Smart Home Security System

A report submitted in partial fulfillment of the semester 6 Networking course assignment of

BSc (Hons) in Computer Science

Supervisor: Dr. Adnan Iqbal

Department of Computer Science
Namal College, Mianwali, Pakistan

Muhammad Hamza
2016-UET-NML-CS-28
Department of Comp Science

Ammar Farooq Khan
2016-UET-NML-CS-20
Department of Comp Science

Umer Farooq
2016-UET-NML-CS-17
Department of Comp Science

Document Version: 1.0
Last Updated: 31-March-2019

1 Table of Contents

2 Introduction

The project describes the smart home security system using the edge concepts of the Internet of Things. In this project, we have used Arduino Uno with an external Wi-Fi Module which connects the Arduino board to the internet or cloud for data storage. This module sends the data of two sensors e.g. Motion and Distance sensors to check the availability of human being or animals outside the home especially in late nights to handle thieves, dacoits, and suspicious people.

2.1 Functional Requirements

1. Client (Arduino) should be capable of sending data to the server.
2. The server must be able to understand the client's request to handle the request.
3. The server must store the data in the database.
4. Real Customer either Web or Mobile App should be able to request for pre-stored data on the database from the server.
5. The server must be able to understand the clients (App) request to handle the request and must respond accordingly.

2.2 Non-functional Requirements

1. Mobile or Web Application Graphical Interface must be user-friendly.
2. It must contain relevant features.
3. The view should be fascinating.
4. The graphical representation must be used to represent the data more effectively.

3 Tools

In this project, we have used an Arduino Board Uno with Wi-Fi Module and two sensors including Distance and Motion sensors.

3.1 PIR Sensor (#555-28027)

The PIR (Passive Infra-Red) Sensor is a pyroelectric device that detects motion by sensing changes in the infrared (radiant heat) levels emitted by surrounding objects. This motion can be detected by checking for a sudden change in the surrounding IR pattern. When motion is detected the PIR sensor outputs a high signal on its output pin. This logic signal can be read by a microcontroller or used to drive an external load.

3.1.1 Features

- Detect a person up to approximately 30 ft away, or up to 15 ft away in reduced sensitivity mode.
- A jumper selects normal operation or reduced sensitivity.
- Source current up to 12 mA @ 3 V, 23 mA @ 5 V.
- Onboard LEDs light up the lens for fast visual feedback when movement is detected.
- Mounting holes for #2 sized screws.
- 3-pin SIP header ready for breadboard or through-hole projects.

- Small size makes it easy to conceal.
- Easy interface to any microcontroller.

3.1.2 Key Specifications

- Power Requirements: 3 to 6 VDC; 130 µA idle, 3 mA active (no load).
- Communication: Single bit high/low output.
- Operating temperature: 32 to 122 °F (0 to 50 °C).
- Dimensions: 1.41 x 1.0 x 0.8 in (35.8 x 25.4 x 20.3 cm).

[1]

3.2 Arduino Uno

Arduino Uno is a microcontroller board based on 8-bit ATmega328P microcontroller. Along with ATmega328P, it consists of other components such as crystal oscillator, serial communication, voltage regulator, etc. to support the microcontroller. Arduino Uno has 14 digital input/output pins (out of which 6 can be used as PWM outputs), 6 analog input pins, a USB connection, A Power barrel jack, an ICSP header, and a reset button.

3.2.1 Technical Specification

[2]

Microcontroller	ATmega328P – 8bit AVR family microcontroller
Operating Voltage	5V
Recommended Input Voltage	7-12V
Input Voltage Limits	6-20V
Analog Input Pins	6 (A0 – A5)
Digital I/O Pins	14 (Out of which 6 provide PWM output)
DC Current on I/O Pins	40 mA
DC Current on 3.3V Pin	50 mA
Flash Memory	32 KB (0.5 KB is used for Bootloader)
SRAM	2 KB
EEPROM	1 KB
Frequency (Clock Speed)	16 MHz

3.3 Distance Sensor (HC-SR04)

The HC-SR04 ultrasonic sensor uses sonar to determine the distance to an object like bats do. It offers excellent non-contact range detection with high accuracy and stable readings in an easy-to-use package. It comes complete with ultrasonic transmitter and receiver modules.

3.3.1 Features

Here’s a list of some of the HC-SR04 ultrasonic sensor features and specs:

- Power Supply: +5V DC
- Quiescent Current : <2mA
- Working Current: 15mA
- Effectual Angle: <15°
- Ranging Distance : 2cm – 400 cm/1″ – 13ft
- Resolution : 0.3 cm
- Measuring Angle: 30 degree
- Trigger Input Pulse width: 10uS
- Dimension: 45mm x 20mm x 15mm

[3]

3.4 Wi-Fi Module (ESP8266)

ESP8266 offers a complete and self-contained Wi-Fi networking solution, allowing it to either host the application or to offload all Wi-Fi networking functions from another application processor.

When ESP8266 hosts the application, and when it is the only application processor in the device, it is able to boot up directly from an external flash. It has integrated cache to improve the performance of the system in such applications, and to minimize the memory requirements.

Alternately, serving as a Wi-Fi adapter, wireless internet access can be added to any microcontroller-based design with simple connectivity through UART interface or the CPU AHB bridge interface.

