SMART DOOR LOCKING SYSTEM USING RFID AND AWS

J COMPONENT FINAL REPORT

Submitted in the fulfillment for the J Component of Privacy and Security in IoT (BCT3004)

CAL COURSE

in

B.Tech. Computer Science and Engineering

by

Dhruv Singh (19BCT0110)

Jasshu Garg (19BCT0082)

Abhay Sharma (19BCT0125)

Under the guidance of

Dr. Anisha M. Lal

SCOPE

School of Computer Science and Engineering

Winter Semester 2021 - 22

1. ABSTRACT

The objective of this solution is to provide an IoT based secure framework for a smart door locking system which can also be later incorporated into other scenarios. In order to keep our valuable items safe, we use security. We also use various levels of security in our devices to protect the confidential data. Similarly, our houses, offices, hospitals also need security. Therefore, a need for such a secure IoT system arises. Our paper proposes a multi factor security solution is proposed which can authenticate, authorize and validate the user and open the door for the user after two types of verification. All existing Locking systems are localized, they do not send data to anywhere nor can they receive signals etc. Not being on the network, i.e. connected to the internet disables them from providing statistical information. If faults develop in existing systems, they cannot be easily discovered since they are not connected to the network. Our proposed method aims to solve this problem and make the existing model better.

2. INTRODUCTION

This paper proposes a solution to implement an IoT-based Door Lock Security System with RFID and Passcode for 2-Factor Authorization. Also, the ability to submit and see logged data to AWS IoT Core services, allowing for wireless access to the data via the Internet. The objectives of this system are to implement a hardware solution with an Arduino UNO, a NodeMCUWiFi module, an RFID-RC522 chip, and a 4*4 keypad to enable 2-factor authentication to open the door, implement a software solution that makes use of the boards to deliver log data to AWS Core IoT services in real time using the MQTT protocol, and Implement AWS Core IoT Services to log data sent through the NodeMCU module so that it may be utilized for monitoring and analysis afterwards.

The benefits of the proposed system over the existing models are,

- This system supports 2FA (two-factor authentication), which is more secure.
- Through AWS Core IoT Services, monitoring services may be readily supplied using Publisher/Subscriber.
- Over the Air Alerts improve security because the user or higher authorities, such as the police, can be easily notified in the event of a security breach.
- RFID tags do not require any electricity.

In this paper, we try to observe and analyze various algorithms, techniques and models that had been previously proposed through various researches. We try to enhance existing solutions in order to provide more efficient and more reliable solution.

3. LITERATURE SURVEY

IoT (Internet of Things) has a large portion of our life. This is manifested by the large number of connected devices. With the exponential growth of IoT devices, IoT security is becoming important. In particular, Smart Door Lock system is extremely important because it is closely related to the safety of the user. [12]Traditional door locking methods like smart and manual are visible for people looking from outside and there may be chance of theft. [11]Automated door locks using a Bluetooth enabled Android smartphone using an Android smart phone to act as the task master, and a Bluetooth module serving as the command agent. The solenoid output of the door lock serves as the controller/data processing centre, and the controller/data processing centre is an Arduino microcontroller. Depending on the functions required, this technology can be used to monitor, inform, or execute. Instead of a key, it employs a digital command delivered through Bluetooth from a smartphone or other mobile device. Advantages of this proposed method are that it provides Android phone/tablet users with security and convenience. Arduino and Android are used in this project. The implementation is low-cost and simple to use for the general population.[13] The implementation rate is low and affordable to the average individual. Its disadvantages are that, Bluetooth is not a reliable solution. [1] Arduino uno microcontroller allows for design simplicity, hence, tasks can be achieved in lesser time compared to other techniques previously employed. [6] Another model based on Arduino and smartphone is proposed by Khaoula Karimi and MaustaphaKabrane. [8]

Bluetooth and Arduino based model is proposed by Ketan Rathod and Prof. Rambabu vatti. The proposed model is based on embedded system where microcontroller is use for home security. The model uses Arduino as its controller and detects whether the door is unlocked or locked using ultrasonic sensor & LDR values. The Ultrasonic sensor measures the distance of door and LDR detects the intensity of laser light falling on it and based on it decides if its locked or unlocked.The overall cost is low and can be easily operated. [14] But, in order to increase the range and to monitor and control the security from anywhere in the world modules required are costlier. [7]

N. Hashim, N. F. A. M. Azmi, F. Idris and N. Rahim proposea smartphone-based door locking system that uses Wireless Fidelity (Wi-Fi) technology. When the Android programmed smartphone is within WiFi range, it may lock and unlock the door. The Peripheral Interface Controller is the primary controller in the design (PIC). This design has an operating range of 40 to 150 metres. The suggested design is also user-friendly, since it has an interior reset button that allows the user to evacuate the door in the event of an emergency. The door will unlock if the correct password is entered or the exit button is pressed. When the magnetic switch makes contact, the door automatically locks. Users can enter the IP address and port number connected with the WiFi's IP address using the Android application interface. This proposal employs WiFi technology to lock and unlock the door through a smartphone.Advantages of this proposed method are that the device's range is fantastic, and it's also a user-friendly system with no

complexity. Its disadvantages are that because application protocol encryption is not enabled, the effectiveness of security policies suffers. [4]

Image processing and mobile application telegraph technologies are also used in smart door solutions. When a stranger approaches the door, the alarm goes off, and a photograph is taken, which is then sent to the owner. The Wi-Fi module, which is connected to the Arduino Wi-Fi module esp8266, will collect data from the mobile phone through cellular data/wi-fi. The method for authorization is carried out using instructions from an authorized device and a wireless protocol with a cryptographic key. It keeps track of who has signed in and sends out alerts when certain events occur. Advantage of such a system is that, using Raspberry Pi for facial recognition system may result in a smaller, lighter, and more effective system with less control utilization, culminating in a computer-based face identification system. [15] Its disadvantage is that, internet is not available at every place which can be hurdle for this system to be implemented in rural or remote areas. [3]

