*International Journal of Health Sciences and Research*
*DOI: https://doi.org/10.52403/ijhsr.20210401*
*Vol.11; Issue: 4; April 2021*
*Website: www.ijhsr.org*
*Original Research Article* *ISSN: 2249-9571*

# COVID-19 Contact Tracing Applications: Design and Operations

**Pulkit Verma[1], Neeraj Kumar[2]**

[1]Scientist 'C', BMI Division, Indian Council of Medical Research, Room No. 118, V. Ramalingaswami Bhawan, Ansari Nagar, New Delhi-110029
[2]Technical Officer, BMI Division, Indian Council of Medical Research, New Delhi

Corresponding Author: Pulkit Verma

## ABSTRACT

The Digital footprint we create in our modern lives through hour handheld devices and wearable gadgets is enormous. It was little known that, in the light of the recent outburst of the pandemic, this digital footprint would play an important role in contact tracing of the infected patients. This digital encounter tracing came into existence in order to relieve the Heath Authorities from the burden of manually tracking the contacts. Different countries developed a number of contact tracing apps based on the different architectures for the same purpose. This paper analyses the different intricacies of the architectures currently in use from the perspective of information storage, division of processing, capabilities and possible security concerns. Effort has been made in order to touch upon the possible attacks in each of the architectures. The paper concludes with the future research efforts in possible optimization of the architecture and the protocols, which in turn may lead to better contact tracing and containment of pandemic.

***Keywords:*** Digital Proximity, Contact Tracing, Bluetooth, GPS, DP, encryption, Anonymized, Decentralized, Open Source, PACT, DESIRE

## 1. INTRODUCTION

In the case of infectious diseases like the COVID-19 pandemic or any other viral outburst, the contact tracking of the infected patient becomes important for isolation and containment strategies. This tracking of the contacts can further be categorized as a primary, secondary, or tertiary contact. Deducing this information from manual interviews of the patient has its own limitations. In this case, digital contact tracking comes to the rescue, by utilizing the digital means of contact tracking through apps or wearable devices. The apps installed on the handheld devices communicate through encounter messages and share/save information on the servers or handheld devices itself for carrying out the patient contact analysis. The type of the information being exchanged, saved, and processed gives rise to the different architectures i.e. centralized, decentralized, and hybrid. The basic protocols being used by all these architectures is Bluetooth and GPS technology for location-based analysis. Though Bluetooth as a protocol was not designed for contact tracking and proximity analysis, still in wake of the exigency of the requirements under the current pandemic it's been used across the globe for contact tracking.

**2. Design Architecture of the contact tracing applications:** Based on different parameters there are three types of architectures we may use as shown in figure 1:

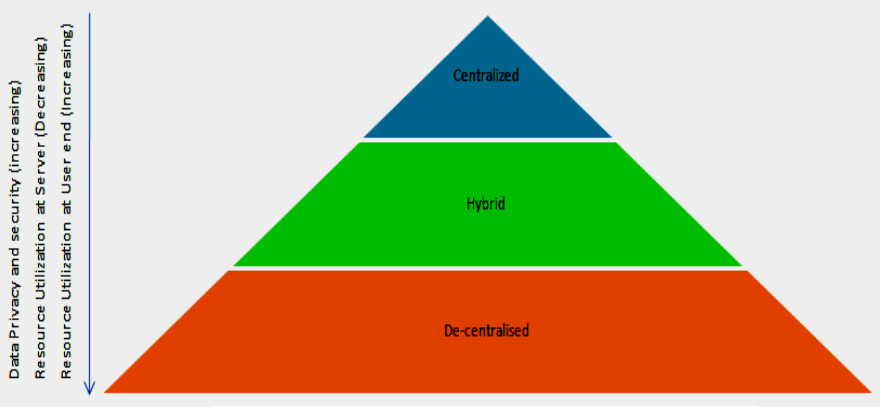


**Figure 1: Types of Design Architecture of contact tracing applications**

## 2.1. Centralized

The centralized architecture could be implemented by a number of protocols, the one we are going to discuss here is Bluetrace protocol[2], it's the most widely used privacy-preserving protocol used for centralized architecture (base protocol for TraceTogether and OpenTrace). Bluetrace works on the principle of capturing Bluetooth encounters of the user's devices in order to facilitate contact tracing while addressing privacy and security concerns. The entities involved in the interactions are the users with the client version of the tracing app, app server/tracing app server, and the Health Authorities (HA). The basic steps involved in the contact tracing app based on the centralized architecture is user registration and assignment of userIDs, generation of TempIDs, Bluetooth low energy handshake/exchange of encounter messages with a saving of encounter messages and upload of encounter data for server-side processing. The complete interaction process is depicted in the Figure 2.

