IoT Based Covid-19 Social Distance Monitoring And Contact Tracing System

Mayank Gupta, Mohit Patil, Yogeshwaran Logashanmugam

Department of Electrical and Computer Engineering

University of Florida

mohitpatil@ufl.edu, mayank.gupta@ufl.edu, y.logashanmugam@ufl.edu

***Abstract*— The pandemic control failure and unstoppable spread have highlighted the need for social distancing monitoring and contact risk alerts. The presented study demonstrates an IoT based low cost and scalable social distance monitoring system to alert users of risk and allow contact tracing to fight COVID-19 in an indoor or limited campus setting. The system allows users to monitor the room occupancy, tracks the users being in contact with each other, and alerts the user in case one of the people they've been in contact with tests positive for the COVID-19. To keep the cost to the user minimum, smartphones are used as user nodes and wifi access point infrastructure already existing in most of the places. The study uses Raspberry Pi B+ as a proxy to actual wifi routers. Thingspeak platform is used for cloud infrastructure and communication. A novel algorithm based on conditional analysis finds the users at the risk of contact with Covid positive users.**

***Keywords*— Contact tracing, Social distance monitoring, IoT, ThingSpeak**

I. Introduction

2020 has been an exceptional year and this pandemic has shown the gaps in our social systems. Social distancing, which was the simplest and most effective method to control the spread of viruses, failed to implement effectively. The project demonstrates the system that tracks the number of people in the room/office space allowing the user to make a decision whether to enter the room or not. It also limits entries to maximum occupancy. Using the room/sector wifi data within the indoor setting an alert system is created if the people density in the room/office space is above a threshold. The contact tracing system alerts the user when people they have come in contact with test positive for COVID-19.

The goal was to keep the cost minimum and cost to the user zero, and at the same time make it robust to allow widespread use, the study achieved this by using wifi access points and cloud set up as the only added hardware. Users just have to use their smartphones so no added cost is incurred to the user. In the presented demonstration, Raspberry Pi B+ is used as wifi access points [Fig.1]. ThingSpeak platform allowed easy implementation of cloud components and all communication channels. ThingView [1] is an open-source app to track the ThingSpeak channel that allowed live tracking of the room occupancy. ThingSpeak website also allows this tracking feature. The user is alerted using the email address that is registered with the employers or building administrators.

Previous studies depend on the added sensors and hardware infrastructure for the contact tracing applications [2,3,4]. Given the sensitivity of the situation, the proposed system can be scaled up quickly for wide implementation [5,6]. This is also an attribute of minimalistic and low-cost components used.

II. System Architecture

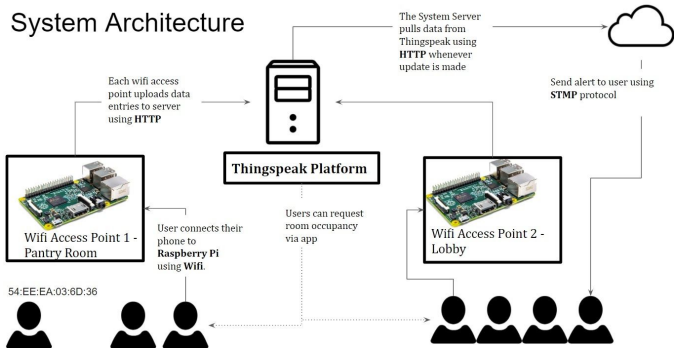

Fig. 1 System Architecture

The system uses smartphones as the proxy node for the user. The user connects to the wifi access point of the room he wishes to enter and use. The wifi access point allows entry only if room occupancy is below the threshold, which in this case is set to 5 people. Each user's data is logged in the ThingSpeak server using HTTP. The cloud

communication is also Hypertext Transfer Protocol(HTTP) based which lets the server pull data from all wifi access nodes for contact tracing purposes. The algorithm (explained in subsequent sections) determines the users who have come in contact with a COVID positive person in case a registered user in the office space is tested positive for COVID. All the users who have been detected as "In-Contact" with the COVID positive user will be alerted via email [Fig.2] which is generated using Simple Mail Transfer Protocol (SMTP). This system design falls under IoT-Level 6 as all individual nodes communicate with the server independently and with each other indirectly as well.

Fig. 2 Email alert for COVID positive contact

The users can monitor the live room occupancy for any room they want to use by checking the ThingView app [Fig.3]. The figure shows the screen of the app when the demo was showing 2 rooms and their live occupancy.

