Blockchain based Contact Tracing: A Solution using Bluetooth and Sound Waves for Proximity Detection

ZiXi Hee, Iftekhar Salam

School of Computing and Data Science,
Xiamen University Malaysia, Sepang 43900, Malaysia

Abstract

In the wake of the Covid-19 pandemic, countries and organizations started looking towards technology to curb the spread of the disease, for instance, conducting contact tracing with smartphones. Many contact tracing applications are on the market, built on different technology, such as Bluetooth, GPS, Sound, and QR code scanning systems. The use of sound is an area that has potential for further exploration; currently, only NOVID is utilizing this technology. On top of that, there is a need for a decentralized backend solution that is both public and auditable to address data manipulation concerns. One of the possible solutions is using a blockchain as the backend for the system. We propose a blockchain-based contact tracing solution that uses sound and Bluetooth to detect proximity. Our proposed solution uses blockchain as the backend of the system for decentralized storage of contact tracing data. In the proposed system, close contact is established if both Bluetooth and sound are detected between the communicating devices. The practicality of the proposed scheme is assessed by a performance evaluation of the proximity detection system and a proof-of-concept of the blockchain backend. The results show that the sound-amplitude based distance measurement can estimate whether an encounter is a close contact (within 3 meters) using a 'threshold' of the amplitude. The use of sound amplitude eliminates situations where the usage of only Bluetooth would show false positives. The proposed approach is the first work that integrates Blockchain, Bluetooth and sound amplitude for proximity detection to the best of our knowledge. Overall, the system shows promising results in distance estimation than if only a Bluetooth implementation is used.

Keywords: Digital contact tracing, Blockchain, Bluetooth, Sound-based Proximity Detection

1. Introduction

Covid-19 is a disease that has evolved into a global pandemic outbreak since its first identification in Wuhan of Hubei Province, China, in 2019. As of 20 December 2021, it has infected more than 271 million people and caused approximately 5.33 million deaths worldwide [1]. Since then, countries have been scrambling to contain the spread of this deadly virus. For instance, China has banned flights, closed down access to cities, and even built temporary hospitals within ten days in an effort to combat the spread of the disease [2]. Malaysia has also implemented its own Movement Control Order in addition to screening and isolating at all points of entry to the country [3]. However, the situation seemed to get only worse and worse, as evidenced by the exponential rise in case numbers. As the virus continued to spread, countries and companies started to look at publicly accessible technologies, more particularly smartphones, as a means of an accessible way to curb the spread of the virus.

There are several contact tracing methods used on the market; one of the most widely used is proximity contact tracing. Proximity contact tracing works by utilizing technology such as Bluetooth to detect the 'proximity' between two smartphone users. When a user is infected, they will upload locally stored 'identifiers' to a database. Other users can then compare their local identifiers with those in the database to determine if they have ever been in close contact with the infected user.

Email addresses: `swe1809649@xmu.edu.my` (ZiXi Hee), `iftekhar.salam@xmu.edu.my` (Iftekhar Salam)

There are two major models to store these data collected from contact tracing, centralized and decentralized. The centralized model requires the application to upload the data received on their phone onto a central server. This introduces complications such as privacy and data manipulation concerns by a central authority [4]. This is where the decentralization model comes into play. Instead of relying on a central server, the decentralized model may rely on blockchain to store the data in a distributed and auditable ledger so that no party can change or modify the data once it is uploaded onto the blockchain.

1.1. Problem Statement

When using Bluetooth, the received signal strength indication (RSSI) is used as a proxy for calculating and evaluating the distance between two devices. The core concept for these applications is periodically transmitting and exchanging anonymized tokens (around every 100 – 270ms, and changing the anonymized token every 15 minutes.). These tokens work by not containing any information about the phone or the phone owner, and each phone stores the sent and received tokens. Although it was stated to use less energy, there have been several studies [5, 6] that have identified the limitations with using only Bluetooth as the medium for proximity-based contact tracing. These limitations include procuring a high number of false positives due to Bluetooth signals being able to penetrate walls easily, and inaccuracy of distance as walls and furniture may affect the signal strength in transmission when compared with free space. Therefore, some implementations on the market have started exploring alternatives such as sound waves to do proximity contact tracing.

Major adopted solutions on the market use a centralized server to store Covid-19 related data. However, this introduces privacy concerns such as manipulation of data, lack of transparency for using the data, and privacy concerns for individuals. This is why a decentralized, auditable and public solution is needed. Although some implementations have proposed using this technology, the technology as a whole is still in its early stage. Therefore, this work will provide a proof-of-concept for a blockchain backend solution for contact tracing purposes. On top of that, an application that uses blockchain as its backend while also utilizing sound waves to detect proximity is not available on the market. Therefore, this work attempts to address the gap by developing a blockchain-based contact tracing solution that incorporates sound and Bluetooth for proximity detection.

1.2. Contributions

We proposed a sound and Bluetooth assisted blockchain-based proximity contact tracing that improves the accuracy of currently available solutions. Specifically, the interest in using sound and Bluetooth for proximity detection focuses more on how these components integrate into a system. In addition, the usage of a blockchain backend focuses on providing a proof-of-concept for a decentralized backend public database.

In the context of this work, Bluetooth is used to exchange information and perform handshakes between two devices that have detected each other. On the other hand, the sound is used for proximity detection by approximating the distance between the two devices. Two techniques for sound proximity detection have been explored: the time difference based distance calculation method and the sound amplitude-based distance calculation method. However, the time difference based distance calculation provides varied preliminary testing results; therefore, it is not explored further beyond initial testing and implementation. For the sound amplitude-based distance calculation, although an accurate distance measurement cannot be done due to variation in hardware, an approximation of whether the phones are far or near from each other can be made by defining a threshold for the measured sound amplitude. The backend of the proposed system uses blockchain to support decentralized storage of contact tracing data. The implementation of blockchain is done as a proof-of-concept, with the primary goal of achieving functionality in aspects such as creating a block, performing consensus algorithm, syncing across nodes and uploading and downloading data. Overall, the proposed solution improves contact tracing accuracy and prevents manipulation of contact tracing data.

2. Overview of Contact Tracing

Contact tracing, as described by the World Health Organization (WHO), is a process of "identifying, assessing and managing people who have been exposed to a disease to prevent onward transmission" [7]. A person that has been diagnosed with a disease, in this case, Covid-19, will be asked to conduct an interview with a health inspector. They will be asked questions such as the places they visited, encounters from recollection from the last 14 days. The inspector will then assess all the potential contacts and notify them for further actions, such as advising to quarantine or taking a test [8]. These actions allow the chain of transmission for a particular virus to be stopped as fast as possible.

However, traditional contact tracing solutions have their limitations. Being a slow and tedious process, it decreases the reaction time of authorities towards the spread of the virus. This manual process also heavily depends on the individual's memory and honesty when being interviewed [9]. On top of that, published models also find that manual contact tracing does not scale well when the pandemic grows beyond its early phase due to the limit of personnel required to carry out the process [10]. To overcome this problem, digital contact tracing is now being explored to automate and increase the efficiency of this process.

Digital contact tracing is mainly accomplished with smartphones, a technology now ubiquitous and widely adopted by most societies. Digital contact tracing integrates different technologies in smartphones such as Bluetooth and Global Positioning Systems (GPS) to approximate the proximity and duration of a person's exposure to surrounding individuals. If an individual gets exposed to a deadly virus (for instance, Covid-19), the person will be alerted of potential exposure, allowing them to take relevant action such as referring to the health department of their area or proceed with self-quarantine [11]. Digital contact tracing solutions on the market can be divided into several types. We discuss the location based, mobile operator based, QR code based, and proximity-based contact tracing below, as they are among the most common ones used.

2.1. Location Based Contact Tracing (LCT)

Location based contact tracing is done by recording the absolute locations of smartphones. This is usually done using GPS or Wi-Fi access points that provide location data to a smartphone [12]. An example of this application is Iceland's Rakning C–19 application, which collects GPS data in the background to build a trail of where its users have been [13]. Another application that also uses location data is India's Aarogya Setu [14], which has become a mandatory app for government and private sector employees. However, it has raised privacy concerns from critics, experts and groups ever since its inception. For instance, the Software Freedom Law Centre states that the government can share the data to anyone that the government deems necessary, although such claims have since been denied [15].

2.2. Mobile Operator Based Contact Tracing (MOCT)

Mobile operator based contact tracing solutions exploit the readily available infrastructures deployed by mobile operators. Examples would include base stations of cellular networks that are connected to smartphones. They function similarly to location based contact tracing in the sense that they also do contact tracing through the location of users. The drawbacks of this method include low accuracy and privacy risks for individual users. Therefore, this solution is generally used to evaluate the impact of lockdown measures and detect hotspots for contagious diseases rather than actually doing contact tracing. This solution is adopted in Israel during the Covid-19 pandemic [12].

2.3. QR Code Contact Tracing (QRCT)

QR code contact tracing employs two forms of contact tracing, symptom tracking and location/venue tracing. Symptom QR code contact tracing uses colour coded QR codes to indicate the health status of a user, mainly orange and green. These QR codes are used as electronic certificates. They include information like contact tracing, exposure risk, health appointments of an individual. They are used in Fujian China, where citizens are required to show the QR codes when carrying out public activities such as taking public transport, entering/exiting schools [16].

