# Systems Programming – Socket Application Testing

Student: Callum Carmicheal, S1829709

## Test Cases

|  |  |
| --- | --- |
| Test Number | Purpose of Test |
| 1. | Server SIG INT / Display Time on closing. |
| 2. | Closing server prematurely to simulate a network disconnection or server crash. |
| 3. | Accepting a socket. |
| 4. | Receiving a random number from the server. |
| 5. | Opening the client when the server is turned off. |
| 6. | Receiving the servers uname information. |
| 7. | Uploading a remote file to the server from the client . |
| 8. | Uploading a remote file to the server from the client without the uploads directory existing. |
| 9. | Downloading a file from the server when the uploads directory does not exist. |
| 10. | Browsing the uploads directory from the server when the uploads directory does not exist. |
| 11. | Browsing the uploads directory. |
| 12. | Downloading a file from the server to the cwd. |
| 13. | Valgrind – Client – Searching for any memory leaks. |
| 14. | Valgrind – Server – Searching for any memory leaks. |
| 15. | Menu System – Viewing a set of items where the list is larger then the visible viewport. |

## Testing Data

|  |  |  |  |
| --- | --- | --- | --- |
| Systems Programming Coursework 2, Trimester A | | | |
| Testing Chart | | | |
| Test Number | Command | Expected Result | Actual Result |
| 1. | ./server  *After some time*  C-c | The server to close with the online time being displayed. |  |
| 2. | ./server  ./client  *After some time*  C-c on the server | The client to disconnect instantly. | The client does not disconnect for some reason, I believe this is something to do with the client not freeing up the socket or it’s the system taking some time to free up the socket. Either way the client can read about 10 to 20 bytes before the client closes with this message: |
| 3. | ./server  ./client | The server to accept the socket and the client to open | The server accepts the socket and the client opens. |
| 4. | ./client | The number to be random | The number is randomised although if the requests are send in quick succession because the seeding and time based randomness the number will not be random, its usually around every 500ms to 1s the number is randomised. This is because its not a true random number, it’s a psuedo random number. |
| 5. | ./client | The client should show a error message | A error message is displayed saying the server is offline or it cannot connect. |
| 6. | ./client | The uname information is received and displayed | The information is dispalyed corectly |
| 7. | ./client | The file is uploaded without any errors | The file is uploaded without any errors |
| 8. | ./server | The directory is created automatically and then the file is uploaded. | The directory and file are created. |
| 9. | ./client | A error message is displayed instead of the menu | There are no files so the error is displayed |
| 10. | ./client | A error message is displayed instead of the menu | There are no files so the error is displayed |
| 11. | ./client | The files in the server are displayed | The files in the directory are received and rendered. |
| 12. | ./client | The file is downloaded to the cwd of the client | The file list is displayed and “Makefile” is selected.    The file is then downloaded      The file shows in cwd |
| 13. | valgrind  --leak-check=full  --show-leak-kinds=all  --leak-resolution=med  --track-origins=yes  ./client | There should be no memory leaks associated with the client application. | Upon first running of valgrind I noticed I had 40 memory leaks and had decided to bring a sledge hammer to the source code, upon adding additional arguments to valgrind to see what was going on, I had noticed most of these leaks, infact all of them were caused by ncurses. So I add created a valgrind supression file labelled valgrind-ignore-ncurses.supp  After excluding all ncurses leaks I had none. |
| 14. | valgrind  --leak-check=full  --show-leak-kinds=all  --leak-resolution=med  --track-origins=yes  ./server | There should be no memory leaks associated with the server. | During development I had found 3 memory leaks, most of these were associated with strings that had been allocated (malloc) during the file upload / download process, these were not cleaned up during failed read or writes to the client so the socket was closed but the memory was not released.  Currently the server does not have any leaks. |
| 15. | ./client | The user should be able to scroll them menu even if it exceeds the viewport. This includes viewing labels (non-buttons). | To show this functionality I will be displaying the root folder which exceeds 10 items (which is current the amount of items limited to be rendered):  There are 22 folders and if I scroll I am able to expand the list: |