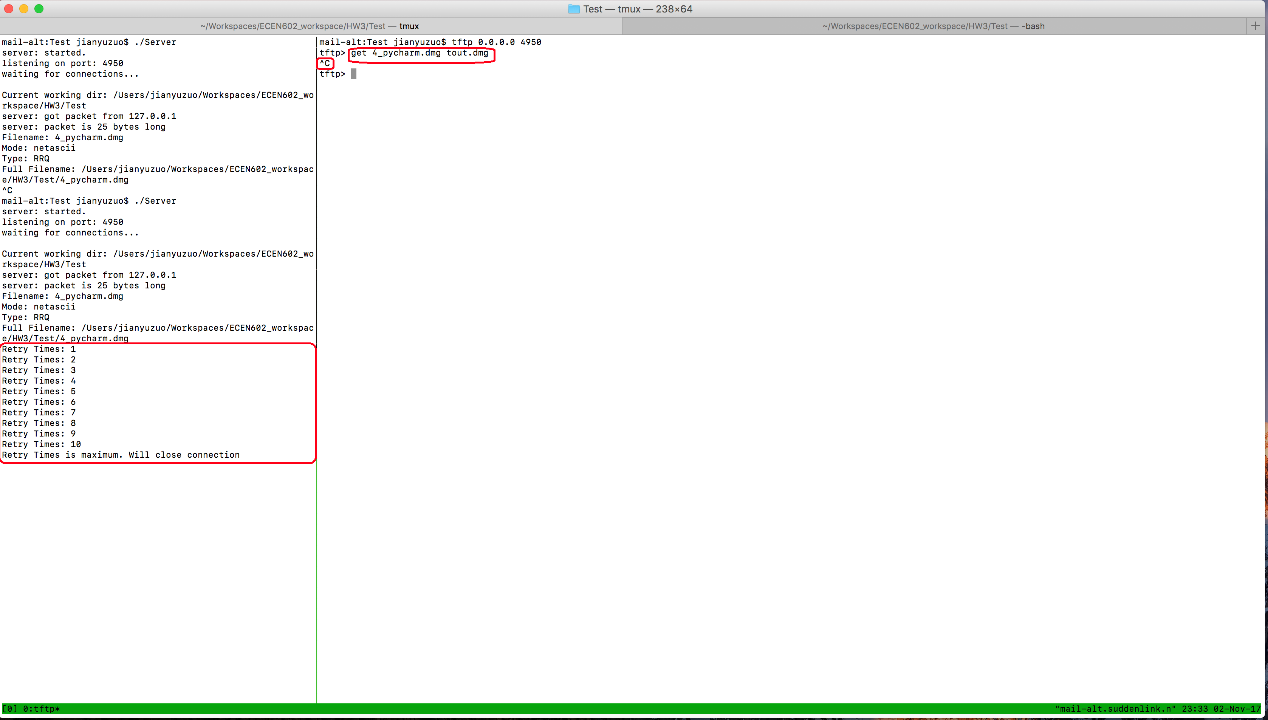


Figure 3

Figure 4 shows the ERROR message type

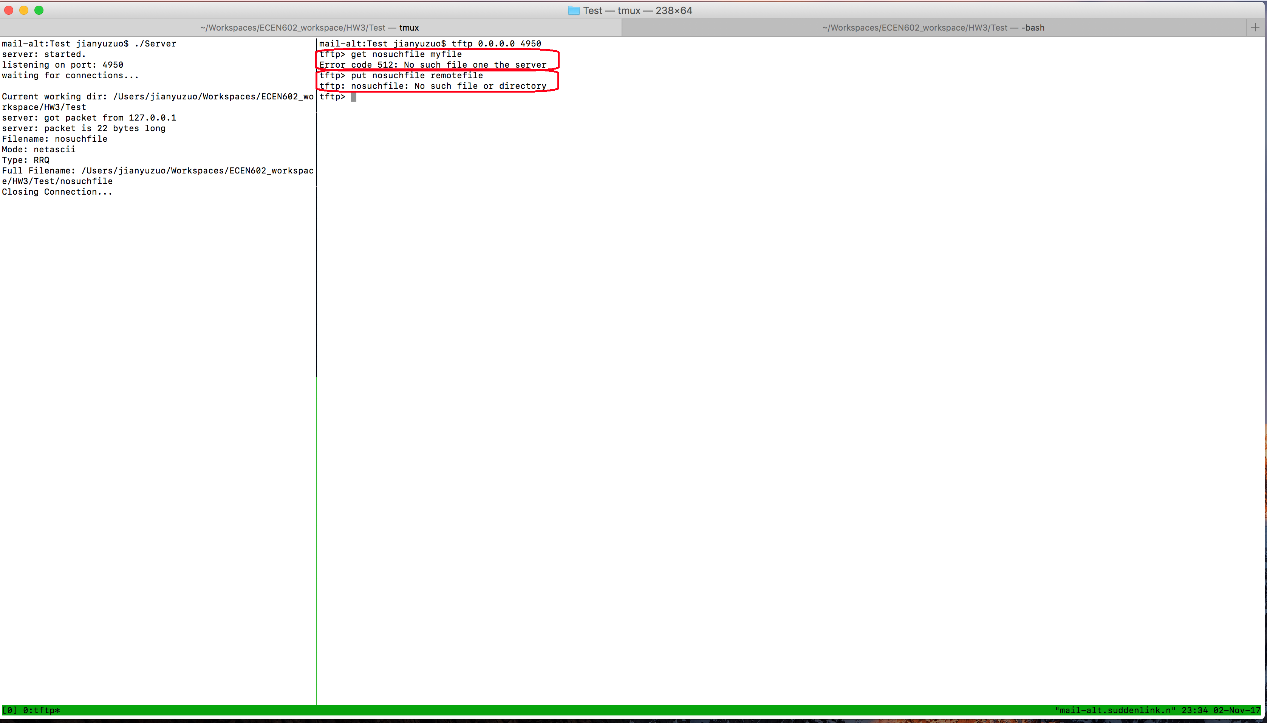


Figure 4

# Test Suite 3:

In this test suite, we will test for BONUS feature WRQ.

We first test for mode ‘binary’. And compare the files. Figure 5 shows the result.