The second form is location based QR contact tracing. A unique QR code will be given to places such as stores, offices, factories and public transport. Residents who visit those places have to scan the QR code on the premises to register their visit automatically. Thus, if cases for a virus are detected on-premise down the road, other users can approximate their risk index based on the places they have checked in. A clear example of this implementation would be the MySejahtera application used in Malaysia [17].

2.4. Proximity-Based Contact Tracing (PCT)

Proximity contact tracing solutions perform contact tracing by detecting the relative positions of smartphones in an area. This type of contact tracing is also the most widely adopted among the solutions discussed. Most solutions on the market use Bluetooth Low Energy (BLE) technology and exploit the Blue-Trace protocol [12]. Smartphones that get near each other exchange different information that varies across implementations. Examples of information exchanged include the phone's mac address, the start and end time of interaction, the strength of the signal received, or a randomly generated pseudo-identifier. Another type of technology used is ultrasound technology, but this is less common in the market. NOVID is one of the implementations that utilizes the time–of–flight of sound to calculate the distance between two smartphones [18]. This is done to more effectively prevent false-positive triggers for an object that is far away to be registered as an object at close proximity.

When a user is diagnosed with a disease, for instance, Covid-19, they broadcast or share transactions containing their health status updates alongside their Bluetooth randomized addresses generated over the past two weeks. The reason for a time frame of two weeks is because older information is deemed not useful, as the incubation period of Covid 19 is at most 14 days. Then, other users would go on to the database to retrieve the new records that have been uploaded. They would compare their local copy of their exchanged tokens to verify if they had been in contact with the infected user. NOVID improves upon this system by also showing the user how 'far' the user is from a disease, in other words, how many contacts away they are from a person that has been diagnosed with a disease [19].

2.5. *Models of Data Storage for Contact Tracing Solutions*

There are two major models to store data collected from contact tracing, centralized and decentralized. For centralized databases, there is usually a central server owned by one or several authorities for information exchange, whether it is used as a bulletin board for users to check their risks or for infected patients to upload their data onto the server. This introduces complications such as privacy and data manipulation concerns by a central authority, where it is possible to reverse data that is uploaded onto the database [20] or alter it to the needs of a specific party [4]. Therefore, this is where a decentralized database implementation, particularly blockchain, comes into play. Instead of relying on a central server, the decentralized model relies on blockchain to store the data in a distributed and auditable ledger so that no party can change or modify the data once it is uploaded onto the blockchain.

2.6. *A Review of Existing Contact Tracing Implementations*

There are several contact tracing solutions on the market. This work briefly reviews nine solutions, namely, Pronto-C2, BeepTrace, NOVID, Privacy-Preserving Contact Tracing launched by Google Inc. and Apple Inc., TraceTogether, PriLok DP3T, MySejahtera and Fujian Bamin Health QR. When using Blockchain for contact tracing, both Pronto-C2 and BeepTrace use the blockchain network as a bulletin board, where users can check the proximity contacts that are uploaded by an infected patient [21, 22]. NOVID is the only existing implementation that uses sound and Bluetooth for proximity detection. Among others, Google and Apple's CT, TraceTogether and DP3T utilize Bluetooth for proximity detection and centralized storage for their backend server. PriLok is one of the solutions that utilizes existing mobile operator infrastructure to do contact tracing. MySejahtera and Fujian Health QR both use QR codes to do contact tracing, albeit with slight differences in their use cases. Table 1 shows an overall comparison of the contact tracing solutions currently available on the market.

Note that in Table 1, the centralized and decentralized refer to the type of database that is implemented in the protocol/application. It does not reflect the architecture of the application itself. When examining Table 1, it is worth noting that there is no decentralized solution that incorporates Bluetooth and Sound to do proximity-based

Table 1: A comparison of available contact tracing solutions

Name	Method	Technology	Data Storage
Google & Apple's CT [23, 24]	PCT	Bluetooth	Centralized
TraceTogether [25]	PCT	Bluetooth	Centralized
BeepTrace [21]	PCT & LCT	Bluetooth, Wi-Fi, GPS, Cellular Tower	Decentralized
Pronto–C2 [22]	PCT	Bluetooth	Decentralized
PriLok [26]	MOCT	Cellular network	Centralized
DP3T [27]	PCT	Bluetooth	Centralized
MySejahtera [17]	QRCT	QR code	Centralized
Fujian Health QR [28, 16]	QRCT	QR code	Centralized
NOVID [19]	PCT	Bluetooth, Sound	Centralized

contact tracing. Therefore, this work attempts to address this gap by providing a proof of concept for a decentralized blockchain-based contact tracing solution that uses sound and Bluetooth for contact tracing.

3. Proposed Framework

This section introduces the framework for the implemented system and the workflow taken in implementing the system. This includes discussing the entities that make up the framework, the on-device proximity detection framework, the distance measurement method used in the framework, the confirmed case workflow framework and the data structures used by the framework.

3.1. Framework for Contact Tracing

Several entities are involved in this contact tracing framework: users, the diagnostician, the authority, and the blockchain network. Figure 1 illustrates the relationship between each entity involved in the framework. The interaction among the entities is partly based on the BeepTrace architecture.

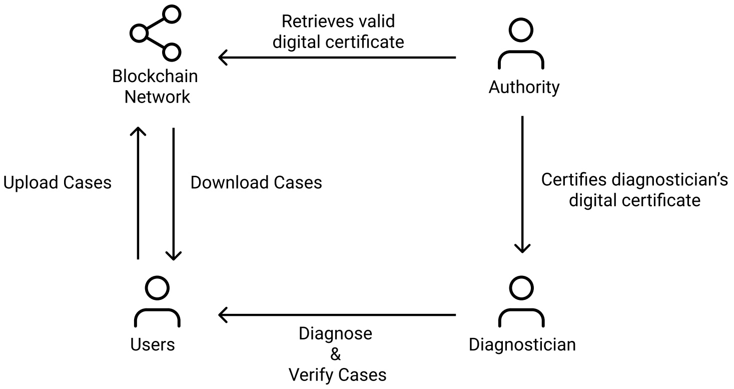


Figure 1: An illustration of entities in the proposed framework

- Users: This is the entity that will be using the application and the framework. The users will walk around with the application and exchange close contact identifiers. If they become diagnosed with Covid-19, they will upload their close contact identifiers to the blockchain network. Other users can download identifiers to compare with their local database to determine their risks of infection.

- Diagnostician: This entity will certify that a user is infected with Covid-19. After diagnosing a user with Covid-19, they will digitally sign the close contact cases on a user's smartphone using their private key before uploading it on the blockchain.

- The authority: This entity is responsible for certifying the digital certificates of diagnosticians. The digital certificates will allow diagnosticians to digitally sign close contact cases using their private key for a confirmed patient. This is done to prevent users from uploading fake cases onto the blockchain and allow uploading only verified cases. This structure is similar to the one proposed in BeepTrace.

- Blockchain network: The backend consists of decentralized nodes holding all the uploaded close contact data. The nodes that receive new data will create a block and use an agreed consensus algorithm to embed the block onto the blockchain. When the nodes receive a case uploaded by a user, they will check if it is signed off by a valid certificate.

3.2. *On-Device Proximity Detection Framework*

The overall proposed framework uses sound waves and Bluetooth to do proximity detection. An illustration is provided in Figure 2. The contact tracing application will be running in the background of the smartphones for users that have installed the application. When two individuals come into close contact, the constantly discovering and broadcasting Bluetooth signals will detect and identify each other through periodically generated identifiers. These identifiers are refreshed after a period of time to ensure that no other identifier can be linked to a particular user, preserving their anonymity in the process.

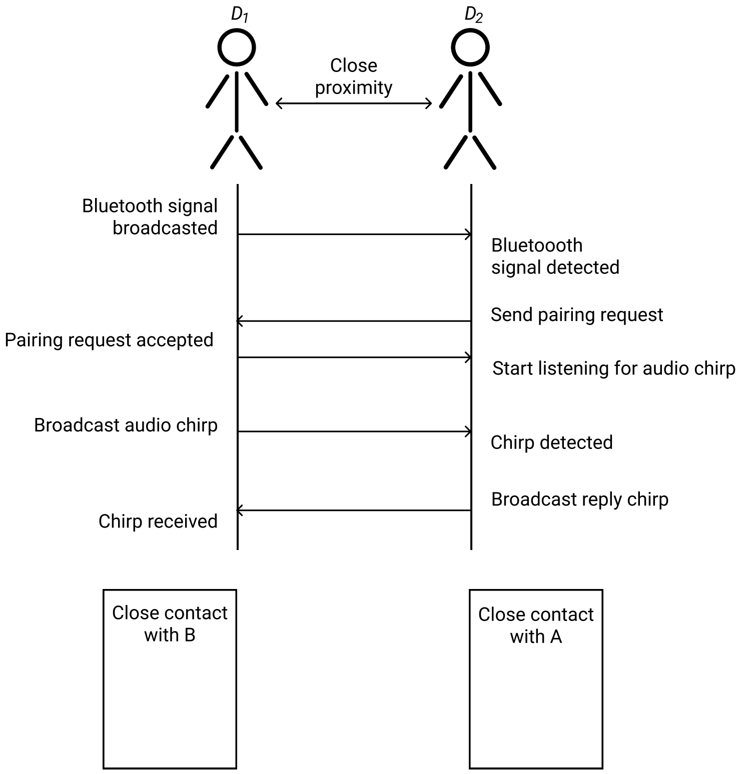


Figure 2: An illustration for the proximity detection process between two phones held by individuals that came close to each other

First, a smartphone, D_1 will advertise itself to other existing application instances. When another smartphone, D_2, detects the advertisement, it will send a pairing request to D_1. After D_1 accepts the pairing request and returns a confirmation, D_2 will start listening for an audio chirp. On the other hand, D_1 will broadcast an audio chirp after a predetermined delay. If D_2 detects the audio chirp, it will proceed by broadcasting a return audio chirp to D_1 and recording the interaction that has occurred in its local database. D_1 will also record this interaction when it detects the return chirp. After the whole exchange has been completed, or if the interaction has proceeded longer than a predefined period of time (indicating that the smartphones cannot hear each other), the connections will be terminated. These smartphones will then stop broadcasting the audio chirp for a specified amount of time even if they detect each other using Bluetooth; this is to allow the other smartphones to participate in the tracing network.