Location-based smart door lock system as proposed by Trio Adiono, SyifaulFuada ,SinantyaFerantiAnindya, Irfan GaniPurwanda,Maulana Yusuf Fathany, creates a secure door lock system that does not require the user's physical input. The system's main components are a microcontroller and a ZigBee module for connecting with the smart home's host and receiving information about the user's GPS position. It has the capacity to remotely lock and unlock itself. The advantage of such a system is that it has automatic unlocking of door based on user's location. However, this system may be inefficient, inaccurate, and insecure. [2] Another model based on ZigBee is proposed by Yong Tae ParkPraneshSthapit -Jae-Young Pyun. The proposed system is based on a wireless sensor network; it is a low-cost, customizable, and easy-to-install system that does not need precise design, wiring, or construction.The smart digital door lock system communicates in two ways: centralized and emergency. In the event of an emergency, the door lock sends an SMS to the user, reporting the incidence. Once the user has been authenticated through password or RFID tag, the door lock is opened and the LCD displays the status of different household devices. The user can either change the current condition of the appliances or leave them alone. The proposed model, smart digital door lock system can detect three events: a person entering the house, a person leaving the house, and an emergency situation.It is a low-cost, adaptable, and simple-to-install system with an emergency option. This model also gives you control over all of your household equipment. However, because of the large number of devices in the network, the possibilities of being attacked rise, because there may be some device that allows the attacker to attack and interfere with the network. [5][9]

Bio Scanning – finger print security mechanism aims to put a stop to the threat in the event of stealing and fraud. Human fingerprints door lock system is incredibly unique with various key advantages such as non-repudiation, not transferable, not guessable. This methodology proposed makes the system best for security, keyless touch screen entry, budget connectivity, economical. Its drawbacks are that Mobile biometrics and cloud biometrics in finger print door lock system

can be added to prove fingerprint technology as one of the prominent biometric recognition methods. [10]

4. METHODOLOGY

Our proposed methodology is a hardware solution to Smart door locking system. The system implements a two-factor authentication. It validates the UID of the person and upon verification only the person is able to enter the passcode. There will be maximum 3 attempts to fill the passcode correctly after that the UID tag needs to be verified again. If the passcode is correctly entered under 3 attempts the door will open. No action will be taken if there is an invalid UID but a Log is created registering the invalid UID. Led will be used to show whether the UID has been accepted or not (green being UID is valid and has been accepted and red being UID is invalid). Furthermore, buzzer will make noise if the password is incorrectly entered more than 3 times. Cloud technology will be integrated in our project using AWS cloud platform and all the data regarding the system will be logged there.

HARDWARE COMPONENTS USED

- Arduino Uno
- NodeMCU (ESP8266) for Internet Connectivity
- MFRC522 RFID Module
- 4 x 4 Matrix Keypad for password input
- Servo Motor to simulate opening door
- Buzzers and LED to generate alerts

SOFTWARE COMPONENTS USED

- AWS Cloud Platform to log and store data
- Arduino IDE to write and compile the Embedded C code

5. SYSTEM FRAMEWORK, ARCHITECTURE AND PSEUDOCODE

5.1 INITIALIZATIONS

password = "2345";

String mypassword;

counter = 0;

```
attempts = 0;


max_attempts = 3;
rfidpasses = 0;
passcodepassed = 0;
```

5.2MFRC522 RFID MODULE

```
//Initialize RFID
rfidpasses = 0;
myservo.write(0);
content = "";
mypassword = "";
Serial.begin(9600); //set baud rate
SPI.begin; // Serial Peripheral Interface (SPI)
Reset rfid to be ready again
print("Approximate your card to the reader...");

//RFID VALIDITY
validrfid()
{

   if (IsNewCardPresent())
   {
      return 0;
```

```
}
if (no card registered) //Dump info about the card
{
   return 0;
}




for (byte i = 0; i<size; i++)
{
       Read card bits and store to content
}

content. UpperCase();
if (content.substring(1) == stored_card)
{
   print(“Passed the Card Authentication")
   return 1;
}
else
{

   print("Unrecognized Card");
    content = ""
```

```
        return 0;
    }
}
```

5.3 4x4 MATRIX KEYPAD MODULE

```
//Initialize keypad
n_rows = 4;
n_cols = 4;
String content;
char keys[n_rows][n_cols] = {
    {'1', '2', '3', 'A'},
    {'4', '5', '6', 'B'},
    {'7', '8', '9', 'C'},
    {'*', '0', '#', 'D'}};
rowPins[n_rows] = {14, 3, 2, 8};
colPins[n_cols] = {7, 6, 5, 4};
makeKeymap(keys), rowPins, colPins, n_rows, n_cols;
```

5.4 PASSWORD VERIFICATION

```
//Password verification and gate unlock or alarm code
void func()
{
    char key =Keypad.getKey();
    if (key)
    {
```

```
        counter = counter + 1;
        if (attempts > 2)
        {
Serial.print("EMERGENCY! EMERGENCY! EMERGENCY!");
            Start_the_alarm_for_declared_time;
            return;
        }
    }
    if (key == '1')
    {
mypassword = mypassword + 1;
    }
    if (key == '2')
    {
mypassword = mypassword + 2;
    }
    if (key == '3')
    {
mypassword = mypassword + 3;
    }
    if (key == '4')
    {
mypassword = mypassword + 4;
    }
    if (key == '5')
```

```
    {
mypassword = mypassword + 5;
    }
    if (key == '6')
    {
mypassword = mypassword + 6;
    }

    if (key == '7')
    {
mypassword = mypassword + 7;
    }
    if (key == '8')
    {
mypassword = mypassword + 8;
    }
    if (key == '9')
    {
mypassword = mypassword + 9;
    }
    if (key == '0')
    {
mypassword = mypassword + 0;
    }
    if (key == '*')
```

```
    {
        if (password == mypassword)
        {
            print(" got entry ");
run_servo_motor; //open gate
            reset;
        }
        else
        {
            print(" filled wrong password ");
            attempts = attempts + 1;
func();
        }
    }
}
```

