ECE 4564 - Group 20

Ethan Brooks
ethanb98@vt.edu

Chris Honaker
chris98@vt.edu

Kishan Parikh
kjp2376@vt.edu

Assignment 1

Forty-Two

In this project, our job is to set up two Raspberry Pi's, one as a server, and one as a client, using the TCP transport protocol. The client uses the Raspberry Pi Camera v2 to scan a QR code, who's question it then encrypts using Fernet, packages using pickle, and sends to the server using the TCP protocol. Once received, the server unpackages the pickle, decrypts the file using the Fernet key provided, and then sends the bitstream file to the IBM Watson API to speak the question out. Once spoken, the question is then sent to the Wolfram Alpha API, which looks up the question and then responds with the answer. This answer is then encrypted by the server using Fernet, packaged using pickle, and then sent to the client. Once the client receives the answer, it unpackages it using pickle, decrypts it with the Fernet key provided, and sends the answer to the IBM Watson API to speak out.

At each of these stages and more, checkpoints are output by both the client and the server, as specified in the grading rubric. Each checkpoint lists the stage the client and server are at, such as whether they are connected, or if the message has been sent or not. The checkpoints acted as good flags of code progress as we ran and decoded the project.

The QR code portion required the libraries from OpenCV and ZBar to operate successfully. Once implemented, the camera took a photo of a QR code, found the four boundary positions based off the inner squares, and decoded the QR code to find the question/URL code contained within. Once found, the rest of the program run as described earlier.

To encrypt the data to be send from one raspberry pi to another, we used Fernet and pickle. We used Fernet to get a key and encrypt the question. We created a tuple to store the key, encrypted question, and the MD5 hash. We pickled the tuple and sent it the other pi. On the other pi, we de-pickled the tuple, and we used the key to decrypt the question. Lastly, we created another tuple with the encrypted answer and MD5 hash. We pickled the tuple and send it to the client.

IBM Watson was used to speak the answer and the question. IBM Watson was sent and spoke the question from the both the server and the client. IBM Watson was accessed through its API and URL which is set in the ServerKeys.py and ClientKeys.py. IBM Watson takes the string to speak, the file format to output and the voice to speak it in. The Watson code creates an mp3 audio file which is then played by pygame.mixer.

ECE 4564 - Group 20

Ethan Brooks
ethanb98@vt.edu

Chris Honaker
chris98@vt.edu

Kishan Parikh
kjp2376@vt.edu

For the WolframAlpha part of the code, we started by getting an access code to access the to the WolframAlpha engine. We used this API code to ask WolframAlpha a question. The answer was sent back in JSON format, so we had to parse this format to get the answer in text format.

Our project was completed and met all the requirements set for it in the rubric. We were able to scan and decode a QR code on the client, encrypt and package the message, send it to the server, have the server unpackage and decrypt the message, speak the question using IBM Watson, ask the question and receive an answer from WolframAlpha, encrypt and package the answer, send it to the client, have the client unpackage and decrypt the answer, and speak the answer using IBM Watson. All checkpoints were output as specified, and the project works as specified.