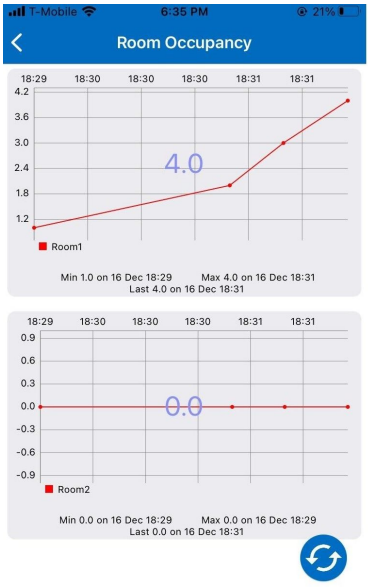

Fig. 3 ThingView app for the room occupancy monitoring

III. Hardware Description And Study Design

A. Setup

The study made use of two components - Raspberry Pi [Fig.4] and a smartphone with a working Wi-Fi module and the ability to receive emails. Raspberry Pi is a low-cost, small size computer that can handle most of the things desktop computers can do, and also, it offers vast modalities to connect and interact with external devices such as sensor modules. Raspberry Pi was chosen because of its low cost coupled with the fact that it already had everything needed for this System Design. Also, Raspberry Pi can manage to run Python scripts, which helps to communicate with the Thingspeak server while making modifications to the data (if required) before sending it to the Thingspeak Server.

Fig. 4 Raspberry Pi as the wifi access point

For demonstration, the Raspberry Pi serves as a terminal for the user to connect their phone to the Wi-Fi and be logged into the system.

In today's world, most people keep their smartphones with themselves at all times, especially in work or an office setup. The study leverages this habit for the functionality of the proposed solution. The smartphone serves as a terminal for a user to be logged into the system. To do that, the MAC Address of the device is obtained, which is static unlike an IP Address, and using MAC Address information people who were connected to the same router in the overlapping duration of time are tracked.

B. Network Architecture and Communication Paths

The Network architecture had two terminals nodes, i.e. Room 1 and Room 2 which were each a Raspberry Pi. Terminal nodes were then directly communicating with the Thingspeak Server, which was recording the entry and exit times of the MAC

Addresses in a room. This data was pulled by the System Server which was maintaining a local hashtable based on the cloud data. At the same time, the System Server was updating the occupancy of each room to the Thingspeak Server, which was being read by each room, and then based on occupancy, a person was allowed entry into the room or rejected.

As soon as a Person was flagged for having COVID-19, Thingspeak pulled the MAC Address of that affected user and proceeded to trace all the people who came in contact with the said MAC Address. The algorithm considers several conditional cases to finalize the list of users to alert. SMTP is used to send an email alert advising the user to take necessary precautions and a link to the CDC guidelines is included in the email.

C. Contact Tracing Algorithm

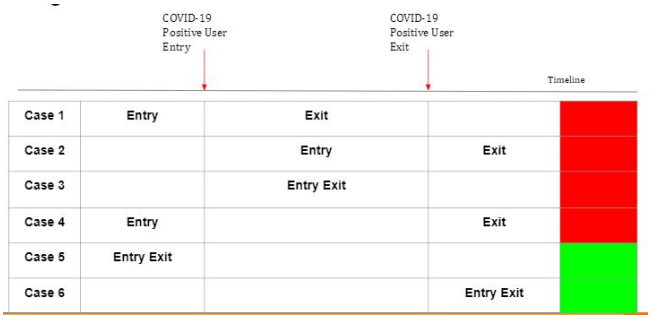

Fig. 5 Contact tracing algorithm conditional cases

The algorithm works on a single input parameter - the MAC address of the COVID positive user. The dictionaries created for each room have a MAC Address, ENTRY time, and EXIT time. The entry and exit times are appended after the MAC Address alternatively, the algorithm assumes that all-time data entries will be even in number for each MAC Address. But this fails in case a user is currently in the room and doesn't have an exit time logged yet. To solve this caveat, we append the current time as the exit time to all odd time entry MAC Addresses. Since we are calculating the contact at the current time, it doesn't interfere with the accuracy.

Once all MAC Addresses (referred to as MAC-id in the report) have an even number of time entries, the algorithm does a stale entry analysis. If the latest entry and exit times are 14 days older than COVID positive users entry and time (Referred to as COen and COex respectively), that MAC-ids are stored in a separate dictionary. When a user is tested positive, his MAC id is used to pick out all his entries and exits for each room.

The algorithm picks the COen and COex and iteratively goes over each MAC-id's entry and exit time pairs. It checks if the user time in the room overlaps with the affected user's time in the room in any of their entries [Fig. 5]. The four cases covered as "In-Contact" are as follows:

- The user enters before COen and leaves before COex.
- The user enters after COen and leaves after COex.
- The user enters after COen and leaves before COex.
- The user enters before COen and leaves after COex.