5.5 NodeMCU MODULE

```
set id and password for wifi
AWS_endpoint = "a3apchnx1hpbj0-ats.iot.us-west-2.amazonaws.com";  //MQTT broker ip
Set MQTT port; //set MQTT port number to 8883 as per standard


void callback()
{
Print (received message)
}
```

```
void setup_wifi()
{
   // We start by connecting to a WiFi network
WiFi.begin(ssid, password);
   print ("WiFi connected- wifiname and ip");

   while (!timeClient.update())
   {
      Try for connection
   }
}

void reconnect()
{
   // Loop until we're reconnected
   while (!client.connected())
   {
         print("Attempting MQTT connection...");
      // Attempt to connect
      if (client.connect("ESPthing"))
      {
         println("connected");
        // Once connected, publish an announcement and resubscribe
client.subscribe("inTopic");
```

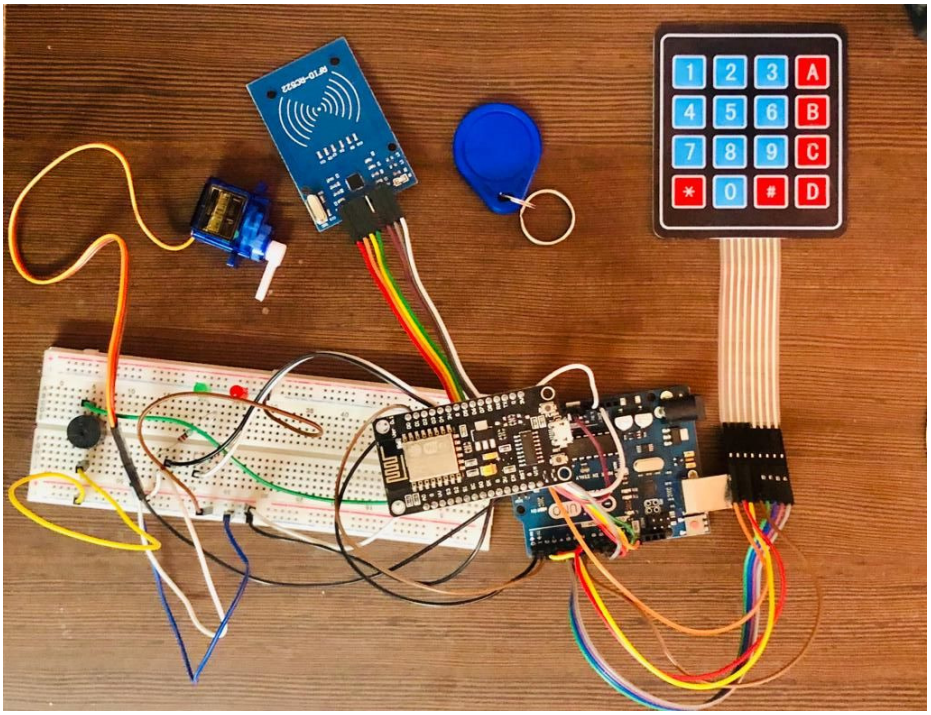
```
      }
      else
     {
          (" try again in 5 seconds");
      }
   }
}

void setup()
{
setup_wifi();
   if (!SPIFFS.begin())
   {
println("Failed to mount file system");
      return;
   }

   File cert = Load certificate file

   File private_key =Load private key file

   File ca = load ca file;

void loop()
{
```

```
 String msg1 = mySerial.readStringUntil('\r');

Serial.println(msg1);

   if (!client.connected()) {

     reconnect();

     }

client.loop();


print("Publish message: ");

println(msg1);

{

client.publish(msg);

}

ESP.getFreeHeap(); //Clear heap as low heap can cause problems

}
```

