

The Implementation of a Smart Home Security Network Using Internet of Things (IoT) System

Alaa M. Odeh¹ and Isam Ishaq²

¹Al-Quds University, Jerusalem, Palestine
alaaodeh.2019@gmail.com

²Al-Quds University, Jerusalem, Palestine
isam@itce.alquds.edu

Abstract. One of the foremost concerns for each family is to keep their home secure and safe for living. And perhaps this is what we keep thinking about all day. For this reason, this paper introduces a designed Internet of Things (IoT) home security system. The proposed work contains three main circuits. Each contains a sensor and an actuator that work based on the data received by the sensor. These three circuits are: a temperature monitoring circuit to monitor the temperature of a room and activate the fan when needed, a gas and smoke detection circuit to monitor gas leakage and notify when gas or smoke is detected, and a touch detection circuit to secure home from robbery and unauthorized entry. All these circuits are connected to an ESP32 controller which controls their working and sends continuous sensors' readings to the ThingSpeak cloud. The paper presents a complete hardware and software description of the system. In addition to some followed security settings to ensure secure data transmission and reception.

Keywords: Internet of Things, ESP32, LM35 sensor, MQ gas sensor

1 Introduction

One of the foremost concerns for each family is to keep their home secure and safe for living. All of us seek to protect our family members from any possible danger. And perhaps this is what we keep thinking about all day. As an example, we check multiple times before going out if we have turned off the iron or the gas pipe. However, Parents sometimes forget to turn off the oven and this leads the meal to be burned. All of this is exhausting. Fortunately, employing the IoT system would make this much easier for parents and they will be able to protect their homes with a click on their smartphones.

For that reason, the researchers designed a system that would allow monitoring different parts of the home, sending data to the processor, and permitting a specific reaction. To be more precise, this IoT system will allow home members to first, monitor room temperature, for example, observe if they are overheated and then, activate an air conditioner or air fan. Moreover, individuals can monitor their kitchen and observe if there is a gas leakage, smoke, or extra heat. In that case, they can control the gas pipe, the oven, or the water sprinkler. In addition, people in the home will be able to detect

if there is unusual motion outdoor, glass breaking, or if some stranger is trying to sneak inside and subsequently launches the siren.

Yet, before delving deeply into implementing such a system, one has to have a clear picture of the process that should be followed, the materials and methods, how this work is related to other previous works, and what improvements it has. Hence, the following sections provide a clear picture of all the needed information.

1.1 Project's Aim and Objectives

The **overall aim** of this research is to employ the IoT concepts in implementing a smart security home system. In this respect, the main **objectives** are:

1. Construct a smart temperature monitoring circuit and use the data extracted to control the temperature inside the house.
2. Construct a smoke detection circuit and utilize the data collected by the sensor to launch the siren when necessary.
3. Construct a touch detection circuit for door security.

2 Literature review

This section provides a review of the utilization of IoT in previous works that are relevant to the current project.

The ESP32 chip can be used in applications that perform various monitoring and security functions. The high-quality, low-cost solution in numerous fields, such as air quality system monitoring, is a common element of the deployed applications which can be found in the work proposed by Sarjerao & Prakasarao (2018). In this work, the MQ-9 is used to measure CO, LPG, and CH₄ levels, the DHT22 is used to measure temperature in degrees Celsius and humidity in percent, and the dust sensor is used to measure dust particles, the MQ-135 is used to measure benzene, ammonia and sulfur dioxide in the environment. These measured data are transferred to the server using the REST protocol (REp-resentational State Transfer protocol) and ESP32. All the measured values from the sensors are displayed on the website and mobile application (Sarjerao, 2018). Although their proposed system had a great effect in controlling air pollution, it differs from this project because it only detects the presence of gas or smoke and does not indicate the type of gas detected.

While Budijono and Felita in their research suggested a Smart temperature measurement with the ESP32, which is used to monitor the performance of the freezer, measure the temperature continuously over some time, and show the measurement results to be calculated for preventive maintenance (Budijono, 2021). However, in their study, they used a DS18B20 temperature sensor and aimed to measure relatively low temperatures, which is different from the system proposed here to measure room temperature using an LM35 temperature sensor.