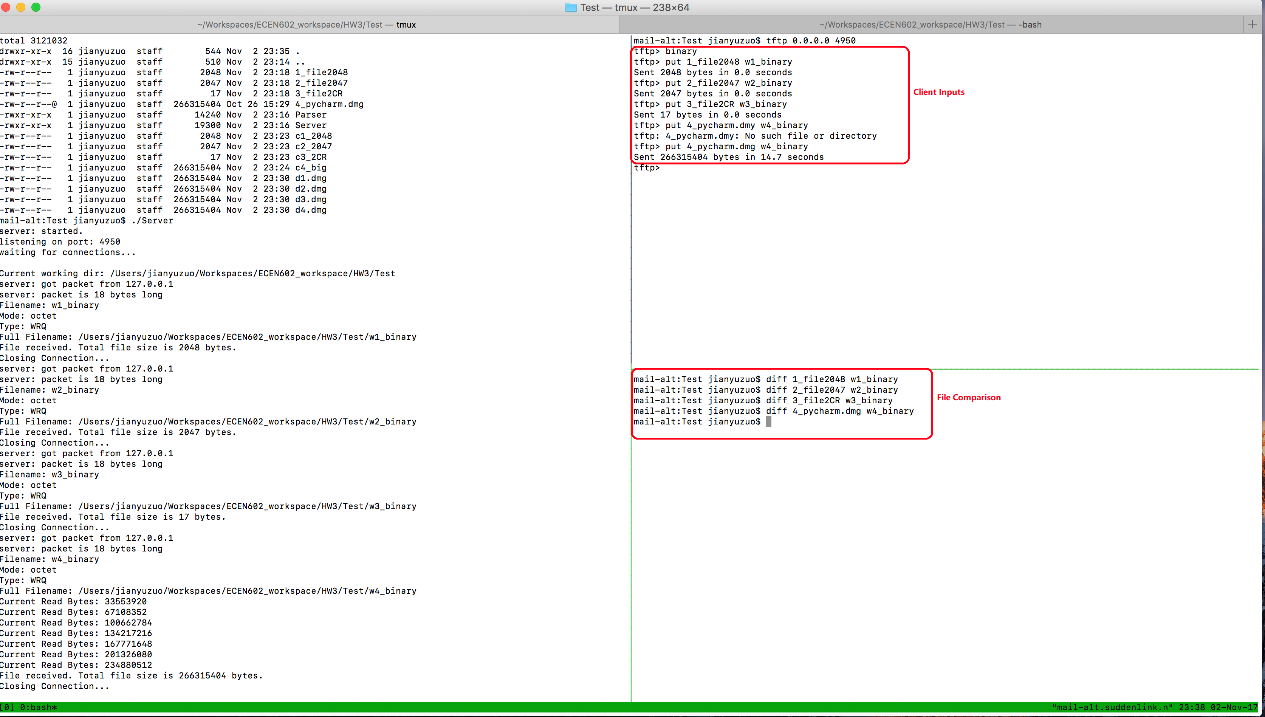


Figure 5

We then test for mode ‘ascii’. Our server only stores the data received to the file directly. To get the original file, we’ve