3.3. *Distance Measurement with Sound*

The use case of sound for proximity tracing has not been explored much, especially in the world of blockchain-based proximity tracing. There are two main ways of doing distance detection using sound, mainly time difference proximity tracing and sound amplitude proximity tracing [6]. Our implementation explores both of these approaches in the sections below.

3.3.1. Time Difference Based Distance Calculation

This approach uses the time difference between the chirp of the broadcaster's phone and the receiver's phone. The distance can be calculated by using the below relationship:

$$\text{distance} = 343m/s \times \frac{(\text{total time} - \text{waiting time} - \text{internal delay})}{2}, \tag{1}$$

where total time is the time when the chirp is broadcasted and the time the phone receives the chirp, and the waiting time is the time delay to wait agreed by both phones [29]. The overall approach is illustrated in Figure 3. The first smartphone will start recording the time and send out the signal. The internal delay time is the time between when the function is called and when the signal is played. After the second smartphone receives the signal, it will wait for a duration before sending the return signal. This waiting time factors in the internal delay for playing and listening for the signal. When the first smartphone receives the signal, there will be an internal delay between when it is heard and recorded. The time for the signal to travel in the air is recorded by subtracting the internal delay and the waiting time.

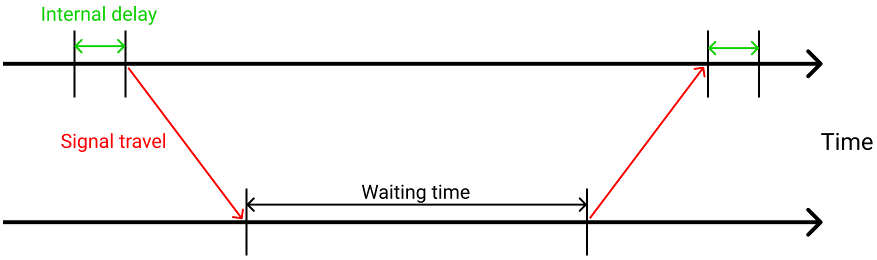


Figure 3: Illustration of "Ping Pong" system

3.3.2. Sound Amplitude Based Distance Calculation

The sound amplitude approach for distance uses the fact that sound pressure decreases as it travels in the air. The relationship for estimation of distance [6] can be written as:

$$\text{distance} = 10 \times \frac{(\text{power of signal at 1 meter-signal strength received})}{10 + n}, \tag{2}$$

where n is the signal propagation constant. This solution uses a 'chirp' around 2kHz – 6kHz with an amplitude of approximately 20db.

3.4. Workflow for Confirmed Cases

The overall workflow for when a patient is diagnosed with Covid-19 is illustrated in Figure 4. When a patient is diagnosed with Covid 19, the smartphone generates the hash of the close contact records and the user's public key to be uploaded to the blockchain. The hash will be sent to the diagnostician for signature. This is done to ensure that no actual data is transferred to the diagnostician but only the hash of the data to preserve privacy.

The time frame for which data will be uploaded is all records within two weeks from the date that the patient is diagnosed. This is because the virus can take up to 14 days to show symptoms; thus, the patient may have been infected for 14 days but has just started to show symptoms.

After the hash is generated, the diagnostician will digitally sign the hash with their private key. This is done to ensure the records are uploaded by a person that has been diagnosed with Covid-19. After the hash is signed, the patient will sign the case with their private key before uploading them to the blockchain. This is done to ensure the case has not been tampered with when in transmission.

The application will periodically ping the blockchain to retrieve the latest reported cases for an average user. After receiving the cases from the blockchain, they will check if their identifier is uploaded onto the blockchain. The presence of matches indicates that the user has a close contact occurrence; thus, they should take appropriate measures as their local government advises.

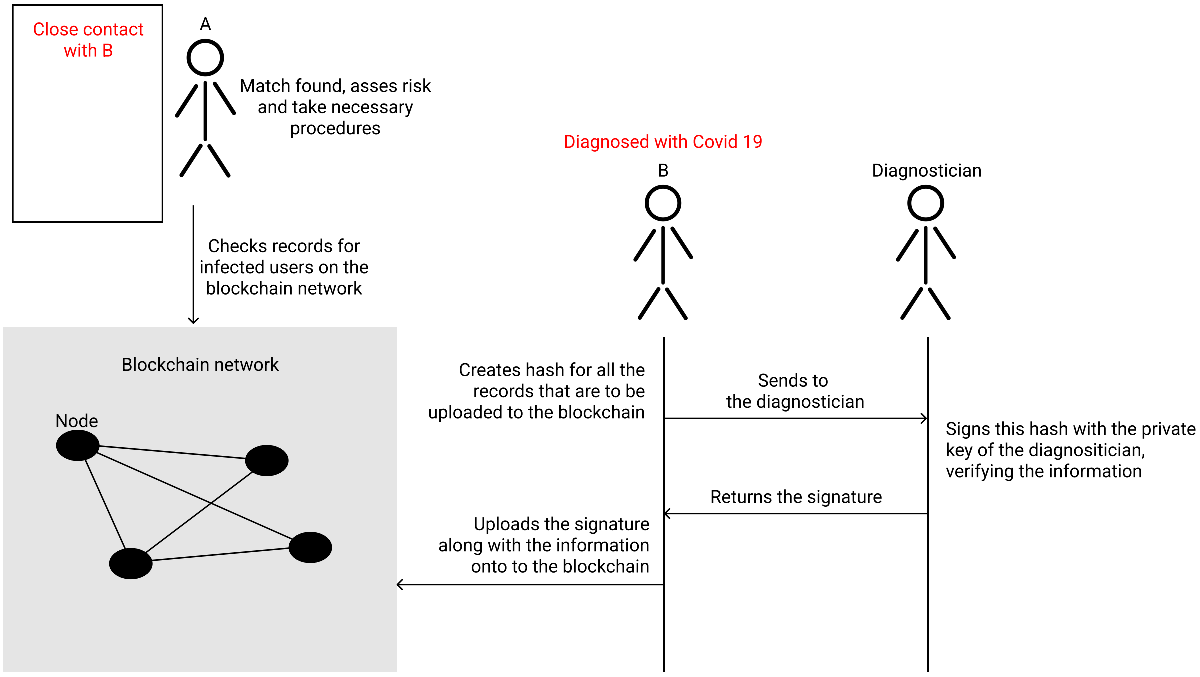

Close contact with B
A
Match found, asses risk and take necessary procedures
Checks records for infected users on the blockchain network
Blockchain network
Node
Diagnosed with Covid 19
B
Diagnostician
Creates hash for all the records that are to be uploaded to the blockchain
Sends to the diagnostician
Signs this hash with the private key of the diagnositician, verifying the information
Returns the signature
Uploads the signature along with the information onto to the blockchain

Figure 4: An illustration for the workflow when a patient is diagnosed with Covid-19

3.5. Data Structures

The data structure for communication between the blockchain can be split into three main parts, the public key, the ledger (where the main information recorded by the blockchain is stored) and the signature. The signature is signed by the patient using their private key to verify that the ledger is not tampered with when in transmission. The ledger comprises three parts: the device identifier, the close contact records that the patient has experienced, and the signature by a diagnostician to confirm the validity of the record.

Each close contact record contains:

(1) The close contact identifier exchanged when two devices come close to each other,
(2) The date when the close contact encounter was recorded,
(3) The estimated distance between the users when this close contact was recorded,
(4) The medium of detection (either Bluetooth or Sound),
(5) Estimated duration of contact for this encounter.

4. Implementation of the Proposed Framework

This section provides a detailed implementation for the Blockchain-Based Contact Tracing Solution with proximity detection using Bluetooth and Sound. In this implementation, we constructed the system as a proof of concept, where the main goal is to carry out the desired functions. It is worth noting that the pseudocodes provided in this section are simplified versions of the implemented code, as there are multiple data structure conversions and checks required when passing and handling data. To avoid over-complicating the presentation of the idea, pseudocode is ultimately chosen instead of the actual code. Before going into the implementation details, we first list an explanation for notations used in the proceeding sections below.

- Arr: A non-constant sized growable array
- Rnd.nextInt: Get the next random number from a random number generator

- Range(N): Gets an array of numbers ranging from 0 to N, differing by 1 each
- F_1: Frequency of sound signal used in sound proximity detection
- M_w: Mining worker, a thread running that is tasked with mining a block of data using a predefined consensus algorithm, i.e. proof of work
- API: Short for Application Programming Interface. An API is a software intermediary that allows two applications to talk to each other
- L_1: Length for randomly generated identifier used in proximity detection
- *characterPool*: A pool of characters consisting of alphabets, numbers and symbols that will be used to generate unique identifiers for the contact tracing process
- B_1: A new block that is received from the pool Node
- Δ: Upper and lower range for frequency identification used in sound proximity detection. A range is provided to account for the possible variation as a result of environment or hardware differences between devices.