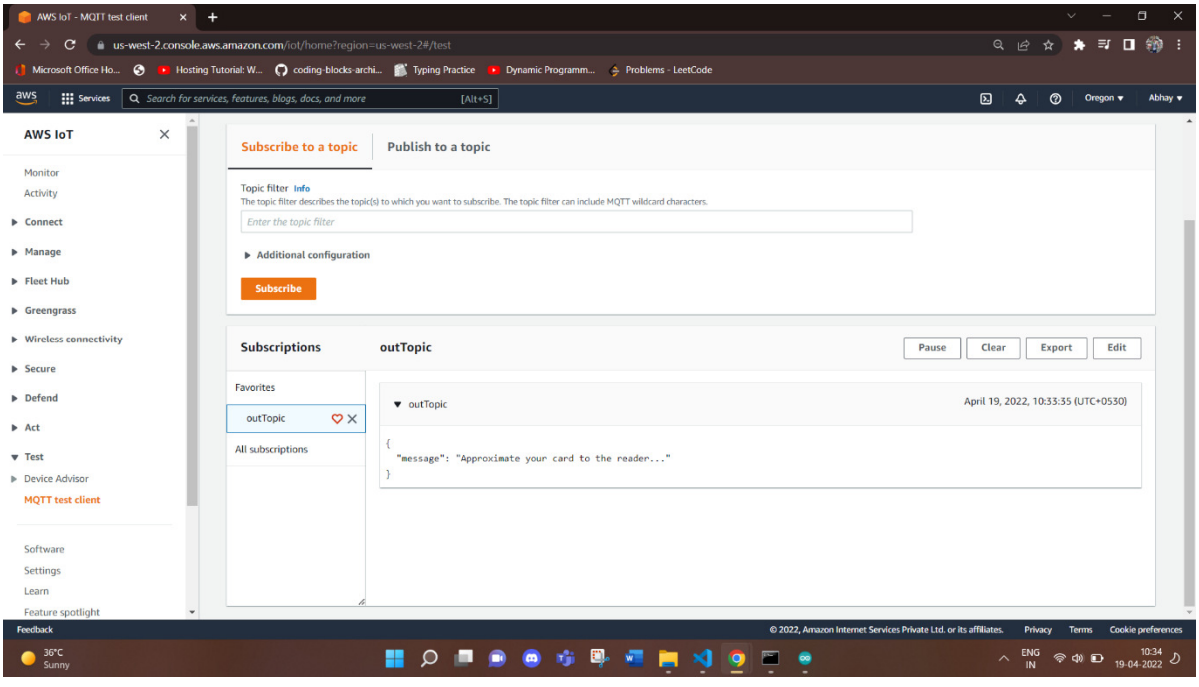
5.6 PROPOSED ARCHITECTURE

- The security system contains a door locking system using a passive type of RFID.
- In order to increase the security, passcode is used for two-factor Authentication.
- The hardware components are Processing Board, RFID reader & tags, keypad, electric door lock, USB connections and connecting cables etc.
- Few Actuators like LED, Servo Motor, Buzzer can also be used to enhance User-Interface
- Each Log will be stored on Amazon Web Services using MQTT services.

