Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 07, July-2020

Lost Smartphones and its Implication to Contact Tracing Designs

Hari T.S. Narayanan
Senior Consultant
NetTools Consulting

Abstract – A **Contact Tracing system collects, stores, and processes data to identify the *contacts and* the *cases* for an infection. All currently deployed *contact tracing* solutions are built with smartphones. The Bluetooth LE (BLE) present in smartphone is used in exchanging proximity data. There are solutions where this is complemented by tracking the subject's location using cellular or GPS technologies or both. Every year close to 70 million smartphones are lost globally, and most of them are lost for good. This is a large number and this paper investigates its implication to global contact tracing solution.**

Keywords - Contact Tracing, Centralized, Distributed, SARS-2, TraceTogether, Exposure Notification, Bluetooth LE, Lost Smartphones

1. INTRODUCTION

A Contact Tracing system collects, stores, and processes data to identify the ***contacts*** *and* the ***cases*** [1] for an infection. A *subject* is a ***case*** when the *subject* is having infection-specific symptoms and signs. A *subject* is a ***contact*** if the *subject* is likely infected but not having any symptoms or signs yet. A *contact* or a *case* happens when a subject is in the *proximity* of an infected person. This event could be *physical contact*, *close contact*, or *proximity contact* for contact tracing purpose. The *physical contact* includes exchange of bodily fluid. The *close contact* or simply *contact* is described by the guidelines released and periodically updated by the *Centre for Disease Control (CDC) and Prevention, US* [2]. The current guidelines for SARS-2 *close contact* is – the closeness of less than 6 feet and lasting 15 minutes or more. The *proximity contact* is described by the *CDC guidelines* as an *extended* period in the presence of an infected person. A subject who fits any of these criteria is considered as a *contact* for recommending *isolation*. These are guidelines from CDC US; other countries may follow a different set of guidelines. This paper uses CDC guidelines for comparative analysis.

Contact tracing IT initiatives enhance the manual *contact tracing* by adding agility to tracking and identifying *contacts* and *cases*. *Contact Tracing* as an IT application area is getting a lot of traction with 2015 *Ebola* outbreak [3] and current *COVID-19 pandemic*. Almost all contact tracing initiatives [4-8] are currently built with *Bluetooth LE* (*BLE*) *t*echnology [9] present in smartphones. Some initiatives also support wearables [5] as an option. The current solutions are designed differently; and their operations are localized to respective countries.

All currently deployed *contact tracing* solutions are built with smartphones. The *Bluetooth LE* (BLE) present in smartphone is used in exchanging proximity data, and in some cases complemented by *cellular* or GPS technologies or both. The exchanged data is processed when needed or periodically to identify the *contacts*. Once a *contact* is found, the *contact* may be recommended either *quarantine* or *isolation*. Every year close to 70 million smartphones are lost globally and most of them are lost for good. This is a large number and this paper investigates its implication to *centralized* and *distributed* contact tracing designs when they are extended for global community. The implications are the same for smaller communities.

There are two approaches to the *processing* of *proximity data*. In one approach, the data is uploaded and processed in a *centralized* server. In the second approach the data is processed in every registered smartphone, thus *distributing* the processing load. The latter approach may also make use of a *centralized* server. However, the bulk of the processing is done in the smartphone. There are only a limited number of contact tracing applications at this juncture. Instead of evaluating every one of these applications, two better documented solutions, one from *centralized* design and another one from *distributed* design are evaluated in this paper. The first one is a *centralized* design, *TraceTogether*, that is deployed in Singapore since April of 2020; and the second one is a *distributed* design, *Exposure Notification*, that is jointly proposed by Apple and Google in May 2020.

The rest of the paper is divided into five Sections. Section 2 provides an operational overview of one of the *centralized* designs, *TraceTogether* [5]. Section 3 provides an operational overview of one of the *distributed* designs, *Exposure Notification* [6-8]. Sections 4 is the core part of this paper that investigates the implication of smartphone loss on the two designs described in Sections 2 and 3. This section also presents the implication of temporary interruption to phone connectivity. Section 5 concludes the paper.