4.1. System Setup

To setup a node for the blockchain backend, the user must first define nodes that it will connect to. This is done so that the user node can propagate the proof of work that has been done, and at first launch, retrieve the longest current chain on the blockchain by requesting it from other nodes. For a local setup, which is used in this proof of concept, the user needs only to define the port number for a specific node. Besides defining the neighbour nodes, the port number has to be unique from the others for a local setup. For future remote node setup, the port number can be constant. After defining the parameters of its neighbours and itself, the user can run the node. The program will retrieve the latest chain and initialize all required parameters to add itself to the blockchain network.

4.2. Blockchain Nodes

There are two types of blockchain nodes, the pool node and the miner node. The pool node is mainly tasked with accepting new entries that will eventually go into the blockchain. The miner node, on the other hand, is assigned with 'mining' the block that will go into the blockchain. Algorithm 1 shows the algorithm for setting up a blockchain node when it is first deployed.

Algorithm 1: Blockchain node setup

```
Read list of neighbour nodes from a provided file
for each neighbour node do
    Request for latest chain through the API
    if all neighbour nodes have been found then
        Remove duplicate chains
        Save the longest chain
Start listening for API calls
if a new block, $B_1$ is received then
    Create new mining worker, $M_w$
    Assign $B_1$ to $M_w$ and begin mining process
    if $M_w$ finishes mining the block then
        Add block to chain
        Broadcast the current longest chain to neighbours
if a chain is received then
    if the received chain is longer than local chain then
        Terminate running $M_w$
        Replace local chain with longer chain
        Broadcast longest chain to neighbours
```

The API calls for accepting a new block from the pool node and accepting longer chains from other nodes are also defined accordingly. For the pool nodes, in release implementation, the miner nodes should request blocks from the pool. However, in this proof of concept, the pool node will transmit new blocks instead of the other way around. Algorithm 2 illustrates the algorithm for a pool node when a new case is received from the contact tracing application.

Algorithm 2: Pool node when a case is received

```
if case has not been in the pool or blockchain before then
    Add case to pool
if amount of pool cases is larger than single block size then
    Prepare new block
    Broadcast new block to miners
    Remove cases from pool
```

4.3. Unique Identifier Generation

Algorithm 3 shows the process for generating a unique identifier used for close contact detection. The smartphone application will generate a unique identifier by selecting characters from random numbers and letters from a predefined pool. This unique identifier will be regenerated within every set time period and saved in the smartphone. The length of the unique identifier is ten characters in this project. However, this can be increased to avoid collisions with generated identifiers between devices.

Algorithm 3: Unique identifier generation

```
Input: Length of random identifier, $L_1$
Output: A random identifier of length $L_1$
characterPool ← 'AaBbCcDd...”
for i in range($L_1$) do
    Arr(characters[Rnd.nextInt])
    return array in string form
```

In this proof of concept, this unique identifier will be generated every time the application relaunches; this is to allow for testing. In a release, this code will run only after a predefined time, for instance, every 15 minutes, and this identifier will be stored locally in the database along with the time stamp when it was generated. After this unique identifier is generated, the application will begin the proximity detection process in the background.

4.4. Proximity Detection Process

The proximity detection process is broken into two stages: the Bluetooth and the sound detection stages. The Bluetooth detection will be tasked with establishing a connection with the other device. After establishing the connection, the sound detection process will be initiated to provide a more accurate distance measurement. This process is shown in Algorithm 4.

The device will look for sounds from a specific frequency to prevent any sound from triggering a false positive. There will be a leniency to the frequency observed to account for some variation, Δ, resulting from environmental or hardware differences between devices. The leniency is determined to be 40 in this project with a frequency of 5040Hz. After the frequency is detected, the distance between the two devices will be estimated. The device is deemed too far away if a sound is not heard.

Algorithm 4: Bluetooth connection establishment algorithm

```
Get permission for device sensors
Start discovery and advertisement of Bluetooth signal
if Bluetooth signal is discovered then
    if identifier is in local database then
        if has done proximity detection with it using sound before then
            return
        else
            Add this identifier to database
    Request to establish connect through Bluetooth
    if connection is established then
        Start listening for sound signal
        Disconnect after predefined time
if receiving connection request from another device then
    if identifier not in local database then
        Add identifier to database
    Accept incoming connection
    Broadcast sound signal
    Disconnect after predefined time
```

Algorithm 5: Measuring internal delay of device

```
for i in range(3) do
    Start timer
    if Sound detected then
        if sound heard is within range of $F_1 + \Delta$ to $F_1 - \Delta$ then
            Stop timer
            Record elapsed time
    Play sound signal
Calculate average elapsed time
```

4.4.1. Time Difference Distance Measurement

Although time difference distance measurement is not used in the final implementation, it is explored as a possible solution to the distance measurement using sound. Algorithm 5 shows the process to measure internal delay in the device. At the launch of the application, the device will do internal calibration by playing a signal and listening to itself three times. This is done to measure internal delay. After a connection is established using Bluetooth, the devices will begin the time difference distance measurement process using frequency, F_1 = 5040Hz. Algorithm 6 shows the time difference process for the broadcasting device. A similar process can be followed to estimate the distance in the

Algorithm 6: Time difference distance measurement for broadcasting device

```
Start timer and broadcast signal at predefined frequency, $F_1$ for a certain time
Listen for return signal after predefined time
if sound heard is within range of $F_1 + \Delta$ to $F_1 - \Delta$ then
    Stop timer
    Estimate distance between two devices
```

listening device.

4.4.2. Sound Amplitude Distance Measurement

Algorithm 7 shows the sound amplitude-based distance measurement process for the broadcasting device. After broadcasting a signal for a predefined time, the broadcaster will listen for a return signal. If a return signal is heard, it will estimate the distance between two devices and record it.

Algorithm 7: Sound amplitude measurement for broadcasting device

```
Broadcast signal at predefined frequency, $F_1$ for a certain time
Listen for return signal after predefined time
if sound heard is within range of $F_1 + \Delta$ to $F_1 - \Delta$ then
    Estimate distance between two devices (refer to the next section)
    Record proximity detection details in local database
```

Algorithm 8 shows the sound amplitude-based proximity detection process for the listening device. The listening device will be looking for the signal sent by a broadcasting device. If a signal is heard, it will broadcast a return signal and estimate the distance between the two devices.

Algorithm 8: Sound amplitude measurement for listening device

```
Listen for signal
if sound heard is within range of $F_1 + \Delta$ to $F_1 - \Delta$ then
    Broadcast return signal after predefined time
    Estimate distance between two devices (refer to the next section)
    Record proximity detection details in local database
```

4.5. Distance Estimation Using Sound

The preliminary testing of the time difference based distance measuring shows high enough variation and would require a much longer time to do per device calibration. We found that variations in the scale of milliseconds may affect the result by a considerable margin; thus, requiring much more precision that could only be achieved through native programming. Therefore, in our work, we mainly estimated the distance using the sound-amplitude based measurement by defining a threshold of the sound loudness. Algorithm 9 shows the process for distance estimation implemented in the application.

Algorithm 9: Distance estimation using sound

```
Measure signal loudness
if $loudness > 70\ dB$ then
    Distance = 'Near'
else
    Distance = 'Far'
```

For distance estimation, accurate distance measurement seems not feasible due to the inaccuracy of sound loudness measurement as a result of hardware variation between devices and environment variables (outdoors, indoors having variation in echoes and air density). However, an estimation of distance can be measured based on preliminary testing. The algorithm uses a threshold-based method to measure close contacts. Sound amplitude detected larger than 70 dB is deemed near and far otherwise.

4.6. *Blockchain Synchronization Process*

The blockchain sync process is divided into two parts: downloading the latest cases from the network and uploading a case to the network. The method of downloading the latest cases is relatively straightforward, where the phone will send a request to a blockchain node and request the current chain that it has. After downloading the chain, the smartphone will compare its identifier with the cases on the chain. The process is described in Algorithm 10.

Algorithm 10: Latest cases download from blockchain network

```
Send get request to blockchain node API
for local identifiers do
    if match is found then
        Display matches to user
```

Due to the limitations in the amount of hardware available and to enable testing for this feature, in our testing, the user will enter an identifier that will be located in the blockchain. In a release implementation, the identifiers to be located will be retrieved from the local database that stores all generated identifiers used for proximity tracing.

For the cases upload process, after the user is identified as a Covid-19 patient, they would have to get a verified diagnostician to sign the hash of their cases before uploading it onto the blockchain. After the cases are signed, the signature will be packaged with the close contact cases to be uploaded onto the blockchain. The process is described in Algorithm 11.