3.4.1 Features

- 802.11 b/g/n protocol
- Wi-Fi Direct (P2P), soft-AP
- Integrated TCP/IP protocol stack
- Integrated TR switch, balun, LNA, power amplifier and matching network
- Integrated PLL, regulators, and power management units
- +19.5dBm output power in 802.11b mode
- Integrated temperature sensor
- Supports antenna diversity
- Power down leakage current of < 10uA
- Integrated low power 32-bit CPU could be used as an application processor
- SDIO 2.0, SPI, UART
- STBC, 1×1 MIMO, 2×1 MIMO
- A-MPDU & A-MSDU aggregation & 0.4µs guard interval

- Wake up and transmit packets in < 2ms
- Standby power consumption of < 1.0mW (DTIM3)

[4]

4 Architecture

Our system is based on pure client-server architecture. This system has 3 to 4 components.

1. Arduino (Client 1: Data Sender)
2. Real Client (Real Customer Application) (Client: Web or Mobile App, the data viewer)
3. Server (Including Database: Data Manipulator)

I. **Arduino** uses the sensors mentioned in the tools sections and sends the data of sensors to server in the form of message format (Sending Message) mentioned in the section Message Format and in case Arduino (client) is not registered with the server, first it has to register itself with the server by sending a message in the message format (Register Message) mentioned in section Message Format.

II. **The server** gets request of Arduino (client) and separates all the messages parts, processes it and stores the data in the database.

III. **Real Client** either Mobile or Web App requests the server in the form of message format defined in the section Message Format (GET Message) for the data of either or both the sensors for a specific interval and in case the particular Web or Mobile App is not registered with the server, first it has to register itself with the server by sending a message in the message format (Register Message) mentioned in section Message Format and the data request can be done.

IV. **Server Response** can vary with clients (real or Arduino client) request, in case the request is about Registration, the client would get “OK” and if the request is about the sensor data or values, the response will contain the values of the sensor as requested by the client.

5 Protocol Design

5.1 Overall Protocol Description

This protocol is designed for communication between Clients and Server. How client (Arduino) will send the request and how the server will respond to it, what will be the message type and what will be the message format which will be known on both sides.

Protocols are the rules and basis of client-server communication and therefore protocol must remain consistent on both side for errorless.

Below are the sections that define the **“Message Types and Formats”.**

5.2 Message Types

There are two types of messages in this application.

1. **Sending Message**
    a. Always Arduino(Using Wi-Fi Module) To Server
    b. Contains the sensors (Motion and Distance) values.
2. **Register Message**
    a. To and fro communication from real client and server.
    b. Contains the IP, Port, Username, and Password.
    c. The server will respond with OK message after registering the user successfully.
3. **GET Message**
    a. To and fro communication from the real client (Application) and server.
    b. Contains the IP, Port Number, Username, Password and Value Code.
    c. For Example Code (Both, Sensor 1, Sensor 2)
4. **Server Response Against (Register and GET Message)**
    a. Always from the server to the client.
    b. Contains “OK” in case successful registered
    c. Contains “Values” in case GET Message.

5.3 Message Formats

1. **Sending Message**

Sending:	Motion	//	Distance	\r\n	Values:	Sensor 1	//	Sensor 2

This is a sending message where the first line contains the description and second lines contain the real values of both the sensors.

2. **Register Message**

Register:	IP	//	Port	\r\n	Username	//	Password

This is a registration message where first lines contain the IP and Port of the sender, 2nd lines contain the username of the user and 3rd lines contain the password of that user who wants to register.

3. **<u>GET Message</u>**

GET:	IP	//	Port	\r\n	Username	//	Password	\r\n	Value Code

This is a get message where first and second line contains the IP and Port, username and `password of the already registered user and the last line contains the value code that we want to get from the server.

Where value code could be S1, S2 or Both.

4. **Server Response Message**

Type:	Response	\r\n

The type could be Registered or Values
The response could be OK or Values of Sensors (S1//S2).

5.4 Message Semantics

- **Sending Message**
    - When the server receives the message of the client (Arduino) it processes the message and saves the values of sensors in the database.
- **Register Message**
    - When the server receives the message of the real client (Web or Mobile App) it processes the message and saves the IP, Port Number, Username and Password in the database.
- **GET Message**
    - When the server receives the message of our real client (Web or Mobile App) it processes the message and extracts the required sensors values according to value code.
- **Response Message**
    - The server sends the extract sensors values to the real client.

6 References

[1] Parallax, "555-28027-PIR-Sensor-Prodcut-Doc-v2.2," [Online]. Available: https://www.parallax.com/sites/default/files/downloads/555-28027-PIR-Sensor-Prodcut-Doc-v2.2.pdf.

[2] Components101, "arduino-uno," 28 March 2019. [Online]. Available: https://components101.com/microcontrollers/arduino-uno.

[3] "complete-guide-for-ultrasonic-sensor-hc-sr04/," 28 March 2019. [Online]. Available: https://randomnerdtutorials.com/complete-guide-for-ultrasonic-sensor-hc-sr04/.

[4] "esp8266-datasheet/," 28 March 2019. [Online]. Available: https://www.electroschematics.com/11276/esp8266-datasheet/.