2. THE FOUR OPERATIONAL PHASES OF *TRACETOGETHER*

The complete operation of contact tracing solution is divided into four phases:

1. App/User Registration (Section 2.1)
2. *Proximity Identifier* Management (Section 2.2)
3. *Proximity data* exchange and logging (Section 2.3)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 07, July-2020

4. Proximity data upload and processing (Section 2.4)

The *Proximity Identifier management* includes the generation and distribution of *Proximity Identifier*. These four operational phases of *TraceTogether* are described in the rest of this section.

The *TraceTogether* uses an authenticated, encrypted, and rotating *Proximity identifier*. The App user needs to complete a registration procedure with a server at App installation time. This server is administered by the *Healthcare Ministry of Singapore Government*.

2.1. App/User Registration

The *registration* operation for *TraceTogether* starts after the user installs the App on user's smartphone. User completes the installation and starts the registration process with the server that is hardwired into the App. The server validates the App and the user, and then adds the phone number to a database that maintains users' participation in contact tracing. The database may also include many other user related data besides the phone number. These other data items are outside the scope of this paper. The server generates a random, unique 21-byte ID, *UserID*, and maps this ID to the user's phone number. This mapping is maintained in the database, and it is used later in locating the user when needed.

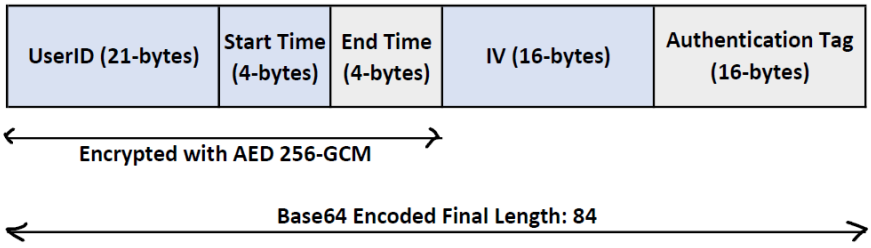

Figure 1. *TempID* of *TraceTogether*

The *UserID* is the *Proximity Identifier* (PID) for *TraceTogether* initiative. In proximity data exchange, a rolling *TempID* with encryption and message digest is used. This 84-byte *TempID* includes *UserID*, *Lifetime* of *TempID*, and other security parameters as shown in Figure 1. The *UserID* and *Lifetime* fields are encrypted with a *common key (for all the users)* maintained at the server. The entire *TempID* is protected with a message digest, *Authentication Tag*. The *UserID* offers anonymity. The *TempID* built with it is used for transport and proximity data exchange.

2.2 Proximity Identifier Management

The management of *TempID* is handled by a functional entity that is different from the one that handles the registration. The *TempID* is rotated every 15 minutes. Every registered App gets its new *TempID* periodically. The lifetime of *TempID* is specified by *start* and *expiry* time. The *start* and *expiry* time are specified using *Unix Epoch Time* [10]. A batch of *TempIDs* could be sent in a single transfer to handle Internet outages.

2.3 Proximity Data Exchange and Logging

Once the registration is done, the App starts exchanging the *proximity data* with the *peer Apps*, the set of same Apps running in other smartphones. The *proximity data* exchanged includes the *TempID* of the transmitting phone and the signal strength at the source. At the receiving side, *timestamp*, and the signal strength (*Received Signal Strength Indicator*, *RSSI*) are added to proximity data before storing it in log. The computation of distance and proximity duration are outside the scope of this paper.

2.4 Proximity Data Upload & Processing

If one of the *subject*s is infected, the healthcare authorities with the consent of the *subject* upload the *Proximity Data* (*TempID, RSSI*, *Timestamp*, and the Signal Strength at the source) from the *subject*'s phone log to the server securely. This uploaded data includes the *UserIDs of* all those subjects who had *contact* with the case in the last 25 days. The data for last 14-days (incubation period of SARS-2) is sufficient for identifying *contacts.* However, *TraceTogether* logs data for the remining 11 days to support other requirements. *The UserIDs* of the contacts need to be mapped to their respective mobile numbers for initiating *quarantine* or *isolation*. This mapping is achieved with the mapping table created at the time of registration. The term contact is also defined differently in TraceTogether: *close contact* (less than about 6 feet, for at least 30 min), *casual contact* (less than about 6 feet, for at least 5 min but less than 30 min), and *transient contact* (less than about 6 feet, but less than 5 min).