Following two cases are flagged as "No-Contact":

- The user enters and exits before COen
- The user enters and exits after COex

The algorithm checks if the user entry is after the COex, then it skips all future entries and exits for that MAC id. This doesn't interfere with the future entries and exits of the affected user as the algorithm iterates over each COen-COex pair.

After going through all the COen-Co-ex pairs and checking for all MAC-ids, we subtract the stale entry list from the alerting list.

IV. Modifications From The Proposal

The study did make one modification to the proposal, in the proposal it was mentioned that study would make use of MQTT, but given that the data to be sent was in form of strings, it was found that HTTP is more suitable and reliable for the purpose of this implementation. Apart from that, most of the implementation of this project remained the same and were able to deliver the proposed functionality.

V. Scalability

As the system architecture is based on three components, users and wifi access points, ThingSpeak server, and cloud computing, each aspect is scalable making whole system scaling easy. The demonstration shows Raspberry Pi B+ systems as wifi access points, however to scale, existing wifi routers and infrastructure can be

leveraged. There is a need to modify each router to connect to a centralized server. One major need to implement in real life scenarios is the room entry control system that can be linked to the router of that room. Once we have all the connected users data successfully being logged to the server, the algorithm can be made more efficient to handle large scale users. The demonstration depends on the ThingView app but to make users stay true to the system, an app that allows them to connect to a particular room while entering, see occupancy and see alerts, all in one place would be an efficient addition.

VI. Assumptions

There are some crucial assumptions made to make the demonstration successful on the practical functionality scale. These assumptions would have to be overcome before implementing the system in a real world setting.

When the Raspberry Pi boots, a service called hostapd (Host Access Point Daemon) starts and creates a Wireless Access Point called “Room 1”. When a device connects to this WAP its MAC Address and time will be logged and stored in a file. Then this information is sent to Thingspeak. The ip lease given to each user expires after 2 minutes, so the user has to stay in the room for at least that duration.

The ThingSpeak server also has a 15 second limit between two communications, hence the gap between two users entering the room needs to be minimum of 15 seconds, currently, the user gets a prompt to wait 15 seconds to avoid crowding and coming into contact while entering.

More generic assumptions are that the users will adhere to the protocol and connect to the room wifi access point to request entry. The study also assumes that there are no blind spots in the environment and all areas are factored into proper wifi coverage.

VII. Conclusion

It is an understatement to say that COVID-19 has changed our lives. It has impacted our work lives, our mobility and the way we communicate. It is very important that we have a ready and scalable system that can be implemented on a short notice. The implemented solution clearly shows full functionality to monitor the people density in the area (room occupancy) and ability to trace all the users who were in contact with the COVID positive user in the last two weeks. Furthermore, the cost of implementation on a large scale is still minimum with no cost incurred to the end user. The study successfully demonstrated the usefulness of IoT by creating a full scale system with very basic and existing components.

Acknowledgment

Group GLP would like to thank Dr. Janise McNaire for her continued support and guidance. The division of work for the presented study was as follows:

Hardware implementation - Yogeshwaran Logashanmugam

Cloud and Server Communications - Mayank Gupta

Algorithm and contact tracing - Mohit Patil

The group is also thankful to the whole cohort of IoT-Fall 2020 for enthusiastically supporting the learning environment.

References

[1] L. Garg, E. Chukwu, N. Nasser, C. Chakraborty and G. Garg, "Anonymity Preserving IoT-Based COVID-19 and Other Infectious Disease Contact Tracing Model," in IEEE Access, vol. 8, pp. 159402-159414, 2020, doi: 10.1109/ACCESS.2020.3020513.

[2] Distance monitoring with Arduino & AskSensors IoT. https://hackaday.io/project/162824-distance-monitoring-with-arduino-asksensors-iot

[3] IoT-based Contact Tracing Systems for Infectious Diseases: Architecture and Analysis arXiv:2009.01902 [cs.CY]

[4] Zhang, Weiping & Kumar, Mohit & Yu, Junfeng & Yang, Jingzhi. (2018). Medical long-distance monitoring system based on the internet of things. EURASIP Journal on Wireless Communications and Networking. 2018. 10.1186/s13638-018-1178-2.

[5] ESP8266-based Wearable Corona Distance Monitor,

https://www.espressif.com/en/news/wearable_distance_monitor

[6] https://play.google.com/store/apps/details?id=com.cinetica_tech.thingview&hl=en_US&gl=U