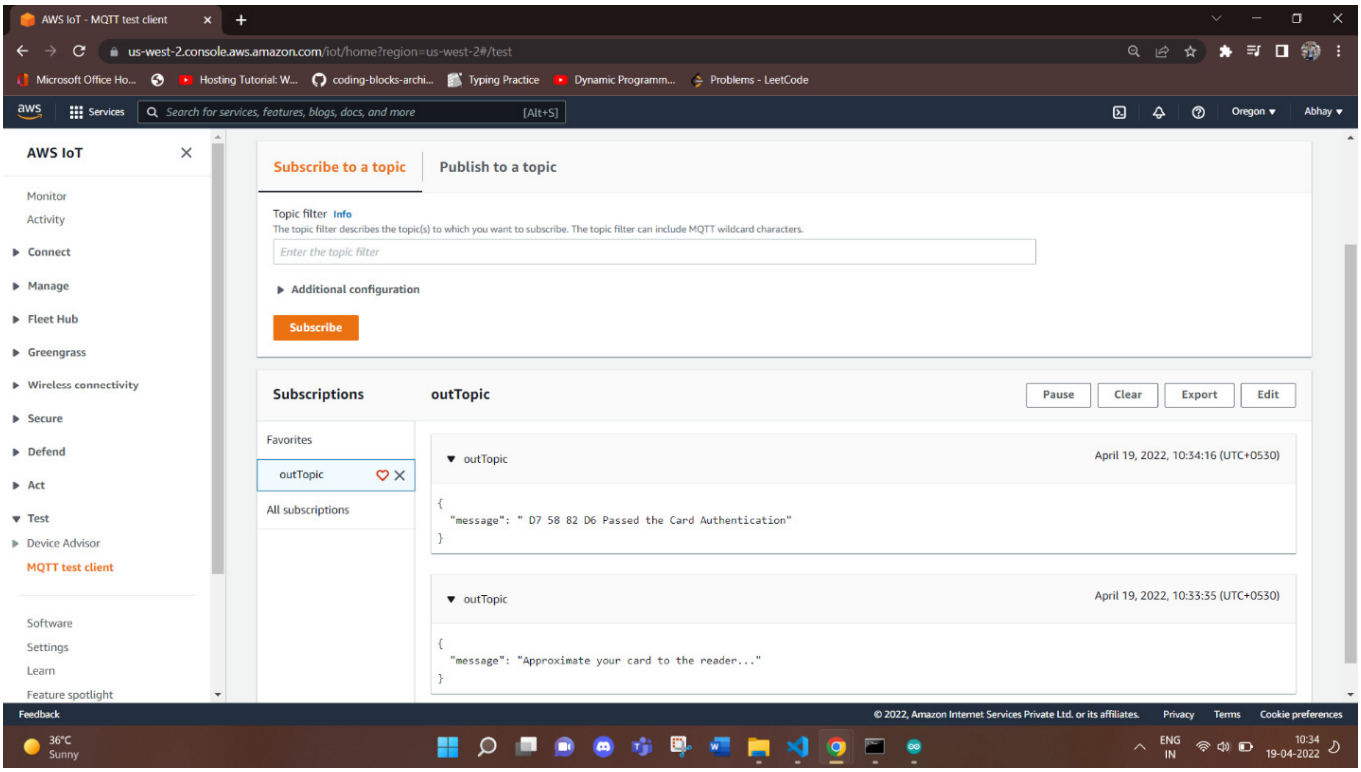
5.7 CIRCUIT CONNECTION



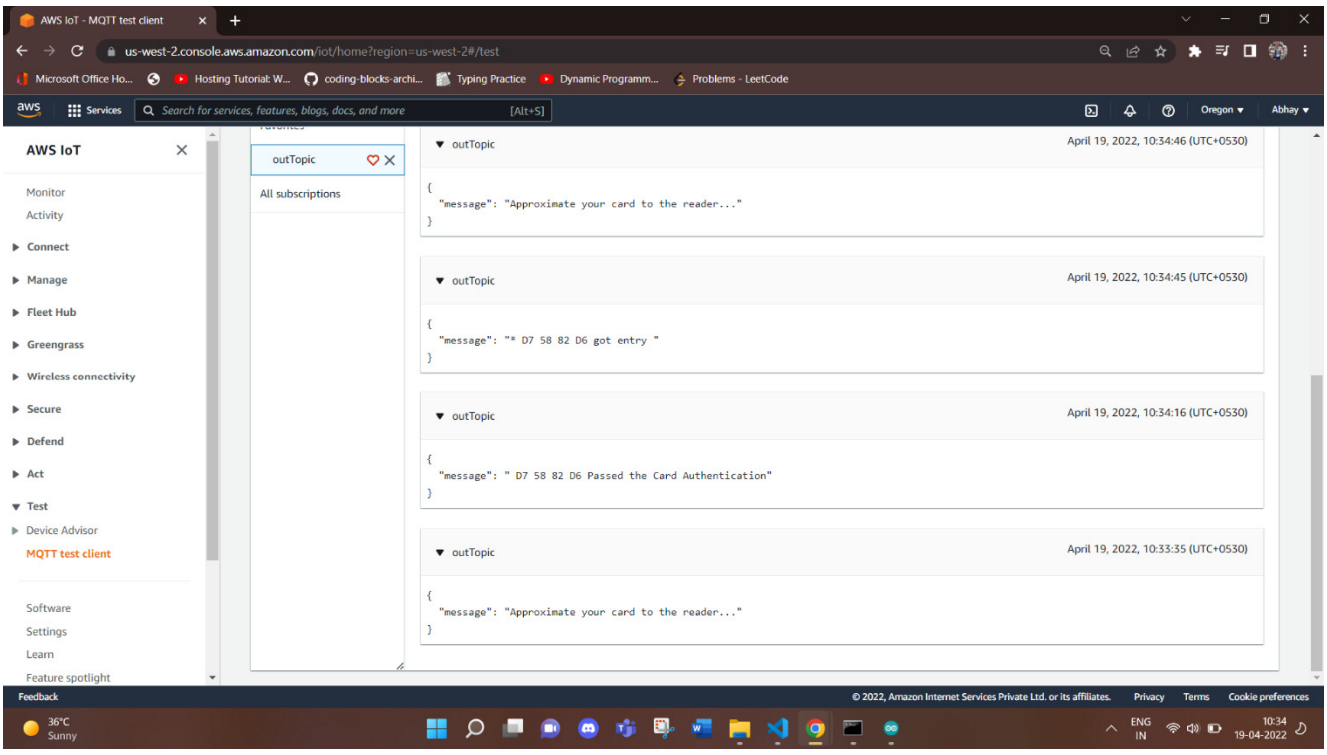
6. RESULTS AND DISCUSSION



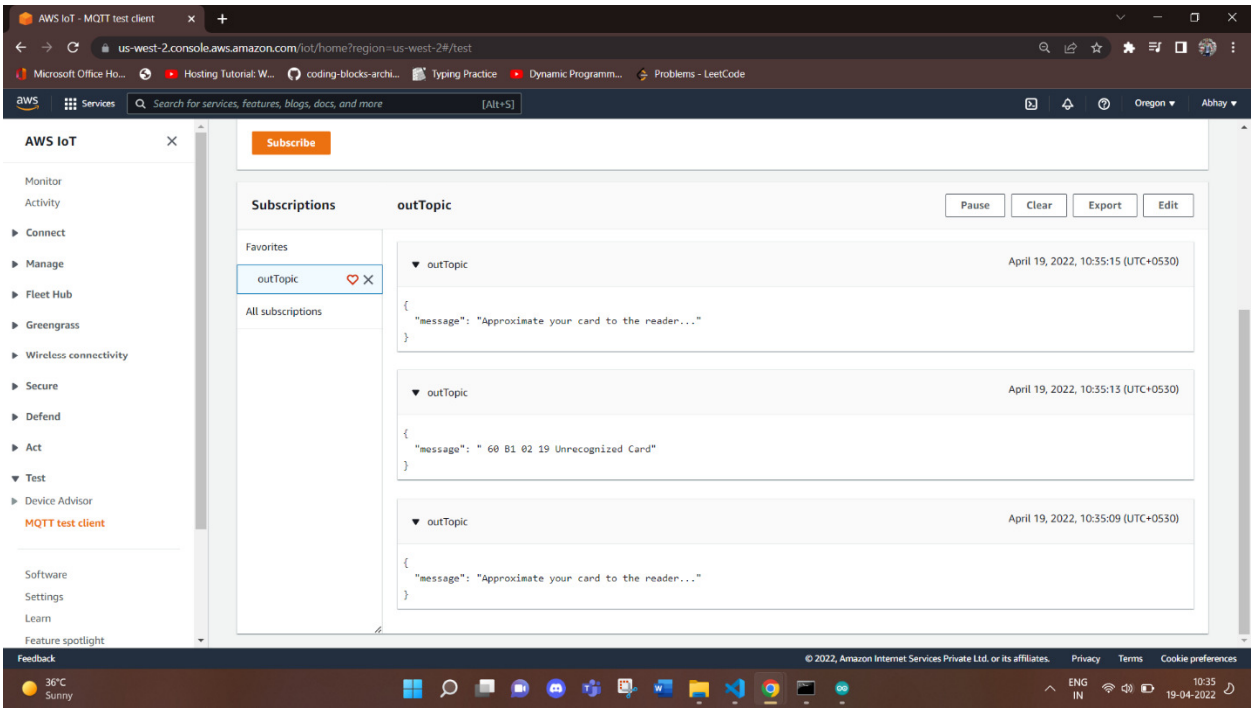
Verify the RFID card



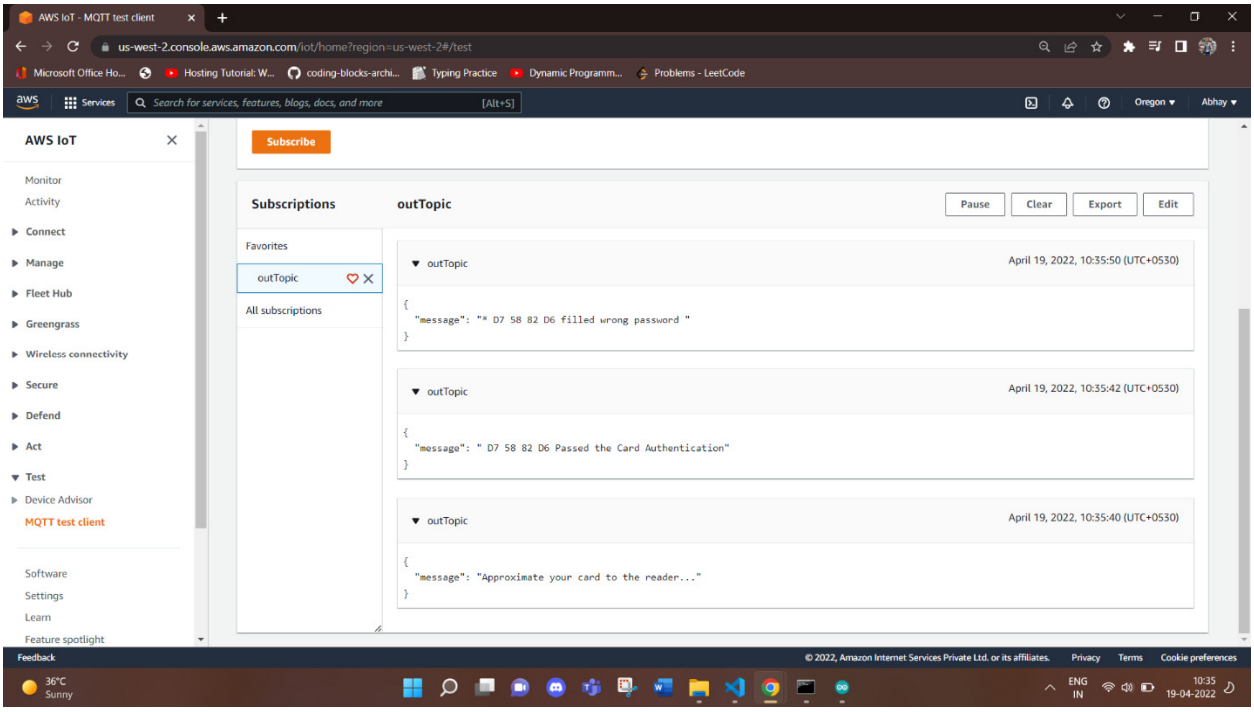
Successful authentication



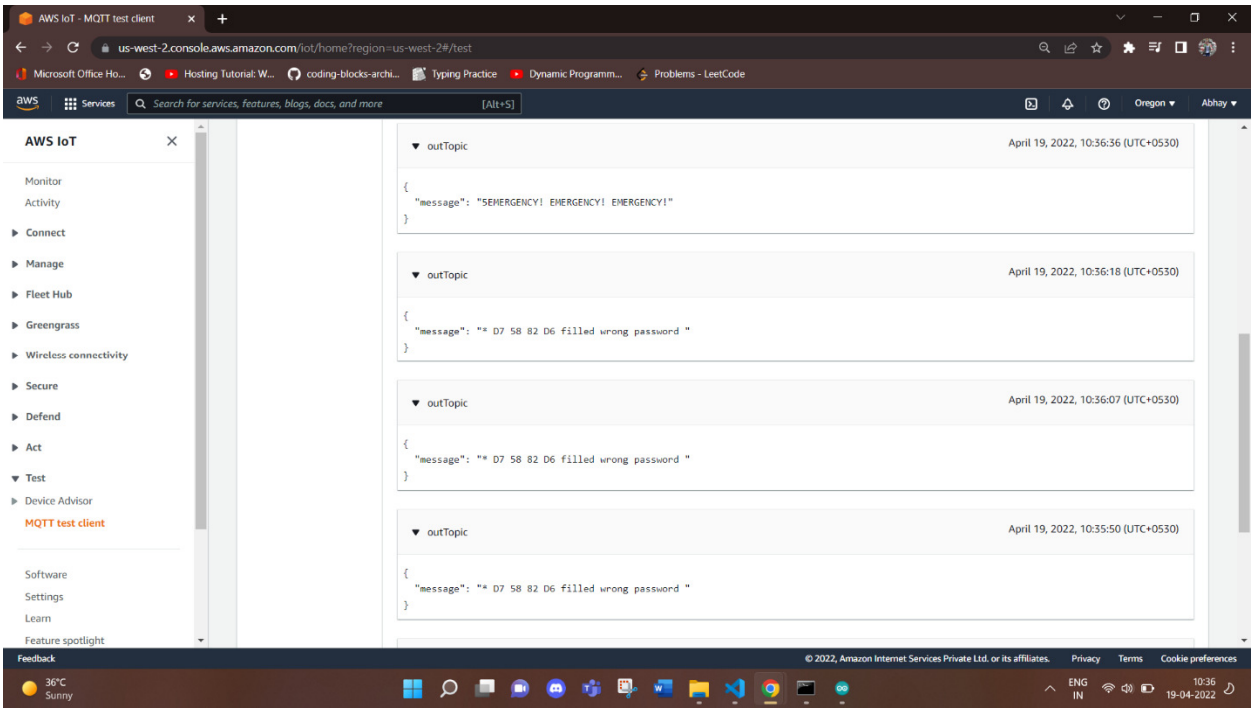
Entered correct password, Got entry.



Unrecognized card



Entered wrong password



Filled wrong password thrice, emergency alarm ringing

7. CONCLUSION

The problem has been formulated and various studies have been reviewed regarding the different electronic lock systems. RFID technology is one of the successfully incorporated in the door lock system. A security mechanism with RFID, Passcode provides two-factor authentication. Logs can be monitored from anywhere in world with internet. The generated logs can also be used for analysis and stored for future reference. Finally, a multi-level authentication digital door access control system can be used in critical zones like Military Institute, Defense, Scientific Laboratory, etc. too, which is cost effective and increase the scale of security successfully. The proposed hardware solution prevents attackers to gain access via password cracking attacks with the help of RFID.

For future work on the system, a camera/fingerprint module can be used to increase security Data Generated by AWS IoT Core can be used as an API or can be used to create some specific Web Application or Mobile Application. Other Modules like GPS, GPRS can be included which will extract the Location from user's Mobile and unlock the door when user is at some specific distance. In the case of multiple users or identities with different passcode and/or RFIDs blockchain can be implemented to store the identities securely.