Andreas and colleagues in their research have proposed an application called Door Security System, which is based on Android and uses IoT technology to monitor door status, control the door, and enhance security in a home. MQTT cloud is the protocol

that is used to communicate between cellphones and the door lock system. PIR sensors are used in door locks to detect movement near the door, whereas touch sensors are used in door handles to detect human hands. If the door is forced open, the alarm will ring and send a notification to inform the occupants of the house about the existence of an intruder in the house (Andreas, 2019). The use of a touch sensor is somehow similar to the current work, but the alert here is by using an alarm system or a buzzer, no mobile application was implemented. But continuous data is sent to the cloud which allows discovery when the door was touched especially if the owners are outdoors.

On the other hand, Upadhyay and his colleagues used the Atmega-8 microcontroller to control a system that is designed to detect data using the reed switch and vibration sensors and communicate to the cloud using an ESP 32 WIFI module. The system uses IoT to detect an anomaly, which refers to the network of connected objects that can communicate and exchange data among themselves. It is used for various applications in the safety and surveillance industry (Upadhyay, 2021). Vibration sensors can detect any unusual movements near objects like doors. But the use of the ultrasonic sensor is more accurate and gives precise results by calculating the difference in distance based on a change in room temperature.

As for security solutions, S. Babar, A. Stango, N. Prasad, J. Sen, and R. Prasad provide a detailed overview and analysis of embedded security, especially in the IoT domain, in their paper. The paper highlights the need to create integrated security within the device itself to provide a flexible infrastructure for dynamic prevention, detection, diagnosis, isolation, and countermeasures against successful breaches. Based on this overview and analysis, security requirements are defined taking into account device computation time, energy consumption, and storage requirements. Then, an embedded security framework is proposed as a feature of the software/hardware co-design method (S. Babar, 2011).

Tsvetanov and Pandurski proposed an algorithm in their article that ensures the security of data transfer from the sensor network to the cloud using the DLS security protocol. The ThingSpeak cloud is used to implement the algorithm, and the MQTT protocol is used to establish a link between the sensor network and the cloud. The flow that collects the sensor data is created with Node-RED, and a self-signed certificate is generated with OpenSSL in this process. The algorithm was tested using Wireshark software, which confirmed that the data packets sent from the sensor network to ThingSpeak use the MQTT data transfer protocol and connect to communication port 8883 via TLS / SSL protocol, which is protected/encrypted (Pandurski, 2021). This proposed work was very useful for adding security solutions to the current work. In addition to the API security solution, the researcher used Wireshark to monitor data transmission from the ESP32 to the cloud.

3 Methodology

3.1 Introduction to project methodology

This project focused on how a smart security system was implemented. For this reason, there is a need to provide a clear picture of the complete process of constructing this

project. So, this chapter provides an outline of the methods that were followed when this project was implemented. It presents a description of the project design; a hardware description and a software description. The researchers also discuss how this project was installed and how its results were tested as well.

3.2 General view

The system was implemented using an ESP32 microcontroller, temperature sensor, smoke and gas sensor, and the ESP32 build-in touch sensor. Arduino programming language was used as software, and to connect with the internet, several channels were created on the ThingSpeak website. Fig.1 represents the block diagram of the system. There are no systems that exactly do the same work. But the proposed system would easily enhance the monitoring and controlling of the home appliances and easily updates to the open-source cloud as well.

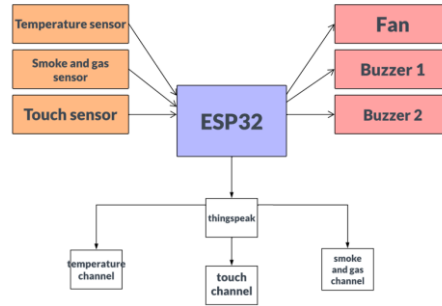


Fig. 1. Block diagram of the system

For more clarification, Fig. 2 shows how the sensors and actuators are distributed and connected with the ESP32 microcontroller, and how it is wirelessly connected with the website.

