Michael Baptist
CE156 – Lab 3

# Lab 3 – Reliable UDP – Application Layer Protocol

In figure 1 and 2 below you can find the state diagram for both the server and the client for my lab 3 project covering reliable data transfer using UDP datagrams. The server works as follow. First it creates a listening socket and just waits for incoming connections. Once an incoming connection happens, it creates a new thread using the pthread library and binds a new socket to a new port for just that client interaction. The client sends the filename, number of connections is is attempting to make, and the connection number that is it. With this info the server will know to find the file, find the chunk size and it will know the offset. It sends an ack for each of the above datagrams then starts to send the file and waits for an ack after each sub-chunk is sent. Once all the chunks are sent it exits the thread and goes back to the main thread which is listening, for new connections.

The client is making all the connections and receiving all the data. The client will receive all the data from each server it contacts then piece all the chunks together into one file.

Known issues:

There is a few portability issues with Linux and mac osx. However I have tested the code running on Debian 7 kernel 3.2 and on Ubuntu 3.5 kernel. Also the code runs on the Linux timeshare at the school.

With three servers running the client can contact each to get a chunk of the file. If any of them fail it will contact a known good server to get the remaining chunks. There are some known problems with the way I coded it, I just ran out of time to make sure it was 100% reliable. For example, if the file chunk was partially written it may still count as a good server, however the client should check that each file chunk is the correct size before calling it a good server. There are some small code errors like that. Another known error is that if you call the client with 3 connections but only 1 server in the server-list it will still only get 1 chunk. Once again ran out of time to fix this issue.

Also on Mac Osx there was some strange kernel pthread locks etc. That I didn't have the time to fix up.

Good Note:

I learned pthreads in this lab and reliable UDP. I spent a long time on the project and had a lot of fun doing it.