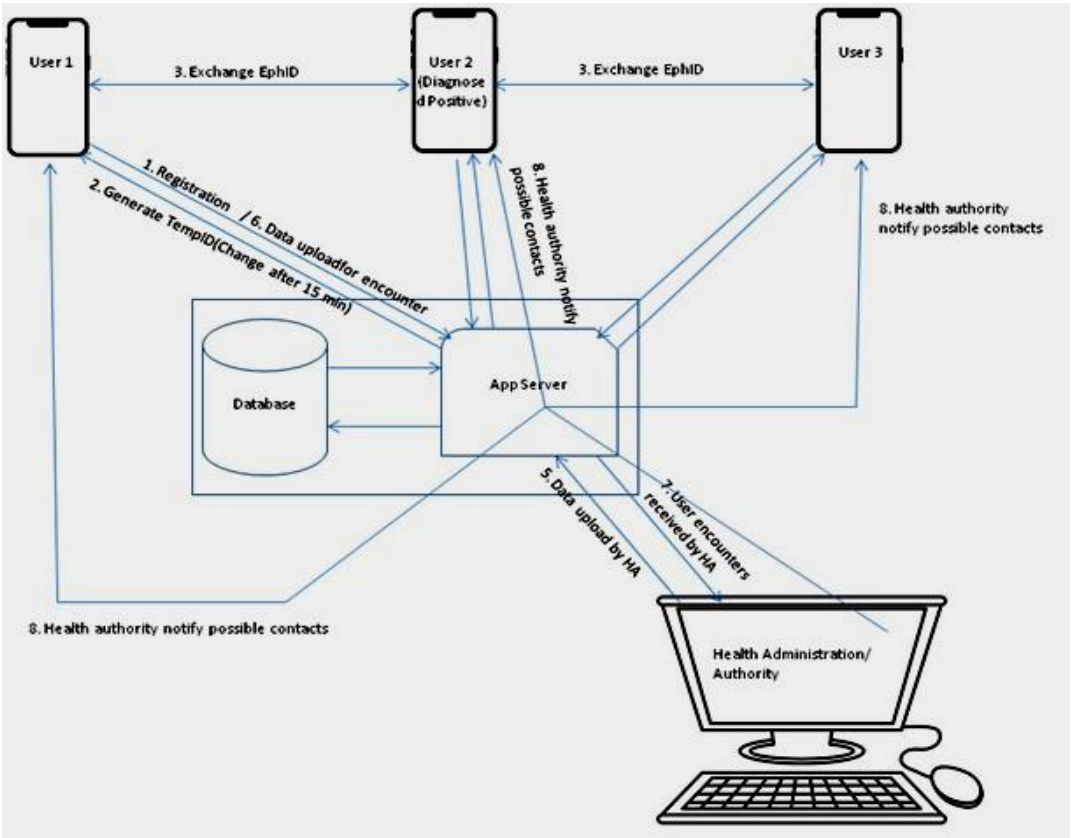


**Figure 2: Interaction flow in Centralized architecture**

### 2.1.1. User registration and assignment of userIDs :

Once the user downloads the apps from the respective app store the registration process begins with the registration of details i.e. phone number (maybe name, address age, etc.). As the reference implementation makes only phone number as mandatory filed, it is verified through a One Time Password (OTP) by the app server. Once the user is verified the server generates the userID for the registered user.

### 2.1.2. Generation and exchange of TempIDs :

As the protocol works by logging encounters of the devices by exchanging messages over Bluetooth. In order to preserve privacy these messages should not contain the user identity and disallow user tracking but at the same time allow health authorities to obtain contact information for processing encounters. This is addressed by allowing the user exchange TempID mapped to the phone number/userID. Each TempID consists of UserID, created time, and expiry time. The TempIDs are encrypted with a secret key, only the Health Authority (HA) has the access to the secret key. The TempIDs have a short lifespan in order to avoid replay attacks (default protocols recommends 15 minutes duration), more on replay, and other possible attacks in later sections.

### 2.1.3. Bluetooth low energy handshake/exchange of encounter messages:

Once the app user comes in contact with another app user, the encounter messages are exchanged and saved as a part of a Bluetooth low energy handshake. The encounter/handshake message consists of TempID, Phone Model, and Transmit Power (TxPower). The devices involved in the encounter may act a both central as well as peripheral for example a phone may act as a central as well as peripheral but a wearable device can only act as a peripheral. In order to access the proximity of the users an important parameter i.e. Received Signal Strength Indicator (RSSI) needs to be recorded by the central devices and returned back to the peripheral for saving, this facilitates the symmetric knowledge by the central and peripheral device. The saved encounter history as per the reference implementation is 21 days before it could be deleted.

