Grayson Hill

Jessica Wei

Britney Khuu

**Group K Project 1 327 Write-Up**

Within the peer-to-peer project, there was a total of three main project files created:

* File Synchronization.cpp
* Server.cpp
* Client.cpp

We made a peer-to-peer file sharing system using asynchronous socket programming with transmission control protocol (TCP). All users would connect to each other using the File Synchronization project. Once connected, the files in their respective “FilesTransferLocation” folder would be read and compared to the others’ “FilesTransferLocation” folder. If there are files that do not exist in certain folders, those folders will receive those files. If there are files with the same name and the contents are different, then the file with the most recent modification date will be kept and overwrite the other files.

How it’s Made

This project was built in C++ with the use of the Boost.Asio library. The server and client cpps were created primarily for testing and learning purposes. Originally they were created to test the principles of client-server connections using the boost.asio library in C++. As knowledge was gained with them code was migrated into File Synchronization for final use as a hybrid client-server.

Challenges

One of the biggest issues was service discovery, most documentation online assumed an ip address had already been found, and thus it took about 8-12 hours of searching before useful code examples were found that allowed the gathering and parsing of local IPs by opening a command prompt pipe and using the “arp -a” command to search the local subnet. As a note, for this to work file/printer sharing must be enabled from the network and sharing center on the control panel and the computers must be on the same vLAN and subnet.

Another issue was navigating the Boost library with little to no descriptive documentation on what each function was used for and what it needed as parameters. This resulted in a whole day of looking up examples and other documentation as a way to understand where and how to use it. The research done to figure it out also helped us to fully comprehend how each function worked and allowed us to take advantage of the library to write code that fit our needs.

An additional difficulty that was encountered during the process of the project was the way how to verify files by comparing them. First was what method would be used in order to compare the files being passed in between computers. At first was the fc command in the command prompt, but it was inefficient at reading files that were coming in from one computer and had to be immediately present within the computer. In the end what was used was a string read of each file and comparing its length to each other in c++. There was also difficulty in gaining a file path of the folder that was used within the project, which was solved with the dir command in command prompt after trial and error.

What We Learned

The biggest thing we learned over the course of this project was primarily how to research, the documentation for boost was relatively arcane for those without highly advanced C++ experience and tutorials were few and far between, most of all *why* things were done was often omitted and had to be pieced together with just context and multiple sources. Additionally with so little documentation on how to do something like port scanning, it took a lot of effort and different search terms which increased our research skills overall.

Additionally we did start to learn how to understand the arcane documentation and it did become easier to use with time, this I think led to a greater understanding of using libraries in C++. Of course we did also learn quite a bit about exactly how a peer to peer network can work and how distributed applications can be structured.

Attempting to structure code for finding files and comparing them was also an experience gained within the course of doing the peer-to-peer project. Exploring different methods and learning of different headers that had their own unique ways in comparing files was interesting to see.