3. THE FOUR OPERATIONAL PHASES OF *EXPOSURE NOTIFICATION*

The operations of *Exposure Notification* are different from that of *TraceTogether*, still it could be understood with the same four phases.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 07, July-2020

1. App/User Registration (Section 3.1)
2. *Proximity Identifier* Management (Section 3.2)
3. *Proximity data* exchange and logging (Section 3.3)
4. Proximity data upload and processing (Section 3.4)

3.1 Registration

The registration operation downloads and installs the App on *subject*'s smartphone. The Server simply registers user's participation by storing user's mobile number in a database. This database content is accessed sequentially later for a *download* operation. No other information about the user is stored in the database. A considerable part of rest of the operations are performed in user's smartphone in the *distributed* solution.

3.2 *PID* Management

When an *Exposure Notification* App is installed on a smartphone, it takes *runtime permission* for the resources in the phone. Once this is done, then a 16-byte random number, *TEK*, is generated using *Cryptographic Random Number Generator* (*CRNG*) on the phone. A *Rotating Proximity Identifier (RPI) Key* is derived from *TEK* using *Hashed Key Defining Function* (*HKDF*). The *RPI Key* and *Discretized Unix Time* are then used to derive a *Rotating Proximity Identifier* (RPI). The *RPI* is the *Proximity Identifier* used in *Exposure Notification*. An *RPI* is generated every 10 minutes. A generated RPI is used for about 15 minutes. This is repeated for 24 hours, and then a new *TEK* is generated. The Figure 2 illustrates the relationship among *TEK, RPI Key, and RPI*. The *TEK* and *RPI* form dual proximity identifiers like *UserID* and *TempID* of *TraceTogether*. The only way to check if a given *RPI* belongs to a phone or not is by having the TEK from the phone that generated it and the *Unix Epoch Time i* [9] at which it is generated.

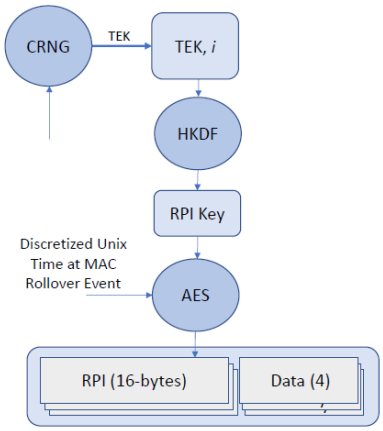

Figure 2. *RPI* Generation Process

3.3 Proximity Data Exchange

The user anonymity is maintained by using *RPI* as *Proximity Identifier*. The proximity message is a normal *Bluetooth* advertisement message (ADV_IND) differentiated by *RPI and Data* (Figure 2) in the 31-byte payload section. The proximity information is exchanged by broadcasting and scanning the contact tracing broadcast messages from the other phones. The *RPI* can only be mapped with the *TEK* from the phone that generated it. The *RPI* is not bound to any hardware address or fixed logical address on the smartphone. Every phone sends periodic broadcast of *proximity data* with RPI. Every smartphone in the neighborhood scans for *proximity data* and stores the scanned record in its *micro SD*. The proximity record includes an RPI, and the data needed to compute BLE signal strength at the source. The *timestamp* and the signal strength at the receiving end are added to the record before storing it in the log.