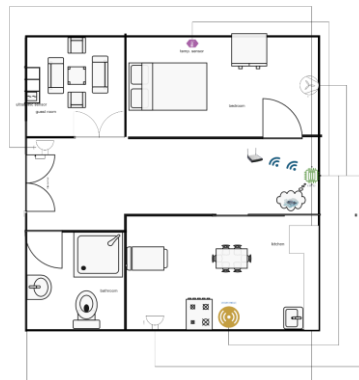


Fig. 2. system symbolic design

3.3 Hardware description

The hardware description section is to describe the structure, design, and operation of the electronic circuits of the proposed system. Essential components will be briefly introduced and at the end, the schematics of the resultant circuit and the connection in real life will also be presented.

3.3.1 Description of the basic components

3.3.1.1 ESP 32

For a wide range of applications, the ESP32 is a feature-rich MCU with built-in Wi-Fi and Bluetooth connections. It's made for mobile phones, portable electronics, and IoT applications. The ESP32 uses a combination of proprietary software to achieve ultra-low power consumption. The ESP32 can be utilized as a stand-alone system or as a slave device for a host MCU, lowering the main application processor's communication stack overhead. The ESP32 can be used to bring Wi-Fi and Bluetooth functionality to other systems.

3.3.1.2 LM35 temperature sensor

The LM35 is a temperature sensor that produces an analog signal proportional to the temperature at any given time. The output voltage can be simply translated into a Celsius temperature value. The advantage of the LM35 over a thermistor is that it doesn't need to be calibrated externally.

3.3.1.3 Smoke and gas detector (MQ)

The MQ2 gas sensor module is excellent for detecting gas leaks (home and industrial). It can detect H₂, LPG, CH₄, CO, alcohol, smoke, or propane, among other gases. A measurement may be conducted as quickly as possible thanks to its high sensitivity and quick reaction time.

3.3.1.4 Touch sensor

There are ten capacitive touch GPIOs on the ESP32. Changes in anything with an electrical charge, such as human skin, can be detected by these GPIOs. As a result, you may detect changes caused by touching the GPIOs with your finger. These pins are easily integrated into capacitive pads and can be used to replace mechanical buttons. In addition, when the ESP32 is in deep sleep, the touch pins can be used as a wake-up source.

3.3.2 circuits' connections

The whole system here is divided into subsystems for better measurements.

3.3.2.1 Temperature detection circuit

To measure the temperature in a room, a temperature sensor was used. Fig. 3 and 4 show the connection of the temperature detection circuit. Only the LM35 sensor and a servo motor were connected with the ESP32. No other components were needed. The

left leg of the LM35 is connected with the Vcc of the ESP32, the right leg is connected with the ground pin, and to provide measurements to the ESP32, the middle leg is connected with pin 35, which is an ADC pin and it can be used as an input pin or output pin. The servo motor or a fan is connected to work when a high temperature is detected. The positive leg of the servo motor is connected with io25. And the negative one is connected with the ground pin of ESP32.

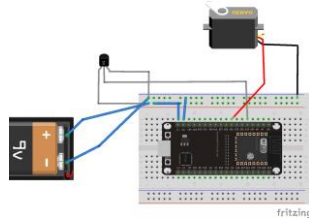


Fig. 3. temperature detection circuit

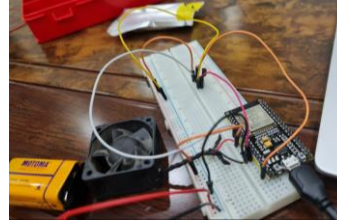


Fig. 4. real temperature detection circuit connection

3.3.2.2 Smoke and gas detection circuit

To detect when there is smoke or gas leakage, smoke and gas detection circuit was built (Fig 5&6). In this circuit, the MQ sensor analog leg A0 is connected with the io33 pin, the digital leg D0 is connected with the io35 pin, the Vin is connected with the Vcc of the ESP32, and the GND is connected to the ground pin of the ESP32. A buzzer is used to produce sound when gas is detected. The positive leg of the buzzer is connected with io32 and the negative leg is connected with the ground pin of the ESP32.