Algorithm 11: Case upload to blockchain network

```
Retrieve close contacts within two weeks from current date from local database
Hash the data
Send hash to diagnostician
if diagnostician returns signed hash then
    Attach with close contacts to create new case to be uploaded
    Sign case with own private key
    Attach case with own public key
    Upload the data onto a pool node
```

4.7. *Development Environment*

The proposed implementation of the blockchain network is implemented on a Windows 10 Home (64-bit) laptop running on an Intel® Core™ i5-10500H CPU @ 2.50GHz, supported by a 16GB total random-access-memory (RAM), and an NVIDIA GeForce RTX 3050 Max-Q (Laptop GPU) graphics card. The proposed application is running on two phones. The first is a OnePlus 7 running Pixel Experience 11.0 with a Snapdragon 855 clocked at max 2.84GHz, supported by 8GB RAM. The second smartphone is a POCO F3 5G running MIUI 12 with a Snapdragon 870 5G clocked at max 3.2GHz, supported by 8GB RAM. The programming languages used in this project for the implementation are Dart for developing the Flutter application and JavaScript in the form of Typescript for developing the Node.js backend blockchain network.

Both the pool node and the miner nodes are built on Node.js, which traditionally is a single-threaded programming language. However, since Node.js v10.5, the ‘worker-thread’ package was introduced to enable multi-threading in a node server. This is relevant to the implementation, as a blockchain node has to simultaneously mine a block and listen for propagation from another node that might have arrived at the ‘proof’ for the block being mined. If a longer chain was received and validated, the mining process would be stopped immediately, as it is now considered invalid, given that the received chain can be reasonably expected to have propagated across the network. Listening is handled by the ‘Express.js’ package, providing an application programming interface (API) for nodes to communicate with each other.

5. Testing and Evaluation

This section assesses the effectiveness of the proposed sound proximity detection solution. First, we discuss the testing criteria and reasoning behind the proposals, followed by the testing methodology. Next, we discuss the testing results. Last but not least, we also discuss the features of our sound-based proximity contact tracing solution compared to the different solutions proposed in the existing literature.

The sound-based proximity contact tracing implementation is evaluated based on whether it improves the existing Bluetooth solution that is widely used in the market. To do this, we inspect the number of false positives that occur for situations when close contact should not be recorded. For instance, when the users are on a different floor, in a separate room, or at a distance where the virus is unlikely to be transmitted to a user.

5.1. Testing Methodology

Initially, the approach utilized is the time-of-flight approach similar to what's implemented in NOVID. However, there were several drawbacks when implemented, namely the non-deterministic internal delay caused by internal smartphone processing. This behavior is also supported in the work of Nguyen, Luo & Watkins [6]. Through testing, it is found that the delay is significant enough (with a difference from each measured value by up to 100ms through the testing of this work, and 180-300ms from Nguyen, Luo & Watkins' testing) that it would affect the final result by a significant amount (up to around 30 meters). It is worth noting that this result is obtained even after internal calibration for the internal delay time. The smartphones in the preliminary test are placed 1 meter apart. Table 2 illustrates the results from initial testing for the time difference calculation. The maximum estimated distance is 34 meters, while the minimum estimated distance found is 7 meters. The variations in readings between each are large enough that this approach was not considered further for this implementation.

Table 2: Preliminary testing for time difference calculation

	Time for signal to travel in air (ms)	Estimated Distance (m)
Test 1	78.112	13.40
Test 2	199.993	34.30
Test 3	63.326	10.86
Test 4	45.246	7.75

Due to these large variations, the time difference approach is not further explored in this work; instead, the sound amplitude method is investigated further. Tests were repeated with the same smartphones for sound amplitude-based proximity detection, a POCO F3 5G and a OnePlus 7. The two phones were placed on two chairs and configured to use a frequency of 5040Hz. The contact tracing app was then started on the two phones and left running for a predefined time (1 minute) or until a close contact was recorded. Tests were repeated three times each, and the results were recorded. Figure 5 illustrates the way the testing is carried out. Tests were conducted at three different locations, including a confined room, indoor space and outdoor space. This set of tests is performed to determine how the ambient sound and type of space affect the results of sound detection by the application.

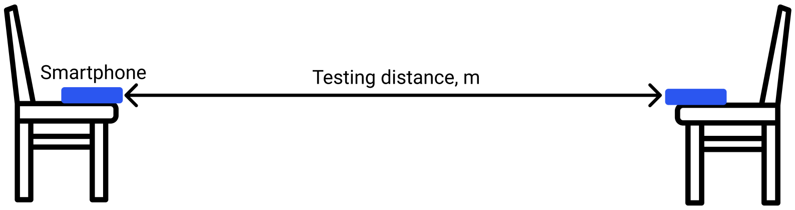


Figure 5: Testing methodology for proximity detection

The results recorded are either 'Near' or 'Far'. 'Near' indicates a close contact has occurred, while 'Far' indicates that a close contact has not occurred. The Centers for Disease Control and Prevention (CDC) recommends practicing social distancing around 6 feet (around 2 meters) between personnel [30]. To ensure that 2 meters produce a 'Near' result, the application is calibrated to show a 'Near' result around 3 meters based on preliminary testing. In comparison, larger than 3 meters will produce a 'Far' result. Tables 3 and 4 show the preliminary testing results for smartphones place in various distances.

Table 3: Preliminary testing for distance measurement in confined room

	First Smartphone Reading			Average	Second Smartphone Reading			Average
1m	80.930	83.190	70.987	78.37	78.530	75.439	79.271	77.75
2m	76.359	72.650	76.664	75.22	76.4944	69.902	76.395	74.26
3m	70.932	72.865	75.873	73.22	76.944	67.627	68.069	70.88

Table 4: Preliminary testing for distance measurement in indoor space (Corridor)

	First Smartphone Reading			Average	Second Smartphone Reading			Average
1m	79.276	76.834	74.989	77.03	81.356	75.870	79.506	78.91
2m	77.784	73.639	75.977	75.80	77.892	69.214	79.491	75.53
3m	75.008	75.266	74.564	74.95	69.654	76.231	71.576	72.49
5m	72.109	77.114	73.437	74.22	66.858	73.151	73.069	71.69
15m	71.024	65.165	67.37	67.85	57.847	60.419	63.183	60.48

Due to variations in readings on both smartphones, the actual distance measurement was forgone for this project. Instead, a loudness 'threshold' is set, whereby any loudness below the threshold is deemed not a close contact. In contrast, any loudness above the threshold is considered a close contact.

The threshold for determining a close contact is chosen to be 70 decibels (dB). Sound loudness that is below 70 decibels will be considered 'Far', while the loudness that is above 70 decibels will be regarded as 'Near'. This is chosen based on the minimum sound loudness required for distances below 3 meters to be deemed a close contact. The minimum average reading seen in the preliminary testing for 3 meters is 70.88 decibels; thus, 70 decibels is chosen.

The blockchain is tested for its functionality with requests from smartphones for blockchain testing. The smartphone uploads its data to the blockchain, and the output is observed. The expected output is that the nodes will compete by performing proof of work when the block size is reached. When one of them finds a valid block, the block will be transmitted across the network, and the other nodes will terminate their current work. The next functionality test involves downloading cases from the blockchain and matching identifiers locally on the smartphone. The expected output is that the smartphone should retrieve the cases from the blockchain and compare them with the identifier to locate them.

5.2. *Testing Results*

This section focuses on the testing results for the implemented system. We first discuss the testing results for the sound-based proximity detection system in terms of the accuracy for determining close contacts. Next, we briefly discuss the interactions between the blockchain backend solution and its functionality.

5.2.1. *Testing Results for Sound Based Proximity Detection*

Table 5 shows the results in a confined room. The maximum distance tested is 3 meters due to resource constraints. In the confined room, all the distances reported are labelled 'Near'; given that the virus would transmit itself the easiest in a confined space, it is important that detections in a confined room indicate a close contact occurring.

Next, Table 6 shows the results for tests conducted in a corridor located in an indoor space with distances up to 20m. At 1 and 2 meters, the smartphones reported consistent results with the confined room reports. At 3 meters, false negatives start to occur. After 3 meters, the results reported show 'Far' consistently, indicating the threshold for transition between 'Near' and 'Far' readings happen at around this distance. Compared to the confined room, the results also suggest that echoes and surrounding noises may play a role in the results reported. The difference in

Table 5: Results for tests conducted in a confined room

	Technology						Proximity					
	Bluetooth			Sound			OnePlus 7			POCO F3		
1m	✓	✓	✓	✓	✓	✓	Near	Near	Near	Near	Near	Near
2m	✓	✓	✓	✓	✓	✓	Near	Near	Near	Near	Near	Near
3m	✓	✓	✓	✓	✓	✓	Near	Near	Near	Near	Near	Near

outcomes between the first and second smartphone at 3 meters indicates that hardware differences impact the threshold for when a signal is deemed ‘Near’ or ‘Far’.

Table 6: Results for tests conducted in an indoor space

	Technology						Proximity					
	Bluetooth			Sound			OnePlus 7			POCO F3		
1m	✓	✓	✓	✓	✓	✓	Near	Near	Near	Near	Near	Near
2m	✓	✓	✓	✓	✓	✓	Near	Near	Near	Near	Near	Near
3m	✓	✓	✓	✓	✓	✓	Far	Near	Far	Near	Near	Near
5m	✓	✓	✓	✓	✓	✓	Far	Far	Far	Far	Far	Far
15m	✓	✓	✓	✓	✓	✓	Far	Far	Far	Far	Far	Far
20m	✓	✓	✓	✓	✓	✓	Far	Far	Far	Far	Far	Far

Table 7 shows the results for tests conducted in an outdoor space. At 15 meters, the sound detection process does not work, but the Bluetooth detection process still works. A possible explanation for the difference in results when compared to an indoor space is that echoes, especially the ones in corridors, play a huge effect on how sound propagates and hence whether the sound signal is detected or not. However, in one of the instances in outdoor space, at a distance of 20m, the sound detection process managed to succeed. Although this is treated as an outlier, it is important to note that the results reported could be affected by many factors such as differences in hardware, environment, ambient noise. Nevertheless, the detection still results in an evaluation of “Far”, indicating that the implementation can still differentiate between close contacts and not close contacts.