3.4 Data Processing

If a subject is found to be infected, then with this subject's permission, the *diagnostic keys* (*TEK, time*) of the subject for the last 14-days of incubation period are uploaded to a cloud server. This set of *diagnostic keys* is the only information uploaded. This key set *does* not identify the user. Every App fetches the list of *diagnostic keys* from the cloud server for all the recently infected *cases periodically* from the cloud. The *diagnostic keys* are then used to generate *RPIs* and check against the proximity data in their respective local log. This processing is distributed to individual's phone. If a match is found, then the data is assessed for its *proximity* and *duration* suggested by the guidelines, and a recommendation is made to the subject. If there is no match, then no action is recommended.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 07, July-2020

4. LOSS OF SMARTPHONES AND ITS IMPLICATION

There is an issue that is common to both designs – centralized and distributed. A staggering 70 million smartphones are lost globally in a year [11]. There are close to 3.7 billion smartphones in use at the time of writing this paper. A simple *uniform distribution* of this loss suggests that there is an approximate probability of 0.5×10^{-4} that a phone is lost on a specific day. This is a recurring phenomenon that can lead to data loss and other issues. A good assessment of the loss is needed to understand the implications of *contact tracing* systems. Appropriate solutions can be pursued if these implications are significant.

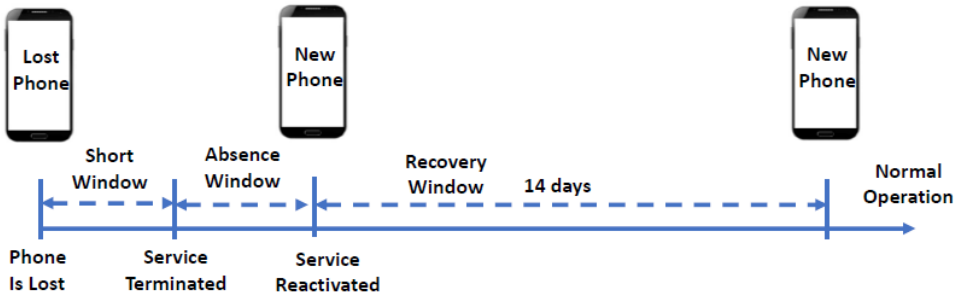

Figure 3. Loss of Phone Transition Phases

A typical sequence of phone loss and reactivation is captured in Figure 3. One of the registered users loses the phone. The phone stays active for a *short window* of time and then the service is terminated to the phone one way or another. The service is re-activated with the same identification with a newer phone after certain duration. This duration is referred to as *Absence Window* in Figure 3. The original owner of the phone starts using the new phone. It takes another 14 days for the new phone to collect contact data for the CDC suggested full incubation period of SARS-2. The complete contact tracing operation is restored after this *Recovery Window* period. The average values assumed for *Short Window* and *Absence Window* respectively are, 3 hours and 12 hours. There is no activity during *Absence Window*.

The factors considered in the evaluation includes if the phone is lost after or before uploading the contact's log, and different types of data received and transmitted from the phone during these *windows*. The highest number of single day new cases is around 250,000 [12] so far. A simple model suggests the probability that a subject becomes a case on a day is 0.68×10^{-8}.

4.1 Implication: Distributed Design, Short Window

There are three possible data exchanges during this short window: *diagnostic keys*, *contact data transmitted* from the phone, and the *contact data received* by the phone from other phones in the neighborhood.

The App on the phone will continue to download the *diagnostic keys* of newly identified cases from the cloud server and carry out the matching process with the data in the log. A large part (close to 14 days) of the data still represents the data received by the original owner except for the *short window* period of about 3 hours. The result of one such processing could identify the owner as a *contact* and recommend *quarantine*. The person who is currently having the phone will come to know of this private information, and more importantly, that information is not available to the owner of the phone. This observation suggests a potential *privacy problem*. A subject who lost the phone AND found to be a contact in one of the processing periods is of low probability. This scenario is unique to *Exposure Notification* and is of low probability, but still possible. There is also a smaller possibility the above contact status is incorrect or *false positive*.