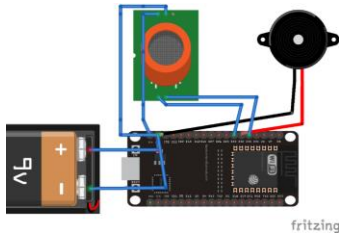


Fig. 5. smoke detection circuit

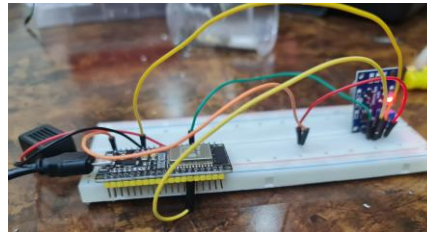


Fig. 6. In practice smoke detection circuit connection

3.3.2.3 Touch detection circuit

To develop a security door system, a touch sensor can be used on the door and its hand so whenever tries to use its hand while everybody is out, an alarm system will work. However, as mentioned earlier in this section, The ESP32 has 10 capacitive touch GPIOs. These GPIOs can sense variations in anything that holds an electrical charge, like the human skin. So, instead of using an external touch sensor and getting bothered with its connectivity, a wire was connected with one of the 10 capacitive touch GPIOs, which is D4. A buzzer was used for wire touching detection. The circuit connection is presented in Fig. 7. and the real circuit connections are in Fig. 8.

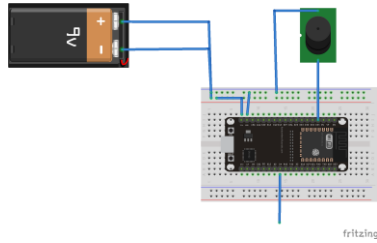


Fig. 7. touch detection circuit

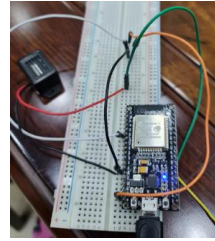


Fig. 8. on-the-ground touch detection circuit connection

3.3.3. The system altogether

After connecting all parts and circuits, the resultant system is shown in Fig. 9. This system represents home security and safety where temperatures, smoke and gas, glass breaking, and door touching can be monitored and detected.



Fig. 9. entire home with circuits connection

3.4 Software description

This system monitors the temperature, the gas and smoke leakage, glass breaking through the ultrasonic sensor, and door security through the touch sensor. If the temperature exceeds the defined value, the fan will work. If the gas sensor detects any smoke or gas, the buzzer will be launched. And finally, when the wire (touch sensor) is touched, the Alarm system will go off. The ESP 32 sends sensors' data to pre-created channels in the Thingspeak website which can be opened and visualized anytime. These all situations can be summarized using a flow chart for the whole system (Fig. 10).

