* * * * *

Smart home IoT

Introducere

Smart home IoT este un proiect ce isi propune sa demonstreze usurinta cu care deviceurile din interiorul unei case pot comunica cu o aplicatie remote. Proiectul isi propune sa implementeze urmatoarele demo-uri:

1. Preventia unei scurgeri de gaze. Cu ajutorul unui senzor de gase conectat la un ESP32, bucataria ar trebui sa transmita in timp real nivelul de gaze resimtit de catre senzor in ambient.
2. Preventia unei scurgeri de apa. Cu ajutorul unui senzor pentru adancime conectat la un ESP32, vom masura daca in baie nivelul apei este peste un prag stabilit.
3. Simularea controlului luminilor pentru mai multe zone. Vom folosi o matrice led conectata a un ESP32 pentru a simula comenzi primite de la remote server.

Descriere generala

SCHEMA BLOC:

Comunicare collector <-> gateway

Pentru a asigura o comunicare eficientă și fiabilă între dispozitivele din rețeaua noastră Smart Home IoT, utilizăm interfața UART (Universal Asynchronous Receiver-Transmitter). Această interfață permite transmiterea și recepționarea datelor seriale între colectorul de telemetrie și gateway.

Prin intermediul conexiunii UART, colectorul de telemetrie trimite datele colectate din senzorii din interiorul casei către gateway. Aceste date pot include informații despre nivelul de gaze sau apă detectat în diverse zone ale casei, precum și starea luminilor în diferite camere.

Gateway-ul primește apoi aceste date și le procesează în consecință. De exemplu, poate iniția acțiuni precum emiterea unor avertismente în cazul unei scurgeri de gaze sau apă, sau poate controla iluminatul în funcție de preferințele utilizatorului.

Prin intermediul interfeței UART, comunicarea între colectorul de telemetrie și gateway este rapidă și robustă, asigurând astfel funcționarea corectă a sistemului Smart Home IoT.

Hardware Design

Nod baie

Nod bucătărie

Nod living

Camera gateway

Software Design

Tool-uri folosite: Arduino Studio, ESP NOW, Arduino Cloud

Flow date

Din punct de vedere software, o problemă reală ce își caută soluții este securizarea și partitionarea resurselor hardware. Vrem să comunicăm cu exteriorul, toată lumea vorbește despre IoT, dar cât de sigur este?

O casă smart este un lucru pe care probabil toți ni-l dorim. Însă, o conexiune la Wi-Fi nu înseamnă întotdeauna doar confort, poate însemna în cazuri particulare pericol. Suntem expuși în exterior și putem fi ținta atacurilor.

Software-ul reprezentat în IoT Smart Home este partitionat astfel încât să reducă riscul cât mai mult. Plecând de la premisa că trebuie să minimizăm riscul eventualelor atacuri asupra sistemului, vrem ca tot flow-ul de date să nu poată fi controlat din exterior. Considerăm cele 4 noduri: (Bucătărie, Baie, Gateway, Living). Având în vedere partitionarea folosită de noi, vrem ca toate datele să fie direcționate către gateway.

ESP32 to ESP32

Pentru a realiza acest lucru folosim un telemetry collector. Telemetry collectorul nostru comunică cu toate nodurile prin ESP-NOW, un protocol lightweight definit de ESP.

```
#include <esp_now.h>
#include <WiFi.h>

// Structura pentru datele primite
typedef struct struct_message {
```

```
    char a[32];
    int b;
    float c;
    bool d;
} struct_message;

// Instanța pentru a stoca datele primite
struct_message incomingReadings;

// Callback pentru a primi date
void OnDataRecv(const uint8_t * mac, const uint8_t *incomingData, int len) {
  memcpy(&incomingReadings, incomingData, sizeof(incomingReadings));
  Serial.print("Bytes received: ");
  Serial.println(len);
  Serial.print("Char: ");
  Serial.println(incomingReadings.a);
  Serial.print("Int: ");
  Serial.println(incomingReadings.b);
  Serial.print("Float: ");
  Serial.println(incomingReadings.c);
  Serial.print("Bool: ");
  Serial.println(incomingReadings.d);
}

void setup() {
  // Inițializează serial monitorul
  Serial.begin(115200);

  // Inițializează WiFi
  WiFi.mode(WIFI_STA);
  Serial.println("ESP-NOW Initialization...");

  // Inițializează ESP-NOW
  if (esp_now_init() != ESP_OK) {
    Serial.println("Error initializing ESP-NOW");
    return;
  }
  Serial.println("ESP-NOW Initialized");

  // Înregistrează callback pentru a primi date
  esp_now_register_recv_cb(OnDataRecv);
}

void loop() {
  // Lăsați gol. Datele sunt primite în callback.
}
```