The exchanging of proximity data with the other phones will continue during the *short window* period. The *contact data received* from other phones during this window is of no consequence. On the other hand, the data sent from this phone to all other phones in the last 14 days is valid except for the data that is sent during the *short window*. This entire transmitted data from this phone is lost along with the *diagnostics keys*. If the owner stays normal for the next 14-days, this loss is inconsequential. If the owner becomes infected within the next 14 days, then there is a loss of data. If the owner becomes a *case* on the same day after the loss, then 14-days' data is lost. If the owner becomes a case after 14 days, then there is no data loss. Thus, on the average, 7-days' data could be lost. The data loss is modeled as a function of both the probability that the owner becomes a *contact or a case* in the next 14 days and the number of RPIs sent out in the last 14 days. The probability of an individual losing the phone and becoming a *case* is low, but possible. The loss is the number of RPIs sent during the *incubation* period. The probability of this occurrence is more than the previous *short window* occurrence due to the number of days. It is the probability that a subject who just lost the phone is becoming a case in the coming 14-days.

If the owner loses the phone after the *diagnostic keys* are uploaded, then the RPIs sent during the *short window* leads to *False Positive* cases assuming the owner is not anywhere near the lost phone during this window. The probability is extremely low for this to happen.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 07, July-2020

4.2 Implication to Distributed Design – Long Window

The data transmitted and received during this *long window* is good. However, there is an issue in processing the *diagnostic Keys*. In *Exposure Notification*, the *Contact Tracing* status of the owner is almost up to date due to periodic checking carried out using the *diagnostic keys of new cases*. On the other hand, from the moment the owner reactivates the phone, it takes another 14 days for the owner to get the full data set for processing. In those 14 days, the result of the processing operation with the fetched *diagnostic* keys is unreliable due to the lost data. This problem is unique to *Exposure Notification*.

This could be understood by the following illustration. If one of the close friends of the owner becomes a *case*, every smartphone will be fetching the *diagnostic keys* of this new case including the owner's new phone. This phone due to incomplete data may not evaluate the status of the owner correctly. A negative status (no contact) is not reliable, though the positive status is fine. This suggests, the registered subject does not get full contact tracing coverage for 14 days even though there is a phone. A simple model to quantify this: no coverage for 70 million x 14 out of 3.7 billion x 365 days for a global solution. This is equal to coverage loss of 0.0725%. That is, 725-person days of coverage is lost on a million-person days of coverage. In other words, 70 million subjects will not have 14-days of contact tracing coverage in a year. This is based on current global statistics of phone loss and the number of smartphones in use. This is unique to *Exposure Notification* design due to device *affinity* [13]. Only the lost phone could have validated the diagnostic keys correctly. This is device *affinity* issue. The coverage loss is a function of the incubation period and yearly loss of smartphones. Longer the incubation period for a virus the larger the *coverage loss*.

4.3 Implication to Centralized Design – Short Window

In the centralized design, the *contact* data in the phone is safe. It can only be decrypted at the server. Thus, there is no privacy issues. The *contact* data that is received earlier and the data received after the loss (nearly 14 days valid data) are lost for processing. If the owner is not infected for the next 14 days, then this data loss is inconsequential. If the owner is infected, then this data loss is significant. On the average 7 days of contact data could be lost if the owner is in infected. This aspect of loss is identical to the distributed design.

The *contact* messages sent from this phone after the loss are no longer representing the original owner of the phone. The data transmitted during this interval could lead to *False Positive* incidences with the original owner falsely identified as a *contact*. This is of low probability considering the short interval.

If the owner becomes a case, then the *manual process* can be made to complement the automated contact tracing – the *contact* data from the phones of the subjects who spent time with the owner can be uploaded and processed to reduce the loss. This kind of semi-manual processing is not possible in distributed design due to the device *affinity*; when the device is lost, the *diagnostic keys* are lost with the phone.

4.4. Implication to Centralized Design – Long Window

In the *centralized system*, the loss of phone does not mean loss of coverage. This is because the owner's *contact status* is decided by the data in other phones unlike the distributed system where the status is decided by the data in the owner's phone. Still, the data lost for 14 days need to be built during this long window. During this period, if one of the friends of the owner becomes case then there is no issue. The contact details are likely with in the phone of the new case. If the owner becomes a case, then on the average 7-days contacts of the owner is lost as described earlier.

There is a short coverage loss of 15 hours here during the *short window* and *absence period*. This coverage loss also occurs with distributed design.

