A Networked Lock: Simple Implementation

E96A Spring 2017
Due by 11:59pm, May-12-2017

Introduction

Security is one of the most important issues in IoT system design, and authentication is one simple way to improve security. In particular, a password can be used for access approval such that only those knowing the password can gain access to the protected resource. In this project, you will implement a simple password-authenticated IoT system for door opening using Intel Edison and light sensors.

Required Equipment

1. 1x Intel Edison Kit
2. 2x USB 2.0 A-Male to Micro B Cable (micro USB cable)
3. 1x powered USB hub OR an external power supply
4. 1x Grove – Starter Kit for Arduino
5. 1x Personal Computer

Project Description

You are hired by a security company to design a security system for a networked lock that controls the opening/closing of the door of an archive room. Unlike traditional electronic locks which use a digit pad for the user to input their password, the lock here uses a keyboard to enter the user ID and a light sensor to enter the password. The password processing module resides in a server hided somewhere else outside the archive room, for security and power issues. Thus, when a user ID/password combination is entered, it is sent to the server located elsewhere, which will process the request and tell the lock to open the door if the user ID/password combination is correct.

The manager has provided you with a flawed design document made by a USC student last year, which has unfortunately caused many security problems. You are asked to design and implement a more secure system.

To begin with, you decide to implement the system with Intel Edison and the Grove Kit first to see what the limitations are. For simplicity, you will just use a fixed, constant user ID for your device at this moment, and concentrate on handling the password. According to the design specification, the system should convert the sampled light intensity into a “1” or “0” bit to form the password based on a threshold, collects a 4-bit password string in total, sends

the password to the server, and then receives the authentication message from the server indicating whether the door is open or not.

You have done the light intensity threshold assignment in E96A, so you are able to find the threshold based on what you did in the assignment. You have also completed Tutorial 6 (Socket Communication) and Tutorial 4 (in particular, page 14), and you find them useful in implementation this system.

You should write your source code and compile it into a single executable file that can

1. sample and convert the light intensity with into a bit (1/0) every second, generating a 4-bit password string in 4 seconds. You can print proper prompts in the terminal every second and perform corresponding action (covering/uncovering the sensor) after each digit. The password is produced from the most significant bit to the least significant bit. For example, if the light intensity is (4) light, (3) dark, (2) light, (1) dark (0), where the digits in parentheses are the countdowns produced by your program, then the password would be a string: 1010. You can reuse your code from the previous assignment.

2. send **the (constant) user ID and password** to the server. Please see the next section in this assignment for the server specification. **PLEASE STRICTLY FOLLOW THE PROTOCOL** to ensure that the server responds correctly.

3. receive the feedback message from the server (YES/NO) and exit properly

Your program should generate proper prompts in the terminal, which include:
1. proper indication that the program is about to record the password
2. countdowns every second for sampling (4, 3, 2, 1, 0).
3. what your recorded password is
4. proper indication that the program has sent the password to the server
5. the message from the server (YES/NO)
6. appropriate error handling

Server Specification

Address: r01.cs.ucla.edu
Port Number: 16000

The only correct format of the message sent to the server is:

ID = <IDstring> Password = <password string>

which should be terminated by a new line. Please replace <IDstring> with your user ID, and replace <password string> with your coded password. Note that there is space before and after each equal sign "=", and there is a space before the string "Password" and after the <IDstring> as well. It is fine to hard code the user ID you are given for your group into your program, if you want to avoid having to type it in every time.

Each of the group will be provided with a unique combination of user ID and password for testing. Please don't tell anyone else otherwise your system will be in significant danger!

The server will send a message back after it receives a message successfully. The server will send a message containing the string "YES" only if the format of the message is correct and the user ID/password combination is matched. Otherwise, the server will send a message containing the string "NO".

Deliverables

a single zip file (*groupname*.zip) that contains

- the source code
- a compiled binary executable as specified above
- a screenshot showing a successful password match
- a screenshot showing an unsuccessful password match
- a README file that contains your group information (group number, names, UIDs) and explains how to run your code in details

Submission

Please send your zip file per group to zhengyipiz@gmail.com by 11:59pm May-12-2017.