8. REFERENCES

[1] Door-automation system using Bluetooth-based android for mobile phone - Lia Kamelia, Alfin Noorhassan S.R, Mada Sanjaya and W.S., Edi Mulyana

[2] IoT-Enabled Door Lock System - Trio Adiono, SyifaulFuada ,SinantyaFerantiAnindya, Irfan GaniPurwanda,Maulana Yusuf Fathany

[3] Smart door system - SiddheshRaorane, Abhishek Hanchate, Umar Gigani, Prof. Surekha Khot

[4] SMARTPHONE ACTIVATED DOOR LOCK USING WIFI - N. Hashim1, N. F. A. M. Azmi1, F. Idris2 and N. Rahim1

[5] Smart Digital Door Lock for the Home Automation (ZigBee) - Yong Tae ParkPraneshSthapit -Jae-Young Pyun

[6] Automated Door Lock System Using Arduino -Chinedu Reginald Okpara - Ononiwu, Gordon Chigozie

[7] Smart door security using Arduino and Bluetooth Application- Ketan Rathod - Prof. Rambabu vatti

[8] Secure Smart Door Lock System based on Arduino and Smartphone App - Khaoula Karimi - MaustaphaKabrane

[9] RFID-Based Digital Door Locking System - Shubham Soni, Rajni Soni, Akhilesh A. Waoo

[10] Unique Authentication for Door Lock system through Bio Scanning-Finger Print Security System -Dilshad Mahjabeen, Moshiur Rahman Tarafder

[11] A Novel Door Lock Operation Using Two Staged Smart Security Verification - Durga K Prasad Gudavalli; I Swetha Monica; M. E. C. Vidya Sagar

[12] Blockchain based smart door lock system - D. Han, H. Kim and J. Jang

[13] IoT Enhanced Smart Door Locking System - M Shanthini;G Vidya;R Arun

[14] Smart Door Locking Mechanism - Janhavi Baikerikar;Vaishali Kavathekar;Nilesh Ghavate;Ronit Sawant;Kharanshu Madan

[15] Smart Door Locking System - D Aswini;R Rohindh;K S Manoj Ragavendhara;C S Mridula

Appendix

Code for Arduino Uno:

```
#include <SPI.h>
#include <MFRC522.h>
#include <Keypad.h>
#include <Servo.h>
Servo myservo;

//green A2
// red A3

const byte n_rows = 4;
const byte n_cols = 4;
String content;
char keys[n_rows][n_cols] = {
   {'1', '2', '3', 'A'},
   {'4', '5', '6', 'B'},
   {'7', '8', '9', 'C'},
   {'*', '0', '#', 'D'}};

byte rowPins[n_rows] = {14, 3, 2, 8};
byte colPins[n_cols] = {7, 6, 5, 4};

Keypad myKeypad = Keypad(makeKeymap(keys), rowPins, colPins, n_rows,
n_cols);
String password = "2345";
String mypassword;
int counter = 0;
int attempts = 0;

int max_attempts = 3;
int rfidpasses = 0;
int passcodepassed = 0;
#define SS_PIN 10
#define RST_PIN 9
MFRC522 mfrc522(SS_PIN, RST_PIN);
```

```
void setup()
{
pinMode(A2,OUTPUT); // green
pinMode(A3,OUTPUT); //red
myservo.attach(A1);
pinMode(A4, OUTPUT);
digitalWrite(A2,LOW);
digitalWrite(A3,LOW);
rfidpasses = 0;
myservo.write(0);
   content = "";
mypassword = "";
Serial.begin(9600);
SPI.begin();
   mfrc522.PCD_Init();
Serial.print("Approximate your card to the reader...");


   for(int i=0;i<5;i++)
   {
digitalWrite(A3,HIGH);
digitalWrite(A2,LOW);
      delay(100);
digitalWrite(A3,LOW);
digitalWrite(A2,HIGH);
      delay(100);
   }
digitalWrite(A2,LOW);
digitalWrite(A3,LOW);

}

void loop()
{
// tone(A4, 2500, 19000);
// delay(10);
```

```
// tone(A4, 1500, 15000);
//digitalWrite(A4,LOW);
//digitalWrite(A3,HIGH);
//myservo.write(90);
//delay (500);
//myservo.write(0);
//delay (500);


   if (rfidpasses == 1)
   {
func();
   }
   else
   {
rfidpasses = validrfid();
   }
   delay(500);
}

int validrfid()
{

   if (!mfrc522.PICC_IsNewCardPresent())
   {
      return 0;
   }
   if (!mfrc522.PICC_ReadCardSerial())
   {

      return 0;
   }


   byte letter;
   for (byte i = 0; i< mfrc522.uid.size; i++)
   {
```

```
//      Serial.print(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " ");
//      Serial.print(mfrc522.uid.uidByte[i], HEX);
content.concat(String(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " "));
content.concat(String(mfrc522.uid.uidByte[i], HEX));
   }

content.toUpperCase();
   if (content.substring(1) == "D7 58 82 D6")
   {
Serial.print(content +" Passed the Card Authentication");
digitalWrite(A2,HIGH);
      delay(1000);
digitalWrite(A2,LOW);
      return 1;
   }
   else
   {

Serial.print(content +" Unrecognized Card");
       content = "";
digitalWrite(A3,HIGH);
      delay(1000);
digitalWrite(A3,LOW);
      setup();
      return 0;
   }
}

void func()
{

   char key = myKeypad.getKey();
   if (key)
   {
Serial.print(key);
      counter = counter + 1;
      if (attempts > 2)
```

```
      {
Serial.print("EMERGENCY! EMERGENCY! EMERGENCY!");

         for(int i=0;i<20;i++)
         {
digitalWrite(A3,HIGH);
           tone(A4, 2500, 19000);
           delay(100);
           tone(A4, 1500, 15000);
digitalWrite(A3,LOW);
           delay(100);

         }

         attempts = 0;
//          delay(1000);
         setup();
         return;
      }
   }

   if (key == '1')
   {
mypassword = mypassword + 1;
   }

   if (key == '2')
   {
mypassword = mypassword + 2;
   }

   if (key == '3')
   {
mypassword = mypassword + 3;
   }

   if (key == '4')
```

```
   {
mypassword = mypassword + 4;
   }

   if (key == '5')
   {
mypassword = mypassword + 5;
   }

   if (key == '6')
   {
mypassword = mypassword + 6;
   }

   if (key == '7')
   {
mypassword = mypassword + 7;
   }

   if (key == '8')
   {
mypassword = mypassword + 8;
   }

   if (key == '9')
   {
mypassword = mypassword + 9;
   }

   if (key == '0')
   {
mypassword = mypassword + 0;
   }

   if (key == '*')

   {
```

```
//       Serial.println(mypassword);

      if (password == mypassword)

      {

Serial.print(""+content + " got entry ");
digitalWrite(A2,HIGH);

myservo.write(90);
          delay(2000);
myservo.write(0);



          content = "";
mypassword = "";
rfidpasses = 0;
passcodepassed = 0;
          counter = 0;
          setup();
      }
      else
      {
Serial.print( ""+content + " filled wrong password ");
mypassword = "";
digitalWrite(A3,HIGH);
          delay(100);
digitalWrite(A3,LOW);
          attempts = attempts + 1;
func();
      }
   }
}
```