### 2.1.4. Upload of encounter data for server-side processing:

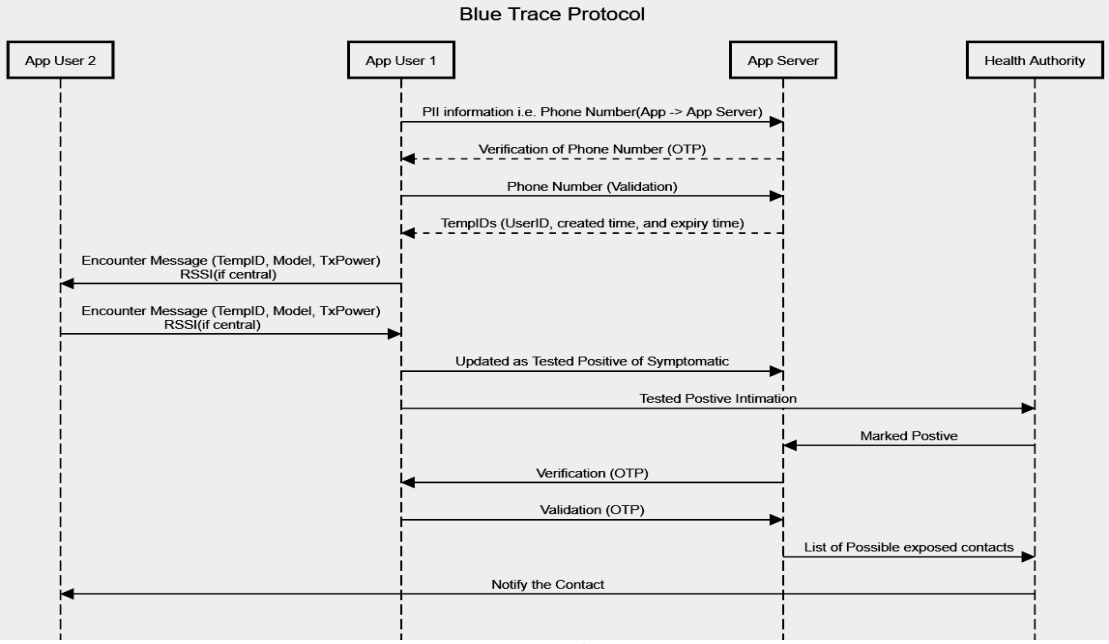


**Figure 3: Sequence Diagram of centralized architecture**

All the encounter messages are stored locally and are only uploaded to the server in case a user tests positive and voluntarily agrees to upload the same to the app server for processing. Once the user and app server authenticates each other through One Time Password (OTP), the encounter data is uploaded to the app server. The health authority decrypts the TempID for each encounter and obtains the UserID and validity period, thereafter the encounter timestamp is verified to be within the validity period, in order to avoid any attacks. Thereafter, the health authorities contact the users having a high probability of exposure to the infected/positive patient (Figure 3).

## 2.2. De-centralized

As discussed above in the case of centralized architecture the core functions related to tracing are performed by the centralized server, this raises concerns related to privacy and security in case a centralized server is compromised. In a decentralized approach, the idea is to shift the majority of processing related to tracing from server to user devices.

In a nutshell, the generation of temporary identifiers for recording the encounters as well as the risk analysis for notifications is shifted to the user devices and the server only acts as a publishing channel for the facilitation of the whole process.

The protocol, which we are going to discuss here is Private Automated Contact Tracing protocol[3] (PACT) as a base protocol to elaborate the complete architecture. In PACT protocol there is no requirement of pre-registration of the user with the app server and no Personally Identifiable Information (PII) is sent by the users to the server. The app is installed by the user and it deploys a random seed generator on the device, this seed generated (every 1 hour) is used along with the timestamp in order to generate the privacy-preserving alias for the user device called as 'chirp'(every 1 minute). These 'chirps' are periodically advertised in the Bluetooth beacon and are exchanged between the devices that come in close contact.

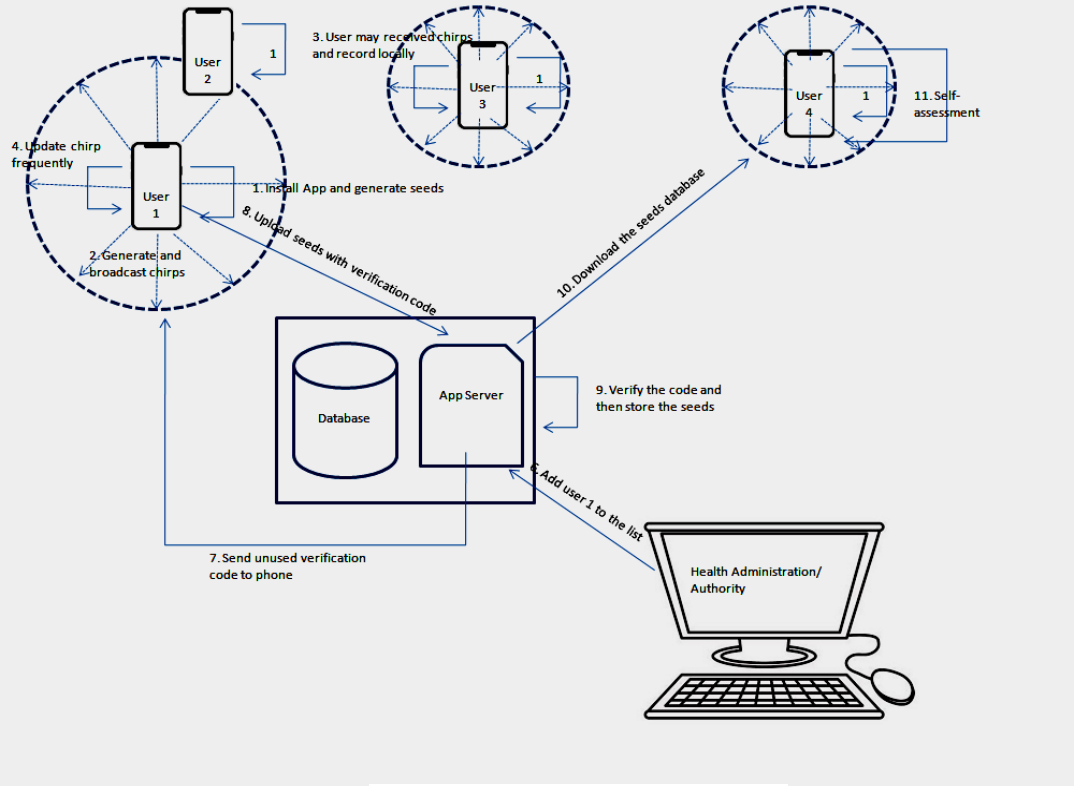


**Figure 4: Interaction flow in decentralized architecture**

Once the user is diagnosed as positive, the seed values and the time information is uploaded to the app server after the user consent. As mentioned earlier the app server just act as a publishing channel for the seed values and the other relevant information. The other app user can download the seed values along with time information in order to construct the 'chirps' sent by the infected users. These chirps are then used by the user app to perform the risk analysis for the possible exposure. The complete interaction process and message timeline is depicted in the Figure 4 and Figure 5 respectively.

