Submission includes:

TCPserver – master server, manages incoming requests from TCPclient, and manages the UDPclients for the micro-services

TCPclient – TCP client that sends the word/sentence to master server and instructions on how to transform it.

6 – UDP servers that represent each micro-service

UDPclient.cpp – class for creating udp clients to communicate to a micro-service

Udp.h – header file for configuration

Run.sh – script to compile everything at once

How to run:

Compiling the Master Server

By script:

```
$ ./run.sh
```

or Manually:

```
g++ UDPclient.cpp IDudpServer.cpp REVudpServer.cpp UPPudpServer.cpp LOWudpServer.cpp
CAEudpServer.cpp SHIFTudpServer.cpp TCPserver.cpp -l pthread -o TCPserver
```

Compiling the client

```
g++ TCPclient.cpp -o TCPclient
```

First run the TCPserver

```
./TCPserver
```

Then run the TCPclient

```
./TCPclient
```

***end server with ctrl+c**

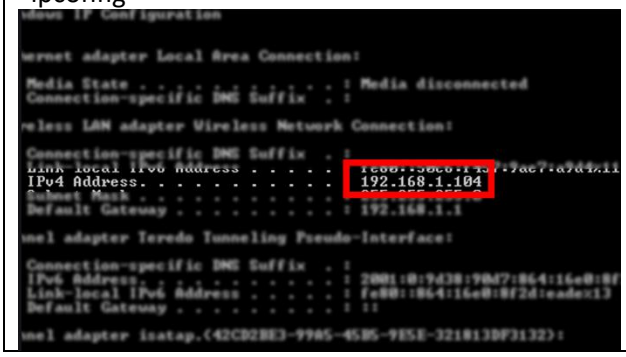
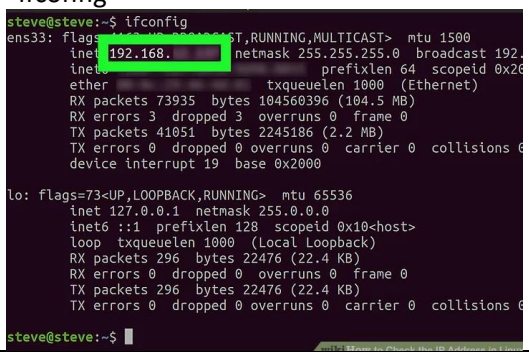
By default, the master server communicates in:

```
#define SERVER_IP "127.0.0.1"
#define MYPORTNUM 30037
```

By setting the ip to localhost, the server and client must communicate in the same device. Feel free to change this.

If you want the client and server to be in different devices, find the ip address of the device in which you are running the server on, and configure it in the header file (**udp.h**)

```
#define SERVER_IP "<device-ip>"
```

Finding your device's ip address: Windows	**Finding your device's ip address: Linux**
"ipconfig" 	"ifconfig" 

Issues: encountered issues have been fixed so there currently is no known issue in this program

****Bonus was done using POSIX Threads and data transformations modifies it word by word, not by sentence. 6-th data transformation is (SHIFTudpServer.cpp) see comments in the file for more info on how it works. Testing was done at home and in school.**