implemented a Parser. The Parser is just doing the reverted job, which is done before sending in mode ‘ascii’. Figure 6 shows the result:

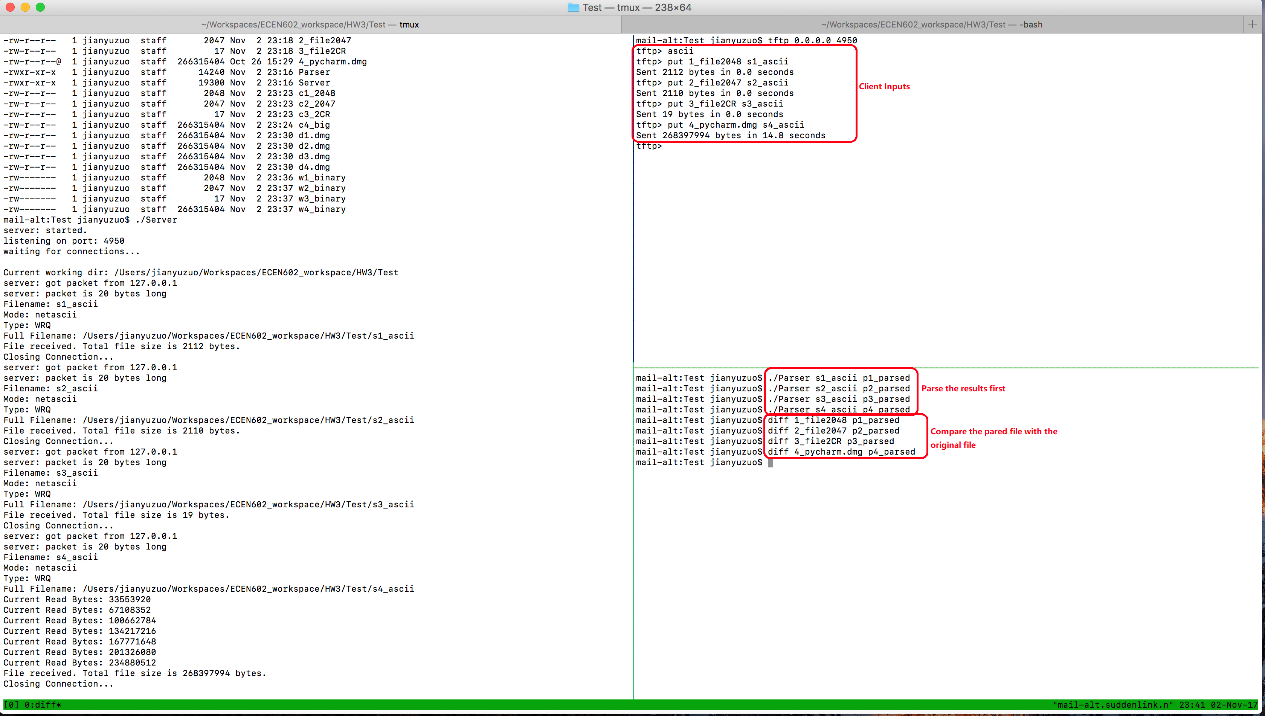


Figure 6