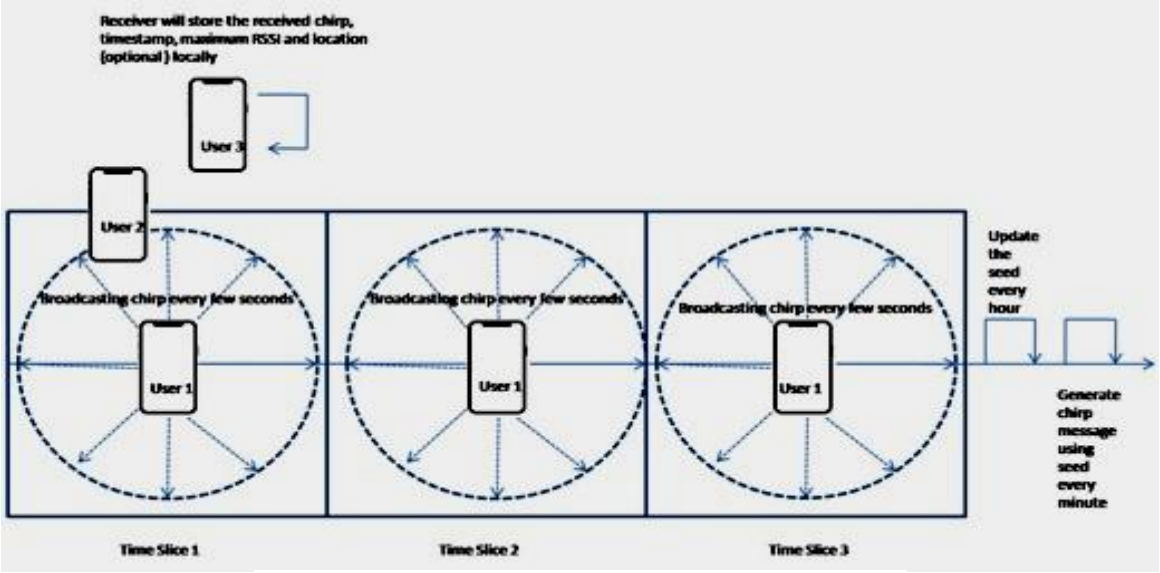


**Figure 5: Timeline of encounter exchange for Decentralized tracing app**

### 2.2.1. App installation:

The users install the app, there is no registration process and the app only installs a random seed generation algorithm on the user device after verification. This seed generation algorithm is not linked to the phone.

### 2.2.2. Generation of chirps for exchange during encounters:

The random seed is generated by the algorithm installed by the app on the user device, which in turn along with the current timestamp is used to generate the chirps (these chirps are anonymous and are not linked to user or devices). These chirps are periodically broadcasted through the Bluetooth beacon. The receiving devices record the chirp, the timestamp of receiving the chirp message, and maximum RSSI value for proximity decision.

### 2.2.3. Upload of Encounter data by positive users:

Once the user is diagnosed as positive, the users have to volunteers to upload the data to the app server and seek authorization from the Health Authority for upload of the seed values along with the creation and expiry time.

### 2.2.4. Contact tracing and risk analysis:

The user willing to perform the contact tracing with the infected patient could download the seeds along with the relevant timestamp values. Based on these parameters the user apps calculate the chirps and perform the matching with the chirps values stored in the local encounter history of the user. In case there is a matching entry for the chirp value in the user local history, the user proximity and the time duration of the exposure are calculated from the timestamps and the RSSI values respectively. (Figure 6).