Table 7: Results for tests conducted in an outdoor space

	Technology						Proximity					
	Bluetooth			Sound			OnePlus 7			POCO F3		
1m	✓	✓	✓	✓	✓	✓	Near	Near	Near	Near	Near	Near
2m	✓	✓	✓	✓	✓	✓	Near	Near	Near	Near	Near	Near
3m	✓	✓	✓	✓	✓	✓	Far	Far	Far	Near	Near	Near
5m	✓	✓	✓	✓	✓	✓	Far	Far	Far	Far	Far	Far
15m	✓	✓	✓	✗	✗	✗	-	-	-	Far	Far	Far
20m	✓	✓	✓	✗	✗	✓	-	-	Far	-	-	Far

Table 8 shows some additional situations that were included in the tests. The situations include: the two phones are separated by a wall, located on two different floors in a building, and separated by several rooms. These different situations will determine how obstacles affect the effectiveness of signal transmission for both Bluetooth and sound.

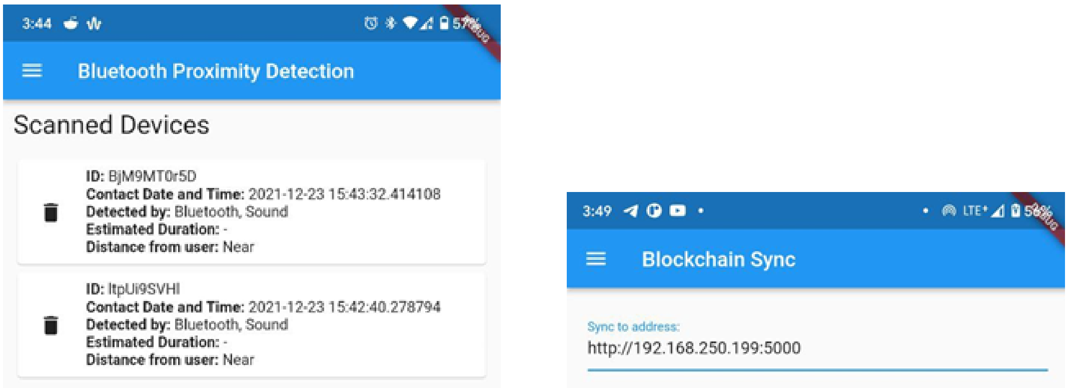
Results show that behind a wall, sound proximity still works. Sound-based proximity detection does not indicate close proximity at different floors, although a Bluetooth connection can be established between the devices. As for different rooms, one of the Bluetooth proximity detections could not identify the presence of the other device. However, all sound proximity detections did not work even when Bluetooth proximity detections succeeded. Although the proximity detection works when devices are separated by a wall, the obtained results show that both devices detect each other as ’Far’. Although the ideal outcome is that the devices should not detect each other’s presence when separated by a wall, the results recorded were ‘Far’, which is a usable result as it does not indicate close contact between the devices.

Table 8: Results for additional situations (A: Separated by a wall with each having distance of 1m from the wall, B: Different floors, C: Different rooms (Devices are placed 2 rooms apart)

	Technology						Proximity					
	Bluetooth			Sound			OnePlus 7			POCO F3		
A:	✓	✓	✓	✓	✓	✓	Far	Far	Far	Far	Far	Far
B	✓	✓	✓	✗	✗	✗	-	-	-	-	-	-
C	✓	✓	✗	✗	✗	✗	-	-	-	-	-	-

5.2.2. *Testing Results for Blockchain Back End Solution*

Some mock data are prepared in the smartphone application to upload to the blockchain back end for blockchain testing. Figure 6(a) illustrates the mock data on the smartphone. The data includes two close contact records that have been captured on the smartphone. Here, “ID” is the unique identifier that is used to identify close contacts. The “Contact Date and Time” shows the time the close contact is recorded. “Detected by” indicates the medium of proximity detection. Finally, “Distance from user” shows if the case is considered a close contact or not through the values “Far” and “Near”. Figure 6(b) shows the address that the application will upload the data to. Note that the node to which the application syncs is the pool node, which is tasked to hold individual cases before they are combined into a block.



3:44

Bluetooth Proximity Detection

Scanned Devices

ID: BjM9MT0r5D
Contact Date and Time: 2021-12-23 15:43:32.414108
Detected by: Bluetooth, Sound
Estimated Duration: -
Distance from user: Near

ID: ltpUi9SVHI
Contact Date and Time: 2021-12-23 15:42:40.278794
Detected by: Bluetooth, Sound
Estimated Duration: -
Distance from user: Near

(a) Mock contact tracing data

3:49

Blockchain Sync

Sync to address:
http://192.168.250.199:5000

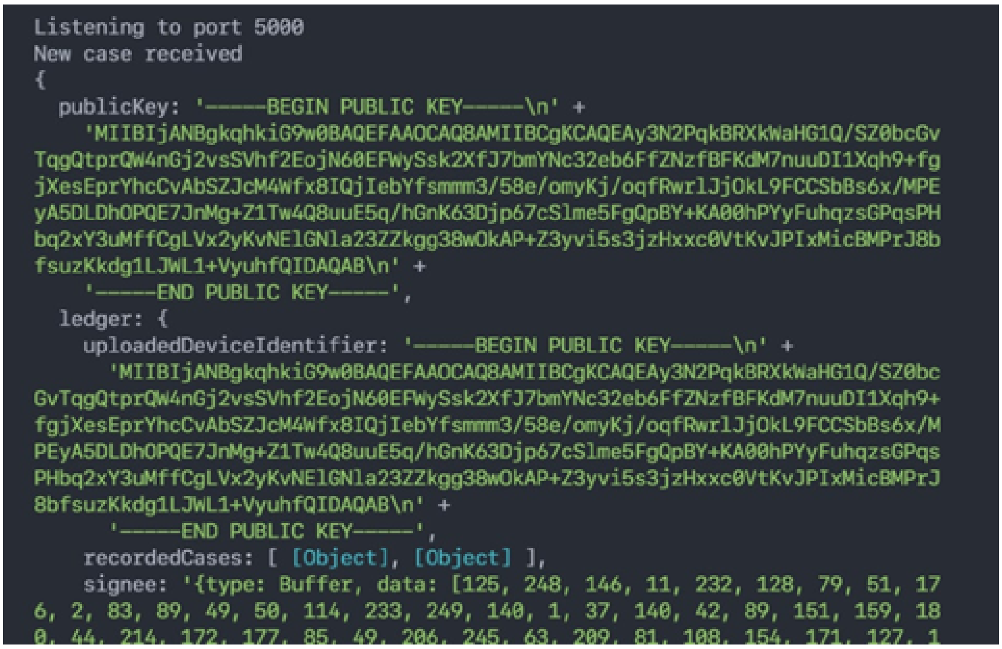
(b) Address of a node that the application will sync to

Figure 6: Interface of the implementation

To simulate the completion of a block, upload is performed several more times. Figure 7 shows the data structure for the data that is uploaded onto the blockchain. The field “signee” is the digital signature from the diagnostician, while the “recordedCases” field is the close contact cases recorded by the smartphones. The field “publicKey” is the public key of the user, used to verify another signature that is also attached to the data. The field “uploadedDeviceIdentifier” is the identifier of the device that uploaded this case, where the public key is used as a placeholder. When the received cases reach the predefined block length, the pool node will combine them and send off the block to miner nodes to mine.

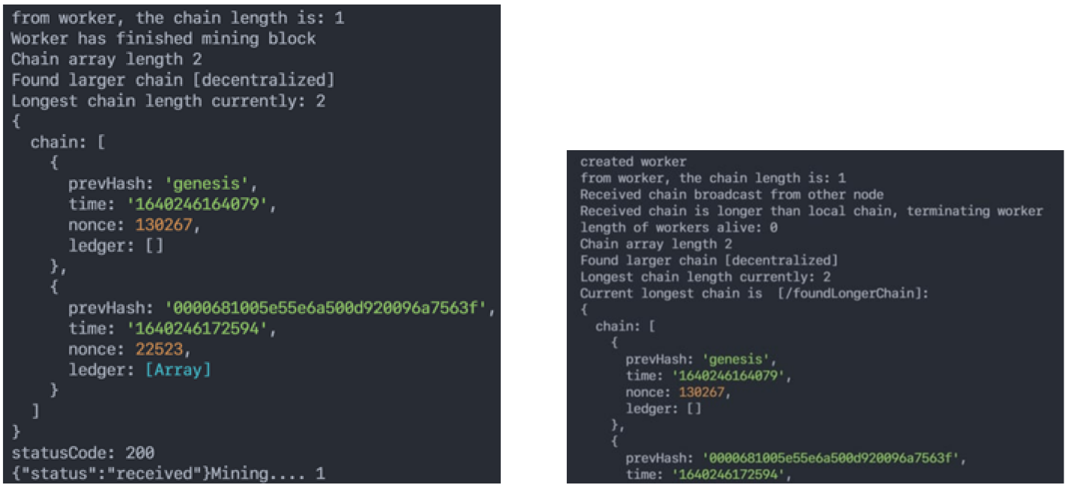
Figure 8(a) shows one of the miner nodes is receiving the new block and manages to find the proof-of-work first. It then proceeds to send the block across the network. On the other hand, Figure 8(b) shows the scenario where one of the miner nodes receives the new block but does not manage to find the proof-of-work before the other miner node. Therefore, it discards the now invalid block after receiving the new block from the other node.