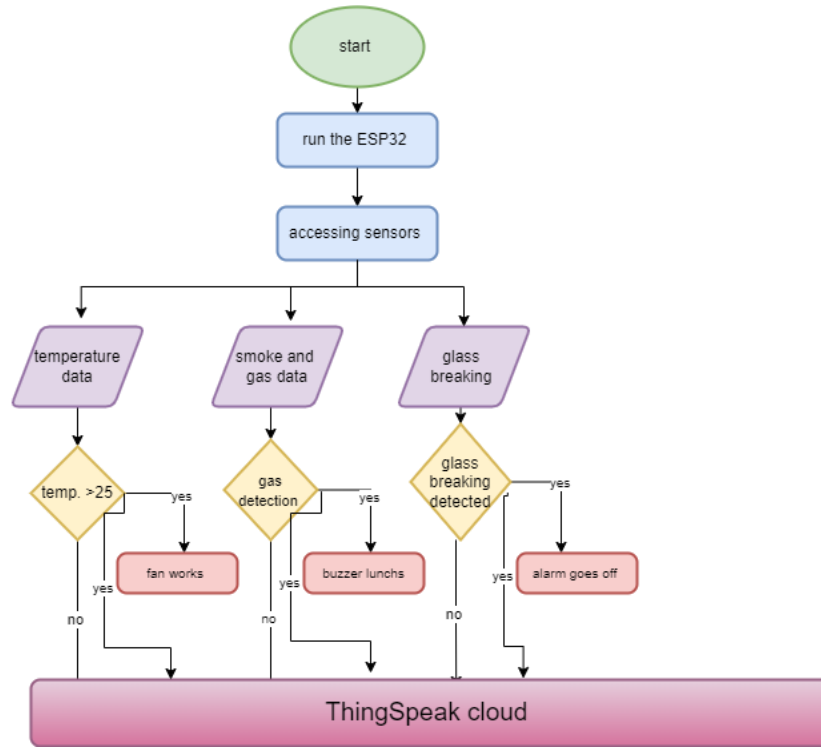


Fig. 10. flowchart of the system

For more understanding of how the system works, a sequence diagram was designed for each detection circuit. In the following, each sequence diagram with clarification is illustrated.

3.4.1 Temperature circuit sequence diagram

- The temperature sensor detects temperature and sends data to ESP32.
- Based on temp. state, the ESP32 decides whether to set the fan on or off.
- All temperature data is sent to the ThingSpeak cloud, specifically, the temperature channel (Fig. 11).

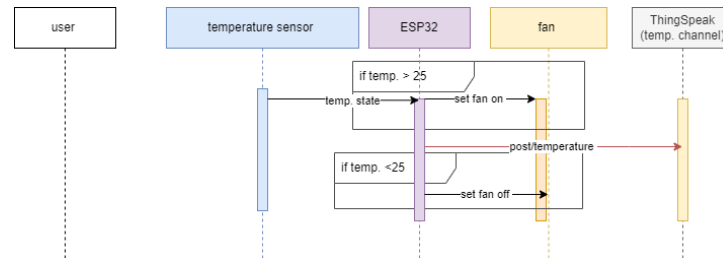


Fig. 11. Temperature circuit sequence diagram

3.4.2 Smoke and gas detection circuit

- The smoke sensor detects gas leakage (Fig. 12).
- It sends data continuously to the ESP 32.
- If there is gas or smoke, the ESP32 will set the buzzer on.
- The ESP 32 sends continuous data to the smoke channel in the ThingSpeak cloud.

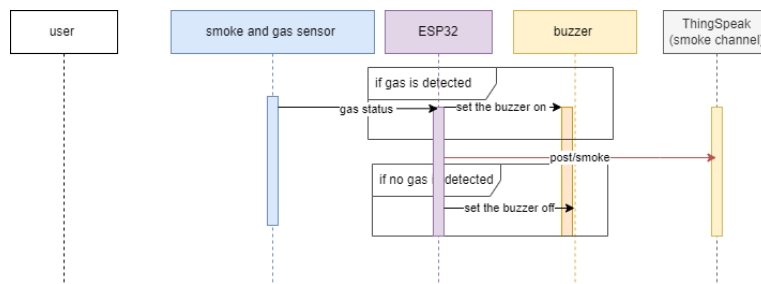


Fig. 12. smoke detection sequence diagram

3.4.3 Touch detection circuit

- As mentioned earlier, there is no touch sensor. Only a wire is connected to the touch GPIO pin of the ESP32. But it acts as a touch sensor
- When someone touches this wire, the ESP32 will record this and set the buzzer on. Otherwise, the buzzer will be off.
- Continuously, data will be sent to the touch channel in the ThingSpeak cloud.

