**3**    **Technical Approach**

The goal of this project was to implement a simple IoT system using the Intel Edison (and various Grove Kit sensors) to unlock a door by sending a username/password string to a server, and receiving back authentication. Initially, the program was achieved using a single light sensor, which was used to encode a binary value to generate a 4-digit password. Moreover, the username was hardcoded into the program. This password was sent as plaintext to the server, and a plaintext response from the server (i.e. a YES or NO) was received by the Edison. Obviously, this simple implementation had serious security flaws such as the short password, and the unencrypted transmission of data to and from the server. This would make the client/server connection susceptible to a number of enemy attacks such as brute-force, dictionary, and rainbow attacks (for the weak password), and man-in-the-middle and replay attacks (due to the lack of encrypted data).

A more secure implementation was developed to combat these security flaws. First of all, the password was increased to 8-digits, and instead of just having a binary value for each of the digits; each digit could have 8 possible values (ranging from 0-7). While still not as strong as it could be, this password length and characteristics served its purpose for this assignment. In addition, instead of a hard-coded username, a user of this IoT system could provide a specific ID string.

The strengthened password helped to protect against several of the attacks mentioned above, but other security measures were taken in the form of a timeout after a certain number of incorrect password attempts took place. After three failed attempts, the server would stop accepting messages from the Edison for a five-minute period. This timeout would help to deter brute-force and dictionary attacks, as an attacker couldn't just enter in an unlimited amount of attempts to crack the password. While the timeout helps in this regard, it is of importance to note that the actual method to input the password (i.e. the use of light sensors and a push button) would already serve to prevent the use of bots to perform the dictionary attacks for an attacker. Lastly, to prevent against man-in-the-middle and replay attacks, the program encrypted the data on the client (and server side) using the Open SSL APIs. In this way, the data would be encrypted going to the server and upon receiving the reply from the server, further strengthening the IoT system.

**4**    **System Design**

This project consists of setting up our Edison as a client and connecting to a server using the standard TCP/IP and Open SSL APIs. Upon running the executable, the user is prompted with a list of instructions on how to enter the username/password string, which is to be sent (encrypted) to the server to request access to unlocking a virtual door. Note that the prompt is relatively lengthy, which is designed so that someone never using the IoT device knows exactly how to generate the username and password. However, for an experienced user of the program, it is relatively easy to enter their information into the system.

The program makes use of four light sensors (each producing a binary value), and button to serve, as a MUX to select which light sensor is active to input the value. The username is a single digit (ranging from 0-7), and the password is 8-digits long, also having 8 possible values (i.e. 0-7). The user will be allotted a designated period of time to press the button to select which light sensor is active before entering the next digit of the password. Moreover, a message will printed to the terminal each time the user presses the button stating which light sensor is currently active.

After the user enters the username/password string, the string is sent over to the server (protected by SSL encryption), which processes the username and password, and sends back an encrypted response (i.e. YES or NO). If authenticated, then the program exits, as the door was unlocked. However, if an incorrect string was sent to the server, and a NO response was received from the server, the program restarts (via a while loop) for the user to try again to enter the correct string. If the user fails to enter the correct string 3 times, then the server initiates a timeout, stops accepting responses from the client, and the Edison is locked out for 5 minutes (after which the user can try to enter the correct string).