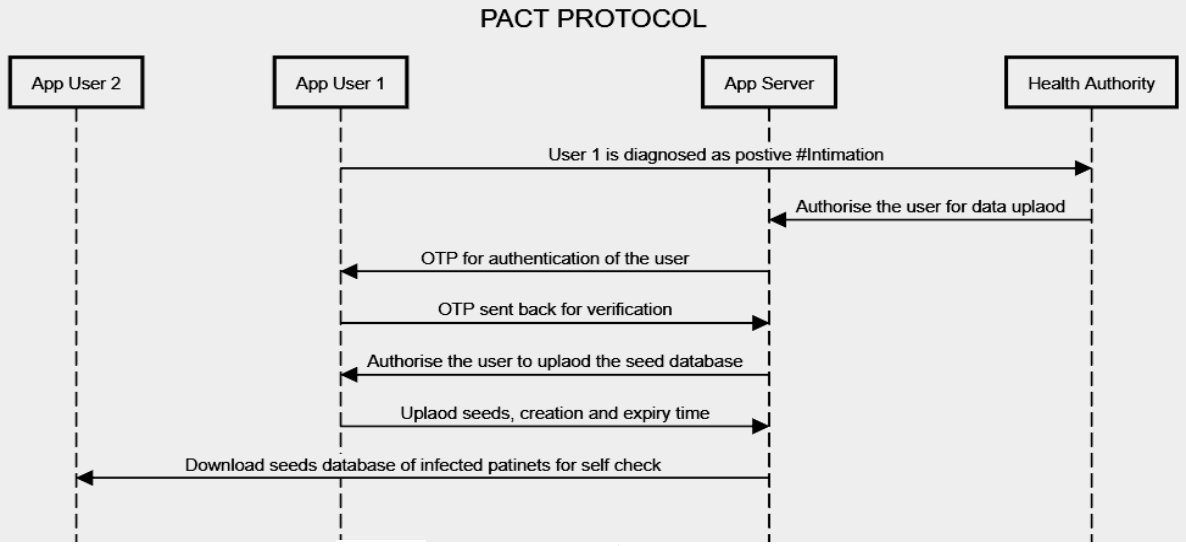


**Figure 6: Sequence Diagram of Decentralized architecture**

## 2.3. Hybrid:

Centralized and decentralized protocols have their own benefits and risks. At one end processing of TempIDs, encryption, decryption, tracing, and alerts are undertaken by the server in the centralized protocols avoiding the abuse by the malicious users, but there is an inherent risk of a malicious server compromising the privacy. Conversely, decentralized systems all the functionalities are delegated to the user devices and the server does only publishing work, it makes ephemeral Bluetooth identities of diagnosed people as public and has inherent trade-off privacy, integrity and availability.

