Riley Cooper : cooperra@vt.edu
Sarah Kharimah : sarahkh@vt.edu
Zachary Yee : zjwy99@vt.edu
Anthony Clifton : antclift@vt.edu
ECE 4564, Assignment 1

Section 1: Objectives

The objective of this assignment is to familiarize ourselves with RESTful API, Python parsing, socket connection, checksums, and encapsulation/decapsulation and client/server models. In this assignment, we model a low-level, socket communication between a client and a server, where both the client and the server are interacting with high-level APIs, including Twitter Streaming and WolframAlpha APIs.

With this assignment, we must a stream of twitter posts to look for our specific tag. Once we find that tag, we take the tweet and parse it to extract a question. We then send the question to the server module, which sends it to Wolfram Alpha. We send the response back to the client module, where we post the response back to Twitter.

Section 2: Team Responsibilities

Section 2.1 Parsing

Riley Cooper took the responsibility for creating and checking the tuples for the messages. His portion of the code took responses from the Twitter and Wolfram Alpha APIs, found the hash of those responses, created a tuple that contains the two, and pickled the tuple for transmission. Then his code took the data received on the socket, unpickled it, and checked the hash in the transmitted tuple.

Section 2.2 Twitter API

Sarah Kharimah was responsible for making sure that the client can access the Twitter API to retrieve a question tweeted to and send an answer to the twitter account @netapp_team01. A Python module called `twitteraccess` that contains simplified Twitter interface, such as obtaining OAuth and sending a tweet to timeline, was created. The Python module `twitteraccess` allows for an encapsulated communication. Every request made to the Twitter API was done inside this module, therefore freeing the client from directly accessing the API.

There are two parts to the Twitter API: retrieving a question and sending a tweet. Utilizing tweety, a connection to Twitter Streaming API was established. The client tracks every tweet containing the word "@netapp_team01" real-time, retrieves the text field of the JSON tweet, and stores it in a single text file and passes it to be parsed and sent to WolframAlpha. When an answer by the WolframAlpha is available, the message is then sent to the team's Twitter timeline as a Twitter status update.

Section 2.3 WolframAlpha API

Zachary Yee had the responsibility for making the server access the Wolfram Alpha API to send a question and receive a response. The Wolfram Alpha client received a question in the form of a string. The Wolfram Alpha client then responded to the question and gave back a

Result object. The code then extracted the string answer from that object. This part of the code was placed in the server's infinite loop so that the server can constantly receive questions and output Result objects.

Section 2.4 Socket Connection

Anthony Clifton was responsible for the socket connections between the raspberry pis. As the socket connections were the basis of the project, *client.py* and *server.py* were created for socket developent and would have the other parts of the project added upon completion.

The socket and client pis were set up similarly as they both would have to act as both a client and server for other parts of the project. The client pi acted as a server for the Twitter client and needed to run continuously, it also served as the client in the client-server pi socket connection. The server pi acted as a server to the client pi and as a client to the wolfram server. Both pis were set up to listen for input, unless already busy, from their respective client and then manipulate the received data and send it to the next server.

Once the other parts of the application were added to the client and server, slight adjustments were made so that the parts would interact properly, for example: splitting the responses into multiple tweets if the question had multiple answers.

Section 3: Conclusions

At the completion of this assignment, we gained a better understanding of RESTful API through communicating with Twitter and WolframAlpha APIs, socket connection, client/server and encapsulation model.

When creating the messages to send between the client and server, we learned a little about the various methods python uses to encode arrays of ascii characters. Confusion about which type we needed and which type we had complicated development at first before we figured out where we needed to make conversions to allow the functions to operate properly. For example, the socket send function cannot send a tuple by itself. We tried for testing purposes and learned that's why we needed to pickle the tuple before we sent the message.