4.5 Summary

There are some common implication and some that are unique to a specific design. Most of the issues are of low probability. There is a low probability privacy issue in *distributed design*. This is unique to distributed design. A unique and a significant issue with the *distributed design* is the lack of *contact tracing coverage*. There is a lack of contact trace coverage during the *long recovery window*. This is significant and stands out compared to all other issues found in either design. This is due to the *affinity* [13] of the contact tracing process to a specific device. There is a data loss of moderate probability that can occur in either design. There is a possibility for a *semi-automated* process to partially recover from the phone loss of a new case in *centralized system*. This is not possible with the *Exposure Notification* due to device Affinity.

5. ABSENCE OF NETWORK CONNECTION

In the last section we analyzed the implication of loss of phones for good. Occasionally, a phone may lose connectivity to network for an extended period, typically few hours. This can happen due to lack of battery charge, absence of service due to billing issues, lack of Internet connectivity, intentionally turned off by the user, etc.

In the centralized design, this may lead to suspension of transmission or exchange of contact data due to the lack of *rotating proximity identifier*. The phone fails to transmit proximity data if it is using *non-intrusive* exchange, otherwise receiving could also be suspended. This creates the risk of missing out *contact identification* opportunities during these intervals.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 07, July-2020

In the distributed design, these interruptions could make the phone to fetch and process *diagnostic* keys for multiple periods or make the phone to skip over *diagnostic* keys of multiple periods. The former case could overload both the phone and the server leading to server load fluctuation and larger server storage requirement. In the latter case, there is a risk of not identifying a contact.

6. CONCLUSION

All currently deployed *contact tracing* solutions are built with smartphones. Globally 70 million smartphones are lost in a year. This is a large number and its implication to contact tracing designs are investigated in this paper. Two solutions, one from *centralized* design and another one from *distributed* design are investigated to find out the impact of this loss. In general, many of the implications are insignificant except for the following two key ones: There is a lack of *contact tracing coverage* of 70 millionx14 person-days per year with the *distributed* design. There is an option to support the loss of a phone (of a case) with a *semi-manual, partial recovery* process *in centralized* design, as opposed to the distributed design where it is not possible for any recovery.

Acknowledgement: Author acknowledges the valuable inputs received from *Jason Bay*, Senior Director, Government Digital Services (GDS), Government Technology Agency Singapore.

7. REFERENCES

[1] Coursera: COVID-19 Contact Tracing, Dr. Emily Gurley, Johns Hopkins University, Baltimore, MD.
[2] Public Health Guidance for Community-Related Exposure, June 5th 2020, COVID-19, https://www.cdc.gov/coronavirus/2019-ncov/php/public-health-recommendations.html
[3] EMERGENCY GUIDELINE - Implementation and management of contact tracing for Ebola virus disease, September 2015, WHO and CDC.
[4] *Arogya Setu* Official site: https://www.mygov.in/aarogya-setu-app/
[5] *BlueTrace.io*: A privacy-preserving protocol for community-driven contact tracing across borders:https://*TraceTogether*.io/static/*TraceTogether*_whitepaper-938063656596c104632def383eb33b3c.pdf
[6] Apple & Google, Exposure Notification Specification for Contact Tracing – a joint initiative of Apple & Google, May 2020: https://www.Apple.com/covid19/contacttracing/
[7] Apple & Google Contact Tracing Initiative, Exposure Notification Bluetooth Specification Preliminary, April 2020 v1.2
[8] Apple & Google Contact Tracing Initiative, Exposure Notification Cryptography Specification Preliminary, April 2020 v1.2
[9] "Bluetooth Smart or Version 4.0+ of the Bluetooth specification". *bluetooth.com.* Archived from the original on 10, March 2017.
[10] Unix Time: https://www.unixtimestamp.com/
[11] Statista: https://www.statista.com/statistics/201182/forecast-of-smartphone-users-in-the-us
[12] Worldometer Coronavirus statistics and other data: https://www.worldometers.info/coronavirus/
[13] Hari T.S. Narayanan, Designing Contact Tracing Solution for Global Community, a manuscript under submission.