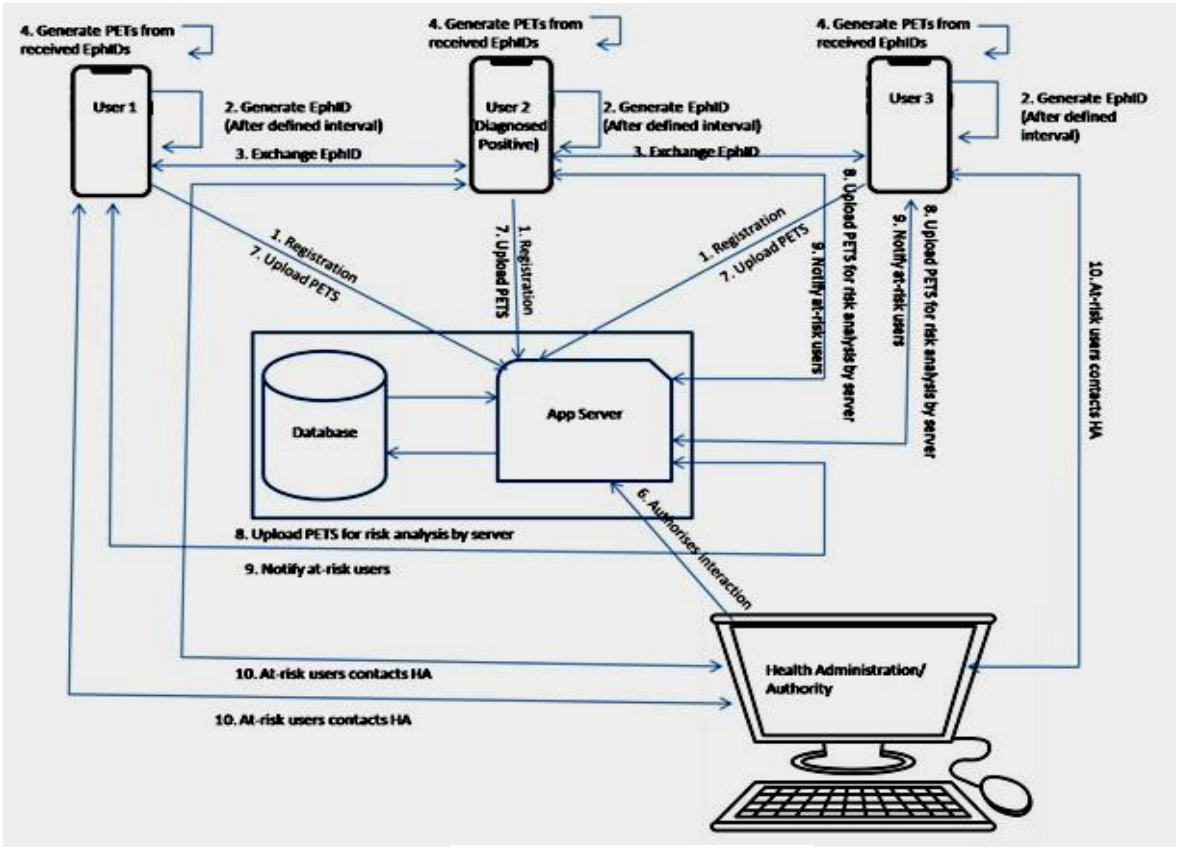


**Figure 7: Interaction flow in Hybrid architecture**

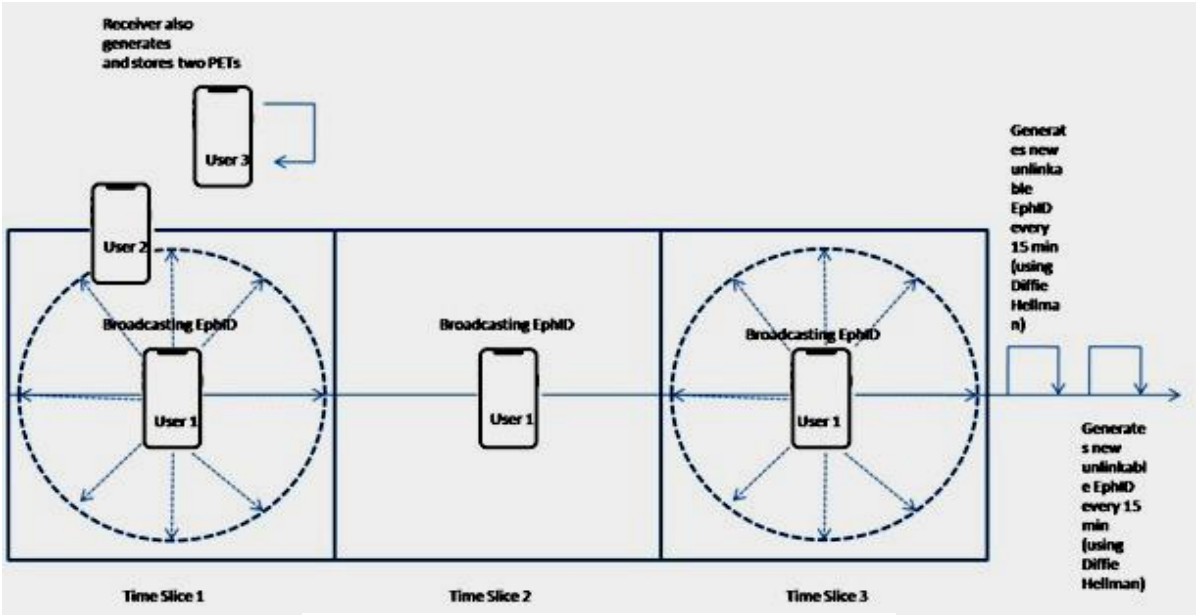


**Figure 8: Timeline of encounter exchange for Hybrid tracing app**

The hybrid architecture provides a third way by splitting the functionalities between two entities. The reference protocol we are going to discuss here DESRIE[4] protocol. The TempIDs generation, and management of encounter stays with the user devices, providing privacy while tracing, risk analysis and notifications/alerts are performed by the server. The complete interaction process and message timeline is depicted in the Figure 7 and Figure 8 respectively.

### 2.3.1. Initialization and Registration:

This step involves two factor authentications, the phone number is verified through OTP and the app is verified through authorization token issued from the server. The app is assigned a permanent identifier (ID) that is saved in the IDTable. The ID along with the encryption key is communicated to the app and the stored information is completely de-identified by deleting the phone number as well as encryption key from the server. The app only identifies to the server in the future through the permanent identifier (ID).

### 2.3.2. Proximity discover and Exchanging Ephemeral IDs:

This step in based on the private encounter token (PET) generated Ephemeral Bluetooth Identifiers (EBID). These EBIDs and PETs are generated and computed by the user devices locally and are practically unlinkable. The creation of PET could be done by any non-interactive key exchange protocol (Diffie Hellman key exchange[5] mechanism is one of them). EBIDs are generated and broadcasted by the user devices, while the EBIDs are collected for the other encountered devices at the same time. PETs are calculated from the collected EBIDs, when certain conditions of the tracing i.e. signal strength, contact length are met.

On each encounter, a user device with another device two PETs are stored in two different lists (upload list and query list). One of the PET token along with time and duration of the contact is used as actual upload to the server in case of a user is diagnosed as positive, while another PET is saved in the query table for contact tracing and detecting exposure of the user. The two unlikable PET token for each encounter helps in disallowing the server from generating the proximity graphs of the user as the server cannot identify any user based on the PET values uploaded.

### 2.3.3. Encounter Data Upload:

Once the user is diagnosed as positive, with explicit voluntary user

consent the data from the user device is uploaded to the server. The data includes the ID, encryption key, and the PET stored in the upload list along with the timestamp and duration of contact.

### 2.3.4. Contact tracing:

Any user willing to check the exposure status with the infected user uploads the PETs stored in the query table to the app server. The risk analysis is performed by the server by matching the PETs of the infected users from the global table with the uploaded PETs, using the timestamp and the duration parameters. Based on the above process the user is notified about the status of the exposure (Figure 9)