The next test involves syncing cases from the blockchain. For testing purposes, the user will key in the identifier to be found. Figure 9 illustrates the smartphone application receiving cases from a blockchain node and locating the



```
Listening to port 5000
New case received
{
  publicKey: '-----BEGIN PUBLIC KEY-----\n' +
    'MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAy3N2PqkBRXkWaHG1Q/SZ0bcGv
TqgQtprQW4nGj2vsSVhf2EojN60EFWySsk2XfJ7bmYNc32eb6FfZNzfBFKdM7nuuDI1Xqh9+fg
jXesEprYhcCvAbSZJcM4Wfx8IQjIebYfsmmm3/58e/omyKj/oqfRwrlJjOkL9FCCSbBs6x/MPE
yA5DLDhOPQE7JnMg+Z1Tw4Q8uuE5q/hGnK63Djp67cSlme5FgQpBY+KA00hPYyFuhqzsGPqsPH
bq2xY3uMffCgLVx2yKvNElGNla23ZZkgg38wOkAP+Z3yvi5s3jzHxxc0VtKvJPIxMicBMPrJ8b
fsuzKkdg1LJWL1+VyuhfQIDAQAB\n' +
    '-----END PUBLIC KEY-----',
  ledger: {
    uploadedDeviceIdentifier: '-----BEGIN PUBLIC KEY-----\n' +
      'MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAy3N2PqkBRXkWaHG1Q/SZ0bc
GvTqgQtprQW4nGj2vsSVhf2EojN60EFWySsk2XfJ7bmYNc32eb6FfZNzfBFKdM7nuuDI1Xqh9+
fgjXesEprYhcCvAbSZJcM4Wfx8IQjIebYfsmmm3/58e/omyKj/oqfRwrlJjOkL9FCCSbBs6x/M
PEyA5DLDhOPQE7JnMg+Z1Tw4Q8uuE5q/hGnK63Djp67cSlme5FgQpBY+KA00hPYyFuhqzsGPqs
PHbq2xY3uMffCgLVx2yKvNElGNla23ZZkgg38wOkAP+Z3yvi5s3jzHxxc0VtKvJPIxMicBMPrJ
8bfsuzKkdg1LJWL1+VyuhfQIDAQAB\n' +
      '-----END PUBLIC KEY-----',
    recordedCases: [ [Object], [Object] ],
    signee: '{type: Buffer, data: [125, 248, 146, 11, 232, 128, 79, 51, 17
6, 2, 83, 89, 49, 50, 114, 233, 249, 140, 1, 37, 140, 42, 89, 151, 159, 18
0, 44, 214, 172, 177, 85, 49, 206, 245, 63, 209, 81, 108, 154, 171, 127, 1
```

Figure 7: Pool node showing the case that is received



```
from worker, the chain length is: 1
Worker has finished mining block
Chain array length 2
Found larger chain [decentralized]
Longest chain length currently: 2
{
  chain: [
    {
      prevHash: 'genesis',
      time: '1640246164079',
      nonce: 130267,
      ledger: []
    },
    {
      prevHash: '0000681005e55e6a500d920096a7563f',
      time: '1640246172594',
      nonce: 22523,
      ledger: [Array]
    }
  ]
}
statusCode: 200
{"status":"received"}Mining.... 1
```

(a) Miner node managing to arrive at the proof of work first and transmits it across the network

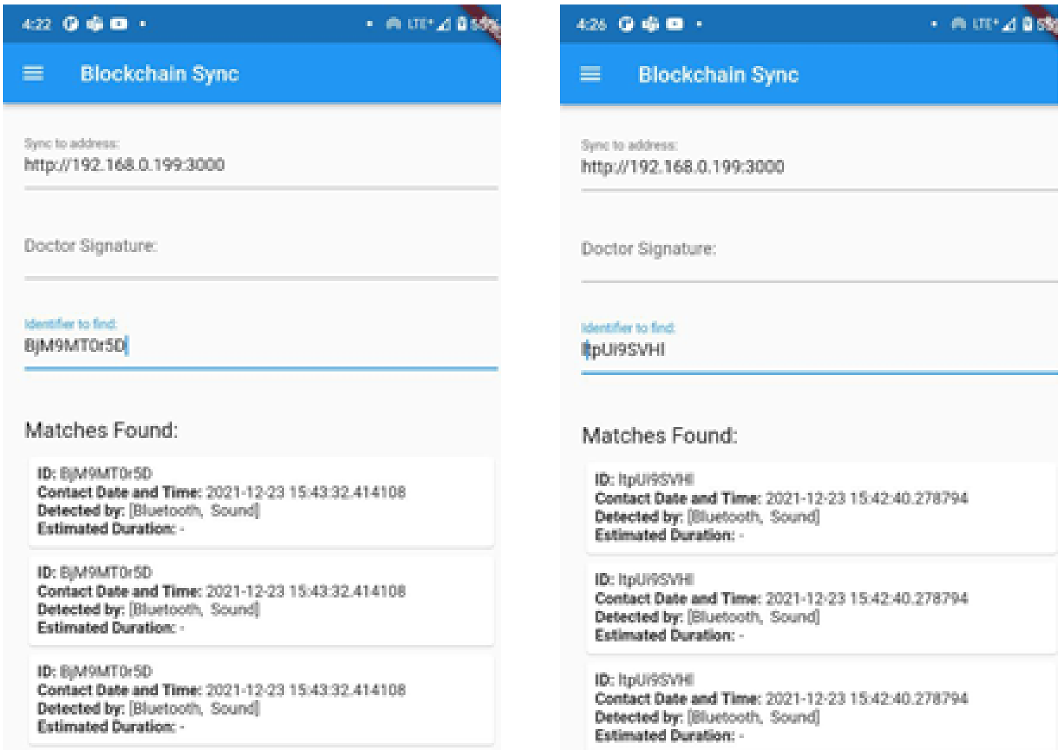
```
created worker
from worker, the chain length is: 1
Received chain broadcast from other node
Received chain is longer than local chain, terminating worker
length of workers alive: 0
Chain array length 2
Found larger chain [decentralized]
Longest chain length currently: 2
Current longest chain is  [/foundLongerChain]:
{
  chain: [
    {
      prevHash: 'genesis',
      time: '1640246164079',
      nonce: 130267,
      ledger: []
    },
    {
      prevHash: '0000681005e55e6a500d920096a7563f',
      time: '1640246172594',
```

(b) Miner node did not manage to arrive at proof of work before receiving the new valid block from the other node

Figure 8: Computing the proof of work

matching identifiers in the chain. "Sync to address" allows the user to choose which node of the blockchain to sync to and from by entering the address of the node. In this case, it is a local node with port 5000, which is the pool node. "Identifier to find" allows the user to key in the identifier they wish to locate from the chain that is synced to the smartphone.

In this example case, the identifiers are "BjM9MT0r5D" and "ltpUi9SVHl". All the cases found in this testing have the same data as they are only mock cases that were uploaded multiple times. The three instances show that the application can locate all the matches present in the chain. After finding the matches, the application will display the matches below the "Matches Found" title.



Sync to address:
http://192.168.0.199:3000

Doctor Signature:

Identifier to find:
BjM9MT0r5D

Matches Found:

ID: BjM9MT0r5D
Contact Date and Time: 2021-12-23 15:43:32.414108
Detected by: [Bluetooth, Sound]
Estimated Duration: -

ID: BjM9MT0r5D
Contact Date and Time: 2021-12-23 15:43:32.414108
Detected by: [Bluetooth, Sound]
Estimated Duration: -

ID: BjM9MT0r5D
Contact Date and Time: 2021-12-23 15:43:32.414108
Detected by: [Bluetooth, Sound]
Estimated Duration: -

(a) local identifier "BjM9MT0r5D"

Sync to address:
http://192.168.0.199:3000

Doctor Signature:

Identifier to find:
ltpUi9SVHI

Matches Found:

ID: ltpUi9SVHI
Contact Date and Time: 2021-12-23 15:42:40.278794
Detected by: [Bluetooth, Sound]
Estimated Duration: -

ID: ltpUi9SVHI
Contact Date and Time: 2021-12-23 15:42:40.278794
Detected by: [Bluetooth, Sound]
Estimated Duration: -

ID: ltpUi9SVHI
Contact Date and Time: 2021-12-23 15:42:40.278794
Detected by: [Bluetooth, Sound]
Estimated Duration: -

(b) local identifier "ltpUi9SVHI"

Figure 9: Receiving the chain from a blockchain node and matching it with the local identifier

5.3. A Comparison with Existing Implementations

In this section, the proposed implementation is compared with available solutions on the market to evaluate the implementation as a whole. Table 9 provides a summary of the features between the available proximity contact tracing implementations in the market. The features that are discussed include:

Table 9: A comparison of features in proximity contact tracing applications on the market

Contact tracing solution	Bluetooth	Sound	Blockchain
BeepTrace	✓	✗	✓
Pronto C2	✓	✗	✓
Google Apple Contact Tracing	✓	✗	✗
DP3T	✓	✗	✗
NOVID	✓	✓	✗
This work	✓	✓	✓

- Bluetooth: The implementation uses Bluetooth for proximity contact tracing.
- Sound: The implementation uses Sound for proximity contact tracing.
- Blockchain: The implementation uses blockchain for proximity contact tracing.