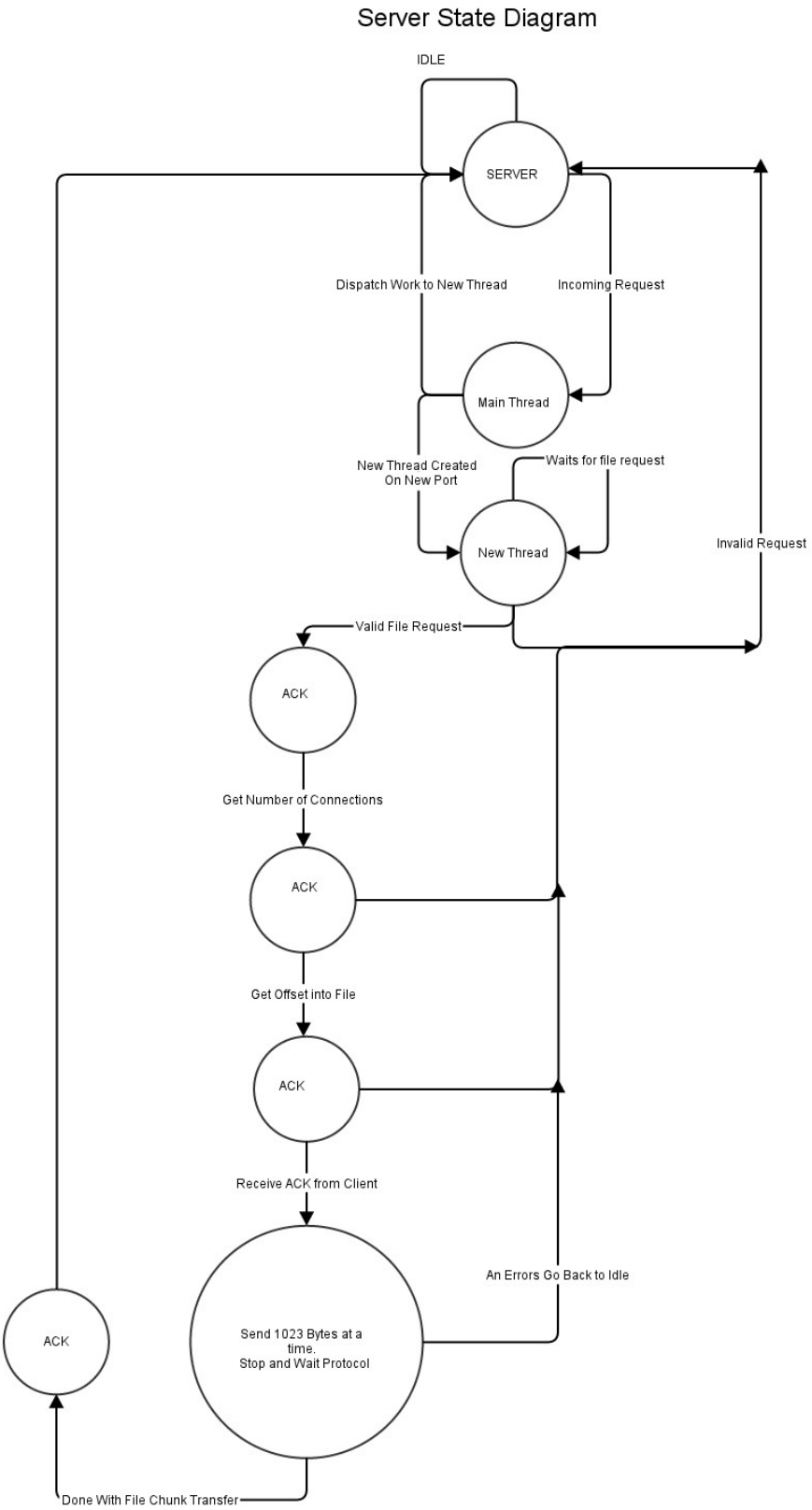


Figure 1. Server State Diagram.

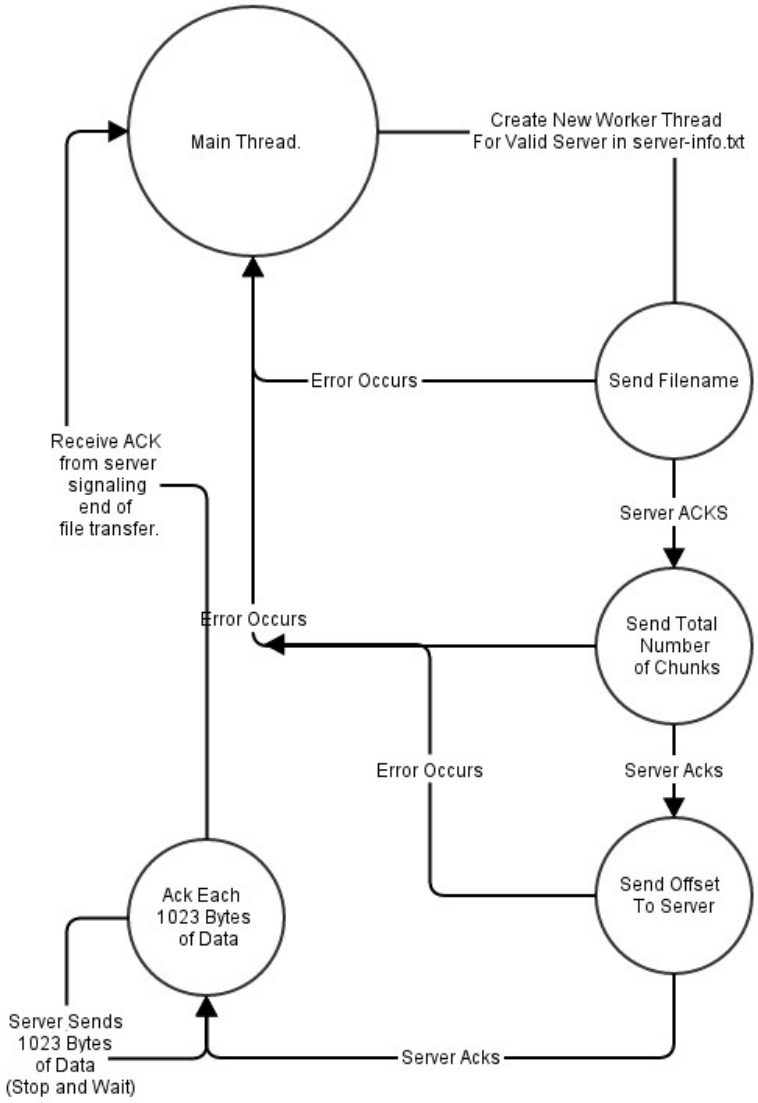


Figure 2. Client State Diagram.