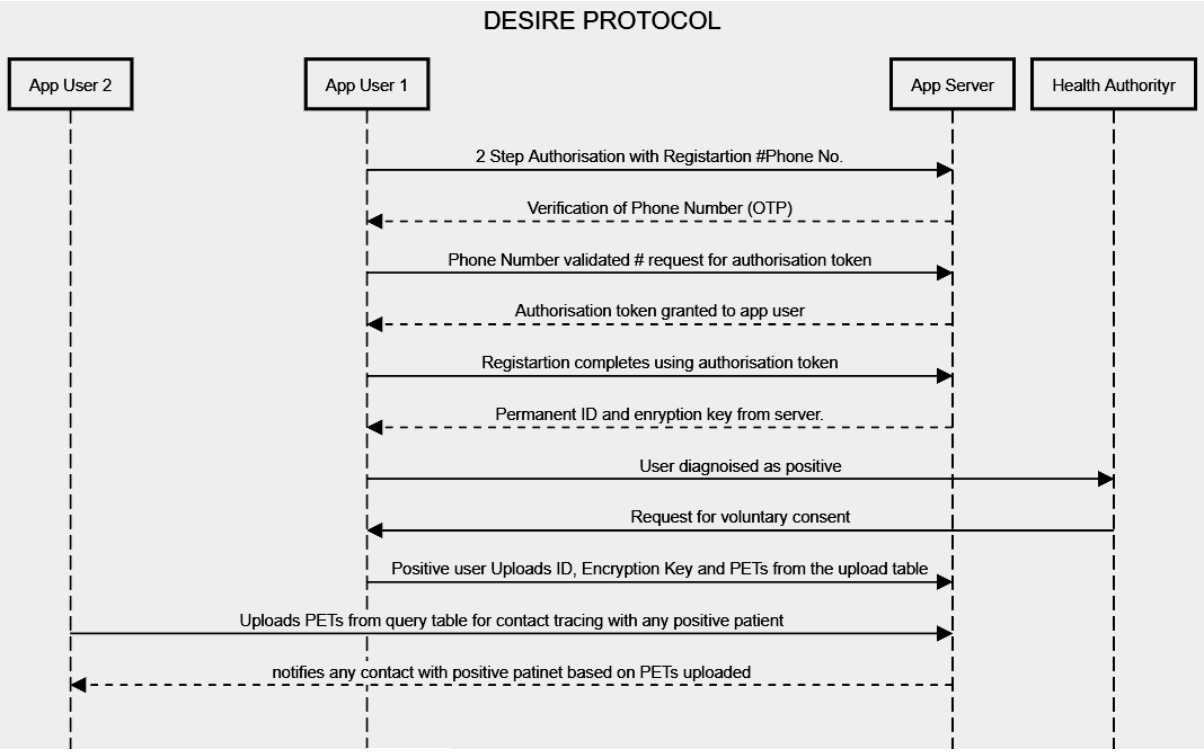


**Figure 9: Sequence Diagram of Hybrid architecture**

## 3. Attacks:

There is always a possibility of an attack on any of the architecture discussed above, the attack type and modus operandi may change. The attempt to improve the security parameters needs to be around minimizing the attack surface in the respective architectures. In this section, an attempt has been made to highlight a few of the most probable attacks[6] and their likelihood in the respective app architectures.

### 3.1. Relay/Refection and Replay attacks:

These types of attacks occur when an impersonator tries to plant a sort of man in-middle attack. The only difference is the attacker is not interested in the stealing of the personnel information, but tries to impersonate the infected patient. The end goal of the attacker is to transmit or broadcast the messages at two different locations at the same timestamp (replay attack) or at two different timestamps at the same location (replay attacks)

In centralized architecture, both replay and relay attacks are possible as the attacker has 15 minutes window before the TempID becomes invalid to replay and relay the messages. The decentralized architecture provides limited replay possible due to very small window of 1 minute after which chirps become invalid, though relay attacks can be easily implanted. While in hybrid architecture relay attacked could be implanted as symmetric information would

exist in the PET table, though replay attacks are not possible as matching PETs would not exist for the receiver of the replayed message.

**3.2. Device Tracking:** The information contained in the BLE handshake messages could be used to track the general movement of app users. The attacker's needs to implant the passive listening devices for the broadcast messaged from the app users. These messages could be processed in order to derive the location of the devices. This would lead to tracking and of the app user, which is a serious privacy concern.

The centralized and hybrid architecture provides similar scope for launching device tracking attacks, as the centralized and hybrid architectures advertise TempIDs and EBIDs respectively, with a time span of 15 minutes each. While in decentralized architecture the chirps are valid for only 1 minute, there is limited possibility of device tracking for the attacker.

**3.3. Denial of Service (DoS) attack:**

These attacks are launched in order to shut down or lower the system performance and make the system inaccessible to the intended users. The attacker would inject the false message in the complete contact tracing ecosystem leading to consumption of critical system resources (processing, battery, storage, network bandwidth etc.) both at the user device level and at the app server.

The DoS attacks under the ecosystem of tracking apps have more serious consequences on the decentralized architecture because the individual users have no means to identify the false implanted messages. In centralized and hybrid architecture the processing is under the preview of the central app server, the implanted messages fail the validation check and are discarded.

**3.4. Enumeration Attack:**

Through enumeration attacks, the attacker wants to estimate the number of infected users or the users who have volunteered to provide the tracking data to the Health authority for risk analysis.

The centralized and hybrid architecture disallow these type of attacks. In Centralized architecture, the contact information stays with the secure server and in hybrid architecture through the functionally is split between the app server and app user the server conceals the infected User ID from the other users. While in the decentralized architecture it's possible for all the users to download the seed database and estimate the number of infected patients.

**3.5. Construction of Social Graph:**

This type of attack is used to build the interaction history of users by data mining and correlating the data available through side channels. All the architecture suffers from the possibility of building social graphs (full or partial) through different means. While the construction of a social graph is fairly easy in centralized architecture, it's difficult to construct to same under decentralized and hybrid architectures.

**3.6. Linkage Attack**

In linkage attacks, the attacker tries to link the anonymous data broadcasted by the users with the data gathered through side channels. As the centralized architecture is though the users are not provided with the TempIDs for comparison, still TempIDs could be associated with the mobile model. While in decentralized architecture as the server shares the seeds along with the relevant timestamps the attackers can carry out the linkage attack on the infected users. This is not the case with hybrid architecture as the server never shares the PETs with other users.

The above attack vectors are summarized in the Table 1 for holistic reference.

**Table 1: Summary of the Possible attacks across architecture**

| Sr. No | Tracing Protocol | Replay | Relay | Device Tracking | DoS | Enumer--ation | Social Graph | Linkage |
|---|---|---|---|---|---|---|---|---|
| 1 | Centralized | Yes | Yes | Yes | Yes | No | Yes | Yes |
| 2 | Decentralized | Limited | Yes | Yes | Yes | Yes | Limited | Yes |
| 3 | Hybrid | NO | Yes | Yes | Yes | No | Limited | Yes |

## 4. Further, research areas:

There are a number of fronts on which the basic operational protocols could be improved. To start with there are requirements to optimize and reduce the false positive intimations. This is possible by either optimizing the Bluetooth protocol itself or develop a new protocol in order to cater to the specific contact tracking requirements. Further, the resource utilization and architecture-specific improvements can't be ignored either. This section tries to highlight the future scope of research and development in this domain of tracking technologies.