Telemetry collector Acest nod deține toată informația din sistem. Senzorul de gaz transmite către el, senzorul de nivel apă transmite către el, iar matricea de LED-uri este controlată de el. Tocmai pentru acest lucru, informația stocată este transmisă prin UART către gateway, utilizând un protocol închis și bine definit. Prelucrarea datelor se face pe acest nod, collector, deoarece vrem să trimitem date către

gateway cu o însemnătate diferită de cele din sistem, astfel un posibil atacator nu poate să modifice starea sistemului determinist.

```
#include <HardwareSerial.h>

HardwareSerial mySerial(1);

// Structură pentru datele de trimis
typedef struct struct_message {
    char a[32];
    int b;
    float c;
    bool d;
} struct_message;

// Instanța pentru datele de trimis
struct_message outgoingReadings = {"Gas Level", 25, 3.14, true};

void setup() {
  // Inițializează serial monitorul
  Serial.begin(115200);

  // Configurarea UART
  mySerial.begin(9600, SERIAL_8N1, 16, 17);  // TX = GPIO16, RX = GPIO17
  Serial.println("UART Initialized");

  // Configurare registrii UART
  WRITE_PERI_REG(UART_CONF0_REG(1), READ_PERI_REG(UART_CONF0_REG(1)) &
~UART_TICK_REF_ALWAYS_ON);
  WRITE_PERI_REG(UART_CONF1_REG(1), (READ_PERI_REG(UART_CONF1_REG(1)) &
~UART_RX_TOUT_THRHD_V) | (20 & UART_RX_TOUT_THRHD_V));
  SET_PERI_REG_MASK(UART_CONF1_REG(1), UART_RX_TOUT_EN);
}

void loop() {
  // Trimiterea datelor prin UART
  mySerial.write((uint8_t*)&outgoingReadings, sizeof(outgoingReadings));
  Serial.println("Data sent via UART");

  delay(2000);  // Trimite date la fiecare 2 secunde
}
```

Gateway Singurul nod conectat cu cloud-ul, care are certificatele de conexiune și care comunică cu exteriorul. Acesta este expus la internet, dar însemnătatea lui este minimizată, deoarece collectorul nu poate fi alterat de datele care vin de la gateway.

```
#include <WiFi.h>
#include <ArduinoCloud.h>
#include <Arduino_ConnectionHandler.h>

// Definește credențialele Wi-Fi
```

```
const char* ssid = "numele_retelei";
const char* password = "parola_retelei";

// Definește variabilele pentru conectare la Arduino IoT Cloud
const char* deviceId = "id_dispozitiv";
const char* thingId = "id_thing";
const char* secret = "secret_dispozitiv";

// Callback pentru schimbări de conectivitate
void onCloudEvent(CloudEvent event) {
  switch (event) {
    case CONNECTED:
      Serial.println("Connected to Arduino IoT Cloud");
      break;
    case DISCONNECTED:
      Serial.println("Disconnected from Arduino IoT Cloud");
      break;
  }
}

void setup() {
  // Inițializează serial monitorul
  Serial.begin(115200);

  // Conectare la Wi-Fi
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Connecting to WiFi...");
  }
  Serial.println("Connected to WiFi");

  // Inițializează conexiunea la Arduino IoT Cloud
  ArduinoCloud.begin(ArduinoIoTPreferredConnection);
  ArduinoCloud.addCallback(ArduinoIoTPreferredConnectionEvent::CONNECT,
onCloudEvent);
  ArduinoCloud.addCallback(ArduinoIoTPreferredConnectionEvent::DISCONNECT,
onCloudEvent);
}

void loop() {
  // Întreținerea conexiunii la cloud
  ArduinoCloud.update();
}
```

Rezultate Obţinute

Prezentare smart home youtube

Layout proiect

Concluzii

Proiectul Smart Home IoT demonstrează cum diverse dispozitive dintr-o casă inteligentă pot comunica eficient cu o aplicație remote, utilizând tehnologia ESP32 și diverse protocoale de comunicare. Implementarea soluțiilor de detectare a scurgerilor de gaze și apă, precum și simularea controlului luminilor, subliniază importanța monitorizării și gestionării resurselor într-un mediu domestic.

Securitatea și partitionarea resurselor hardware au fost aspecte centrale în designul acestui sistem, abordând problemele potențiale legate de atacurile cibernetice. Prin utilizarea unui protocol închis pentru comunicarea internă și limitarea expunerii nodurilor critice, am reușit să minimizăm riscurile asociate conectivității IoT.

Acest proiect nu doar că oferă o soluție practică și funcțională pentru gestionarea unei case inteligente, dar servește și ca bază pentru dezvoltări viitoare în domeniul IoT, punând accent pe securitate și eficiență.

Bibliografie/Resurse

Referinte

- Arduino Cloud
- Lab 5 - 2023-2024
- Lab 4 - 2023-2024
- Lab 1 - 2023

Export to PDF