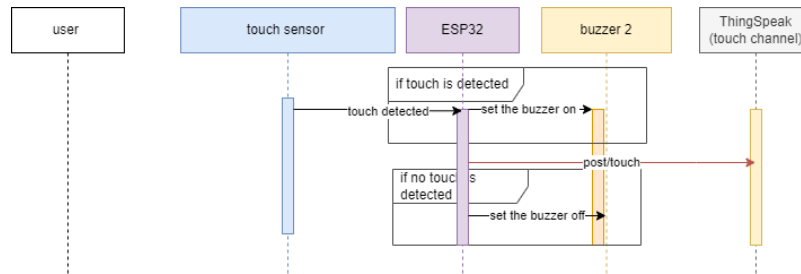


Fig. 13. sequence diagram of the touch detection circuit

3.5 System programming

3.5.1 Programming with Arduino IDE

There is an add-on for the Arduino IDE, which allows you to program the ESP32 with the Arduino IDE and its programming language. You just have to select the ESP32 board, connect it to the computer, and then start writing the codes. Fig. 14 shows an example of code written in Arduino IDE and upload on the ESP32.

```

sensorsmoke | Arduino 1.8.10
File Edit Sketch Tools Help

sensorsmoke
8 Serial.begin(115200);
9 pinMode(Buzzer, OUTPUT);
10 pinMode(Gas_digital, INPUT);
11 }
12
13 void loop() {
14   int gassensorAnalog = analogRead(Gas_analog);
15   int gassensorDigital = digitalRead(Gas_digital);
16
17   Serial.print("Gas Sensor: ");
18   Serial.print(gassensorAnalog);
19   Serial.print("\t");
20   Serial.print("Gas Class: ");
21   Serial.print(gassensorDigital);
22   Serial.print("\t");
23   Serial.print("\t");
24   if (gassensorAnalog > 1000) {
25     Serial.println("Gas");
26     digitalWrite (Buzzer, HIGH) ; //send tone
27     delay(1000);
28     digitalWrite (Buzzer, LOW) ; //no tone
29   }
30   else {
31     Serial.println("No Gas");
32   }
33   delay(100);
34 }

Done uploading.
Leaving...
Hard resetting via RTS pin...
  
```

Fig. 14. uploading the code to the ESP32 is done successfully

3.5.2 Creating ThingSpeak channels

- Open ThingSpeak website
- Create an account
- Start creating channels
- Copy the API key of each channel and paste it into the code so each sensor data will be sent to that specific channel (Fig. 15. presents these steps).

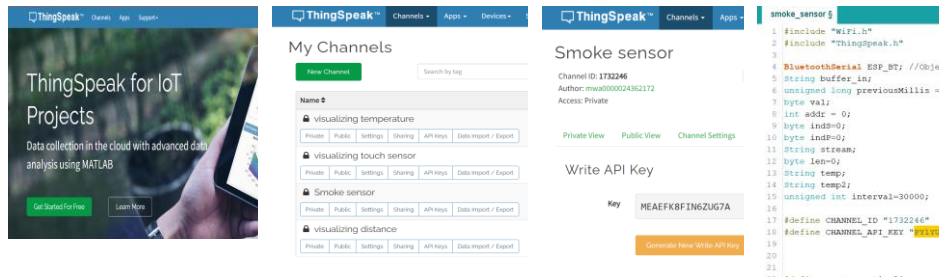

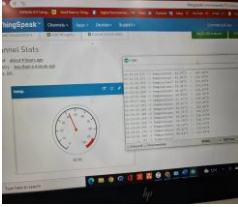




Fig. 15. creating channels in ThingSpeak to store data

4 Results and discussion

Each sensor in this system receives data from the environment and sends it to ESP32. The ESP32 records this data and sends it to ThingSpeak channels. Also, when some condition is satisfied, a consequence will happen depending on that condition. For example, if the temperature is greater than 25, the fan will work. However, the recorded data can be viewed from the ThingSpeak channel visualization or using the Arduino IDE monitoring. In the following table, each sensor circuit's results are summarized with comments along with the result.

Name	Normal situation	Abnormal situation	Comments
Temp detection circuit			These two pictures show the results on both, Arduino IDE and ThingSpeak channel visualization. In the first picture when the detects the approximate room temperature of 25°. In the second picture when the hair drier is n to the LM35.
Smoke detection circuit			The left picture is the sensor readings when it started to detect cigarette smoke. The right picture is when the smoke is very close to the MQ.