Code for NodeMCU:

```
#include "FS.h"
#include "ESP8266WiFi.h "
#include "PubSubClient.h "
#include "NTPClient.h "
#include "WiFiUdp.h "
#include <SoftwareSerial.h>
SoftwareSerialmySerial(D1,D2);
//// Update these with values suitable for your network.
//
const char *ssid = "Garg";
const char *password = "1@333_55555";

WiFiUDPntpUDP;
NTPClienttimeClient(ntpUDP, "pool.ntp.org");
//
const char *AWS_endpoint = "a1d7zgjv8pytti-ats.iot.ap-south-
1.amazonaws.com"; //MQTT broker ip

void callback(char *topic, byte *payload, unsigned int length)
{
Serial.print("Message arrived [");
Serial.print(topic);
Serial.print("] ");
   for (int i = 0; i< length; i++)
   {
Serial.print((char)payload[i]);
   }
Serial.println();
}
WiFiClientSecureespClient;
PubSubClient client(AWS_endpoint, 8883, callback, espClient); //set MQTT
port number to 8883 as per //standard
long lastMsg = 0;
char msg[50];
```

```
int value = 0;

void setup_wifi()
{

   delay(10);
   // We start by connecting to a WiFi network
espClient.setBufferSizes(512, 512);
Serial.println();
Serial.print("Connecting to ");
Serial.println(ssid);

WiFi.begin(ssid, password);

   while (WiFi.status() != WL_CONNECTED)
   {
      delay(500);
Serial.print(".");
   }

Serial.println("");
Serial.println("WiFi connected");
Serial.println("IP address: ");
Serial.println(WiFi.localIP());

timeClient.begin();
   while (!timeClient.update())
   {
timeClient.forceUpdate();
   }

   espClient.setX509Time(timeClient.getEpochTime());
}

void reconnect()
{
   // Loop until we're reconnected
```

```
  while (!client.connected())
  {
Serial.print("Attempting MQTT connection...");
    // Attempt to connect
    if (client.connect("ESPthing"))
    {
Serial.println("connected");
      // Once connected, publish an announcement...
client.publish("ESP8266_Smart_door_log", "hello world");
      // ... and resubscribe
client.subscribe("inTopic");
    }
    else
    {
Serial.print("failed, rc=");
Serial.print(client.state());
Serial.println(" try again in 5 seconds");

      char buf[256];
espClient.getLastSSLError(buf, 256);
Serial.print("WiFiClientSecure SSL error: ");
Serial.println(buf);

      // Wait 5 seconds before retrying
      delay(5000);
    }
  }
}

void setup()
{

Serial.begin(115200);
mySerial.begin(9600);
Serial.setDebugOutput(true);
  // initialize digital pin LED_BUILTIN as an output.
pinMode(LED_BUILTIN, OUTPUT);
```

```
setup_wifi();
   delay(1000);
   if (!SPIFFS.begin())
   {
Serial.println("Failed to mount file system");
      return;
   }

Serial.print("Heap: ");
Serial.println(ESP.getFreeHeap());

   // Load certificate file
   File cert = SPIFFS.open("/cert.der", "r"); //replace cert.crt eith your
uploaded file name
   if (!cert)
   {
Serial.println("Failed to open cert file");
   }
   else
Serial.println("Success to open cert file");

   delay(1000);

   if (espClient.loadCertificate(cert))
Serial.println("cert loaded");
   else
Serial.println("cert not loaded");

   // Load private key file
   File private_key = SPIFFS.open("/private.der", "r"); //replace private eith
your uploaded file name
   if (!private_key)
   {
Serial.println("Failed to open private cert file");
   }
   else
Serial.println("Success to open private cert file");
```

```
   delay(1000);

   if (espClient.loadPrivateKey(private_key))
Serial.println("private key loaded");
   else
Serial.println("private key not loaded");

   // Load CA file
   File ca = SPIFFS.open("/ca.der", "r"); //replace ca eith your uploaded file
name
   if (!ca)
   {
Serial.println("Failed to open ca ");
   }
   else
Serial.println("Success to open ca");

   delay(1000);

   if (espClient.loadCACert(ca))
Serial.println("ca loaded");
   else
Serial.println("ca failed");

Serial.print("Heap: ");
Serial.println(ESP.getFreeHeap());
}

void loop()
{

 String msg1 = mySerial.readStringUntil('\r');
Serial.println(msg1);
   if (!client.connected()) {
     reconnect();
     }
```

```
client.loop();


Serial.print("Publish message: ");

snprintf (msg, 75, "{\"message\": \"%s\"}", msg1.c_str());
Serial.println(msg1);
if(msg1!="")
{
client.publish("outTopic", msg);
}

Serial.print("Heap: ");
Serial.println(ESP.getFreeHeap()); //Low heap can cause problems
// wait for a second

digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage
level)
delay(100); // wait for a second
digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the voltage
LOW
delay(100); // wait for a second
}
```