As seen from Table 9, the proposed implementation is the only one that integrates Bluetooth, Sound and Blockchain technologies into a coherent solution. It is worth noting that, among all these solutions, only NOVID has explored the use of sound waves to estimate the distance. The difference between our work and the solution used for NOVID is that we utilized sound amplitude for calculating the distance. In contrast, NOVID utilized time-of-flight to estimate the distance. Nevertheless, both work validates the potential of using sound wave as one of the methods for estimating proximity.

Overall, the proposed system shows better results in distance estimation than if only a Bluetooth implementation is used. The system has also demonstrated its ability to reduce false positives when the phones are on different floors or separated by several rooms. Therefore, the proposed system accurately reflects conditions where virus transmission can occur between smartphone users detected by Bluetooth.

6. Conclusion

In this work, we propose a sound assisted blockchain-based contact tracing solution to solve the limitations of using only Bluetooth for contact tracing. On top of that, the solution provided gives proof of concept for a blockchain backend solution that solves problems posed by centralized solutions.

Results obtained show that there is a promise in implementing a sound assisted solution, although it is susceptible to problems such as differences in hardware or echoes. Although these problems may not allow for accurate distance measuring using the sound amplitude method, it can still be used to obtain useful results relevant to the environment and circumstance where the two devices come into contact. For instance, when compared to Bluetooth, the sound proximity detection method will not work on phones on different floors or across different rooms, but Bluetooth will. Therefore, the proposed solution will reduce the number of false positives. Moreover, although the detection may vary across environments such as outdoors or indoors, the relevant close contact cases (within 3m) are still recorded correctly. For the blockchain part of the implementation, results show that the application can upload and download cases from a blockchain node. The pool node can also create a new block when it reaches a specific block size. Results have also shown that the nodes compete to arrive at the correct "proof-of-work" and propagate the new block across the network when it arrives at the "proof-of-work". The nodes will also stop their current "proof-of-work" if they receive a longer chain from the other nodes, as their current block is now invalid. The use of blockchain solves the problem of data manipulation by a centralized entity; it also provides auditability by the masses.

Acknowledgments

This work has been funded by the Xiamen University Malaysia Research Fund under grants XMUMRF/2019-C3/IECE/0005 and XMUMRF/2022-C9/IECE/0032.

References

[1] Who coronavirus (covid-19) dashboard, https://covid19.who.int/, accessed: 20 December 2021.

[2] M. S. Bangura, M. J. Gonzalez, N. M. Ali, R. Ren, Y. Qiao, A collaborative effort of china in combating covid-19, Global health research and policy 5 (1) (2020) 1–3. doi:10.1186/s41256-020-00174-z.

[3] A. U. M. Shah, S. N. A. Safri, R. Thevadas, N. K. Noordin, A. Abd Rahman, Z. Sekawi, A. Ideris, M. T. H. Sultan, Covid-19 outbreak in malaysia: Actions taken by the malaysian government, International Journal of Infectious Diseases 97 (2020) 108–116. doi:10.1016/j.ijid.2020.05.093.

[4] What's the difference between a blockchain and a database?, https://www.ibm.com/blogs/blockchain/2019/01/whats-the-difference-between-a-blockchain-and-a-database/, accessed: 30 January 2022 (2019).

[5] H.-C. Hsiao, C.-Y. Huang, S.-M. Cheng, B.-K. Hong, H.-Y. Hu, C.-C. Wu, J.-S. Lee, S.-H. Wang, W. Jeng, An empirical evaluation of bluetooth-based decentralized contact tracing in crowds (2021). arXiv:2011.04322.

[6] K. A. Nguyen, Z. Luo, C. Watkins, Epidemic contact tracing with smartphone sensors, Journal of Location Based Services 14 (2) (2020) 92–128. doi:10.1080/17489725.2020.1805521.

[7] World health organization. coronavirus disease (covid-19): contact tracing, https://www.who.int/news-room/questions-and-answers/item/coronavirus-disease-covid-19-contact-tracing, accessed: 2021-11-29.

[8] F. Legendre, M. Humbert, A. Mermoud, V. Lenders, Contact tracing: An overview of technologies and cyber risks (2020). arXiv:2007.02806.

[9] K. Warner, S. Nowais, Coronavirus: Doctors urge public to help track covid-19 cases with tracing app, The National (2020).

[10] L. Ferretti, C. Wymant, M. Kendall, L. Zhao, A. Nurtay, L. Abeler-Dörner, M. Parker, D. Bonsall, C. Fraser, Quantifying sars-cov-2 transmission suggests epidemic control with digital contact tracing, Science 368 (6491). doi:10.1126/science.abb6936.

[11] Contact tracing: Using digital tools, https://www.cdc.gov/coronavirus/2019-ncov/downloads/digital-contact-tracing.pdf, accessed: 30 January 2022.

[12] L. Ricci, D. D. F. Maesa, A. Favenza, E. Ferro, Blockchains for covid-19 contact tracing and vaccine support: A systematic review, IEEE Access 9 (2021) 37936–37950. doi:10.1109/ACCESS.2021.3063152.

[13] I. Hamilton, Iceland had the most-downloaded contact-tracing app for its population size. authorities there say it hasn't made much difference, Business Insider 12, accessed: 30 January 2022.

[14] Aarogya setu app, https://www.aarogyasetu.gov.in/, accessed: 30 January 2022.

[15] A. Clarance, Aarogya setu: Why india's covid-19 contact tracing app is controversial, https://www.bbc.com/news/world-asia-india-52659520, accessed: 30 January 2022.

[16] I. Nakamoto, S. Wang, Y. Guo, W. Zhuang, A qr code–based contact tracing framework for sustainable containment of covid-19: Evaluation of an approach to assist the return to normal activity, JMIR mHealth and uHealth 8 (9) (2020) e22321. doi:10.2196/22321.

[17] Mysejahtera, https://mysejahtera.malaysia.gov.my/, accessed: 30 January 2022.

[18] P.-S. Loh, Accuracy of bluetooth-ultrasound contact tracing: experimental results from novid ios version 2.1 using 5-year-old phones, https://www.novid.org/downloads/20200626-accuracy.pdf (2020).

[19] P.-S. Loh, Flipping the perspective in contact tracing (2020). arXiv:2010.03806.

[20] S. Nakamoto, Bitcoin: A peer-to-peer electronic cash system, http://bitcoin.org/bitcoin.pdf (2008).

[21] H. Xu, L. Zhang, O. Onireti, Y. Fang, W. J. Buchanan, M. A. Imran, Beeptrace: Blockchain-enabled privacy-preserving contact tracing for covid-19 pandemic and beyond, IEEE Internet of Things Journal 8 (5) (2020) 3915–3929. doi:10.1109/JIOT.2020.3025953.

[22] G. Avitabile, V. Botta, V. Iovino, I. Visconti, Towards defeating mass surveillance and sars-cov-2: The pronto-c2 fully decentralized automatic contact tracing system, Cryptology ePrint Archive, Report 2020/493, https://ia.cr/2020/493 (2020).

[23] Apple and google partner on covid-19 contact tracing technology, https://www.apple.com/newsroom/2020/04/apple-and-google-partner-on-covid-19-contact-tracing-technology/, accessed: 30 January 2022.

[24] Apple and google partner on covid-19 contact tracing technology, https://www.blog.google/inside-google/company-announcements/apple-and-google-partner-covid-19-contact-tracing-technology/, accessed: 30 January 2022.

[25] Tracetogether app, https://support.tracetogether.gov.sg/hc/en-sg, accessed: 30 January 2022.

[26] P. Esteves-Verissimo, J. Decouchant, M. Völp, A. Esfahani, R. Graczyk, Prilok: Citizen-protecting distributed epidemic tracing, arXiv preprint arXiv:2005.04519.

[27] C. Troncoso, M. Payer, J.-P. Hubaux, M. Salathé, J. Larus, E. Bugnion, W. Lueks, T. Stadler, A. Pyrgelis, D. Antonioli, L. Barman, S. Chatel, K. Paterson, S. Čapkun, D. Basin, J. Beutel, D. Jackson, M. Roeschlin, P. Leu, B. Preneel, N. Smart, A. Abidin, S. Gürses, M. Veale, C. Cremers, M. Backes, N. O. Tippenhauer, R. Binns, C. Cattuto, A. Barrat, D. Fiore, M. Barbosa, R. Oliveira, J. Pereira, Decentralized privacy-preserving proximity tracing (2020). arXiv:2005.12273.

[28] Fujian bamin health qr app, https://mztapp.fujian.gov.cn:8190/mztAppWeb/app/about/download.jsp, accessed: 30 January 2022.

[29] J. Herrera, H.-S. Kim, Ping-pong: Using smartphones to measure distances and relative positions, The Journal of the Acoustical Society of America 134 (5) (2013) 4134–4134. doi:10.1121/1.4831185.

[30] How to protect yourself & others, https://www.cdc.gov/coronavirus/2019-ncov/prevent-getting-sick/prevention.html, accessed: 30 January 2022.