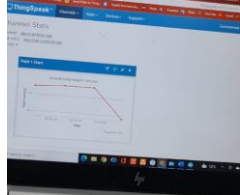
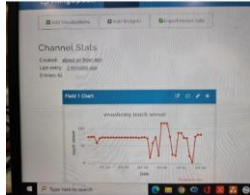
Name	Normal situation	Abnormal situation	Comments
Touch detection circuit			The left picture shows the reading when first the wire was touched and then left. The second picture shows continuous touching of the wire that is connected to ESP32.

Table 1. results

5 System security

The biggest challenge for IoT systems is IoT security. IoT devices were not designed with security in mind, leading to potential vulnerabilities in a system with multiple devices. In most cases, there is no way to install security software on the device itself. Also, sometimes they come with malware that then infects the network they are connected to. However, you can set some security settings to ensure that data is sent from the ESP32 to ThingSpeak channels without being hacked. These settings are:

- ThingSpeak API servers support secure HTTPS and MQTT connections between devices and ThingSpeak. When creating a channel for receiving data, this channel has an ID and an API key. These two are used in the code to ensure that data only will be sent to the channel that has this ID and this API key. No one else can view the data unless sharing is permitted.
- Use the Wireshark tool to monitor and analyze sent and received data packets and analyze network logs. Wireshark is the world's leading and widely-used network protocol analyzer. It has many important features, some of which meet our requirements:
 - Deep inspection of hundreds of protocols, and new ones are being added all the time.
 - Live capture and offline analysis
 - Captured network data can be searched through a graphical user interface (GUI) or the T-Shark utility in TTY mode.

Analyzing send and receive data packets were done with these steps:

1. Clear the cache of the browser
2. Open Wireshark and click on capture
3. Open the browser and type thingspeak.com

The capture should display that the source has the IP number the same as the computer IP number (192.168.1.245) and the destination has the same IP number as the ThingSpeak website IP number (3.226.58.189). the results from Wireshark are shown in Fig 16.

The screenshot shows a Wireshark packet capture window titled 'thingSpeak.pcapng'. The packet list pane shows several packets, with packet 70 selected. The packet details pane shows the structure of the selected packet, and the packet bytes pane shows the raw data in hexadecimal and ASCII.

No.	Time	Source	Destination	Protocol	Length	Info
68	13.852102	136.143.191.144	192.168.1.245	TLSv1.2	85	Application Data
69	13.897900	192.168.1.245	136.143.191.144	TCP	54	49610 → 443 [ACK] Seq=73 Ack=63 Win=511 Len=0
70	14.991007	192.168.1.245	3.226.58.189	TLSv1.2	1500	Application Data
71	15.285651	3.226.58.189	192.168.1.245	TLSv1.2	551	Application Data
72	15.339221	192.168.1.245	3.226.58.189	TCP	54	59178 → 443 [ACK] Seq=4308 Ack=1492 Win=516 Len=0
73	16.999143	192.168.1.245	198.41.30.198	TCP	55	49677 → 443 [ACK] Seq=1 Ack=1 Win=515 Len=1 [TCP

Internet Protocol Version 4, Src: 192.168.1.245, Dst: 3.226.58.189

Transmission Control Protocol, Src Port: 59178, Dst Port: 443, Seq: 2862, Ack: 995, Len: 1446

Source Port: 59178

Destination Port: 443

[Stream index: 4]

[Conversation completeness: Incomplete (12)]

[TCP Segment Len: 1446]

0000 c4 68 d0 76 b3 92 a8 7e ea 06 39 43 00 00 45 00 ..h.v....-9C..E-

0010 05 ce 20 28 40 00 00 00 00 00 c0 a8 01 f5 03 e2 ..@.....&{30-P-

0020 3a bd e7 2a 01 bb d9 e6 16 26 7b 33 4f 93 50 18&{30-P-

0030 02 00 06 fd 00 00 17 03 03 05 a1 00 00 00 00 00m.F.ek...-cq

0040 00 00 6d b9 46 a4 65 6b fb c8 5f 7d fe 2d 63 71 ..-@-L...->R0'..