### 4.1. Improvement in basic technology (Bluetooth) for proximity and direction:

As already highlighted and we all would agree that Bluetooth was not designed for the sake of proximity calculation. This limitation of the protocols gives rise to a large number of false positive alarms, the reason is the proximity tracking based on the RSSI value in the Bluetooth even varies based on whether the handset is in your hand or in your pocket, front pocket or back pocket and even it is in portrait mode of landscape mode in your pocket. Further, the possible scenarios of false alarm may include even when the user just crosses an infected person in a car or bike in an open environment, one is sitting in the adjacent apartment to the infected patient as well.

The above highlighted limitations are not basically the issue with the developed application, but the limitation of the Bluetooth protocol being deployed. This requires either modification in the current protocol or development of a protocol for the specific contact tracking purpose.

### 4.2. Collation of Data from Multiple Sources:

As mentioned above the usage of Bluetooth data alone or any other sensor per se would surely generate potential false alarms. Because no single sensor currently available one handheld devices could take in to account all the real life scenarios. This issue could be preferably addressed by using data from multiple sources i.e. ambient light sensor, accelerometer and Gyroscope etc. In order to identify whether the phone is in user pocket or not. The ambient light sensor data could be easily collated with the proximity calculation algorithm for the purpose of tracking calculations.

### 4.3. Interoperability of the tracking apps:

There is a plethora of tracking apps based on different architecture within the same as well as across the different geographical regions. Currently, there is no provision for interoperability among different apps. This interoperability requirement ideally needs to flow along with the improvement in the protocol technology. The data collated by the different apps are currently residing in silos and this actually defeats the actual intended purpose.

The idea is to incorporate the interoperability feature in the protocol stack for effective data sharing among different applications across the horizon.

**4.4. Artificial Intelligence(AI) cloud implementation :**As the data sources mentioned above increases, the ability of AI algorithms[7] to self-learn and provide outcomes comes to the forefront. The AI computation capabilities could be used to study the user interaction behavior and further clubbed with the data from wearable devices. These interaction patterns could offer the ability to calculate user adherence to travel and other rules. Further, AI and the

application of graph theory can further augment the resultant outcomes. The graphs could also offer effective insights into the number of probable infections that could take place.

## CONCLUSION

The world has never seen a pandemic of this scale before and that's the reason not many deliberations took place regarding the development of the state of the art, highly accurate, interoperable, and cross region digital tracing and tracking solution. This immediate need for such a digital solution resulted in key development decisions i.e. architecture, customizations, and interoperability taken in silos by different authorities. But, now we have learnings based on the app development experiences on all the different architectures and data to support further development decisions. Though there is no single straight forward way in developing the ideal contact tracking app, but the guiding parameter for the development of such apps remains the same i.e. minimization of false alarms, optimum use of resources, transparency in implementation, and thoroughly addressed Security/privacy concerns.

**Authors' Contributions:**

**Pulkit Verma:** Literature Search, Figures, Study Design, Data Analysis, Data Interpretation, writing.

**Neeraj Kumar:** Literature Search, Figures, Study Design, Data Analysis, Data Interpretation, Writing

**Acknowledgement:** None

**Conflict of Interest:** None

**Source of Funding:** None

**Ethical Approval:** Approved

## REFERENCES

1. Dar AB, Lone AH, Zahoor S, Khan AA, Naaz R. Applicability of Mobile Contact Tracing in Fighting Pandemic (COVID-19): Issues, Challenges and Solutions. SSRN Electron J 2020 : 1–31.
2. Bay J, Kek J, Tan A, Hau CS, Yongquan L, Tan J, et al. BlueTrace : A privacy-preserving protocol for community-driven contact tracing across borders. Gov Technol Agency, Singapore 2020 : 9.
3. Rivest RL, Callas J, Canetti R, Esvelt K, Gillmor DK, Civil A, et al. The PACT protocol specification. PACT Protoc Specif 2020; 1 : 1–13. Available from: https://pact.mit.edu/wp-content/uploads/2020/04/The-PACT-protocol-specification-ver-0.1.pdf.
4. Castelluccia C, Bielova N, Boutet A, Cunche M, Lauradoux C, Métayer D Le, et al. DESIRE: A Third Way for a European Exposure Notification System Leveraging the best of centralized and decentralized systems. 2020. Available from: http://arxiv.org/abs/2008.01621.
5. Kallam S. Diffie-Hellman:Key Exchange and Public Key Cryptosystems. Math Comput Sci Dep Indiana State Univ 2015 : 1–25. Available from: http://cs.indstate.edu/~skallam/doc.pdf.
6. Ahmed N, Michelin RA, Xue W, Ruj S, Malaney R, Kanhere SS, et al. A Survey of COVID-19 Contact Tracing Apps. IEEE Access 2020; 8 : 134577–601.
7. Lalmuanawma S, Hussain J, Chhakchhuak L. Since January 2020 Elsevier has created a COVID-19 resource centre with free information in English and Mandarin on the novel coronavirus COVID- 19 . The COVID-19 resource centre is hosted on Elsevier Connect, the company's public news and information. 2020.

How to cite this article: Verma P, Kumar N. COVID-19 contact tracing applications: design and operations. *Int J Health Sci Res.* 2021; 11(4):1-11. DOI: *https://doi.org/10.52403/ijhsr.20210401*

******