0050 af 7b 8e 40 f0 4c e5 f5 88 a1 3e 52 30 27 92 29 ..uq....A...3~\

0060 e2 ee 75 71 0e b8 da be e5 41 dc de 97 33 7e 5c ..c]g5-I 1)...d...

0070 b7 16 63 5d 67 53 9d 49 31 29 0e ca 64 c0 01 b7

Fig. 16. The results from the Wireshark packets capture.

This packet capturing allows users to monitor where data are going all the time and ensure that they are received by the correct destination.

6 Importance of the project

There is no doubt that a home security system aims to protect one's property and those inside it from robbing, home intrusion, fire, and other environmental disasters such as burst pipes. However, since this project has many security aspects, each one of them has its importance. For example, most robbery crimes are committed when the owners are outside their houses and the robbers attempt to break a window to get into the house. So, developing a low-cost glass breaking detection would prevent any attempt of breaking glass and hence, prevent robbery. Another important aspect is protecting residents from fires and air pollution. The third goal of the developmental sustainable goals is ensuring good health and well-being (UNDP, n.d.). One of its targets is that By 2030, the number of deaths and illnesses caused by hazardous substances and pollution of the air, water, and soil can be cut in half. And since this project will make it possible to equip every home with a very low-cost gas detection system, the number of deaths from polluted air and fire will surely decrease as well.

7 Future work

Even though this project can be implemented easily, it has relatively low-cost components, and its software was easy to be designed and described, but, somehow the period when this project was implemented was relatively short. So, some future modifications can be added to this project to ensure having complete smart home security and safety system. These modifications are:

- Add a webcam to the system and train it for face recognition. This will increase security affairs
- Connect the system with a mobile application and control the appliances from this application. This would enhance the overall controlling and monitoring of the system
- Instead of using electricity to power the system, power the system using solar energy. This will decrease the cost and hence, increase the number of homes that will use it.

8 Conclusion

In this paper, the researcher proposed a designed IoT home security system. The proposed work contained three main circuits. Each one of them contains a sensor and an actuator that works based on the data received by the sensor. The three circuits are a temperature monitoring circuit to monitor the temperature of a room and launch the fan when needed, a Gas and smoke detection circuit to monitor gas leakage and notify when gas or smoke is detected, and a touch detection circuit to secure home from robbery and unauthorized entry. All these circuits are connected to the ESP32 controller which controls their working and sends continuous sensor readings to the ThingSpeak cloud. The paper presented a complete hardware and software description of the system. It also provided some followed security settings to ensure secure data transmission and reception. The importance of the system lies in its role in decreasing robbery attempts, controlling rooms' temperature, and decreasing the number of deaths caused by fires and gas leakage which is the third goal of the sustainable development goals.

References

1. Andreas, C. R. (2019). Door Security System for Home Monitoring Based on ESP32. *Procedia Computer Science*, 673-682.
2. Budijono, F. (2021). Smart Temperature Monitoring System Using ESP32 and DS18B20. *4th International Conference on Eco Engineering Development 2020*.
3. Chen, S. Z. (2020). Indoor temperature monitoring using wireless sensor networks: A SMAC application in smart cities. *Sustainable Cities and Society*.
4. Pandurski, F. A. (2021). Security of the sensory data in the cloud. *2021 IOP Conf. Ser.: Mater. Sci. Eng.* 1032.
5. S. Babar, A. S. (2011). Proposed embedded security framework for Internet of Things (IoT). *2011 2nd International Conference on Wireless Communication, Vehicular Technology, Information Theory and Aerospace & Electronic Systems Technology (Wireless VITAE)*, (pp. 1-5).
6. Sarjerao, B. &. (2018). A Low Cost Smart Pollution Measurement System Using REST API and ESP32. *3rd International Conference for Convergence in Technology (I2CT)*, (pp. 1-5).
7. Upadhyay, P. a. (2021). Theft Detection Using Data Science. *Proceedings of the 4th International Conference on Advances in Science & Technology (ICAST2021)*.