Contents

The Taiwan Social Distancing Application and Its Contact Tracing Algorithm

Poga Po<poga@ailabs.tw>

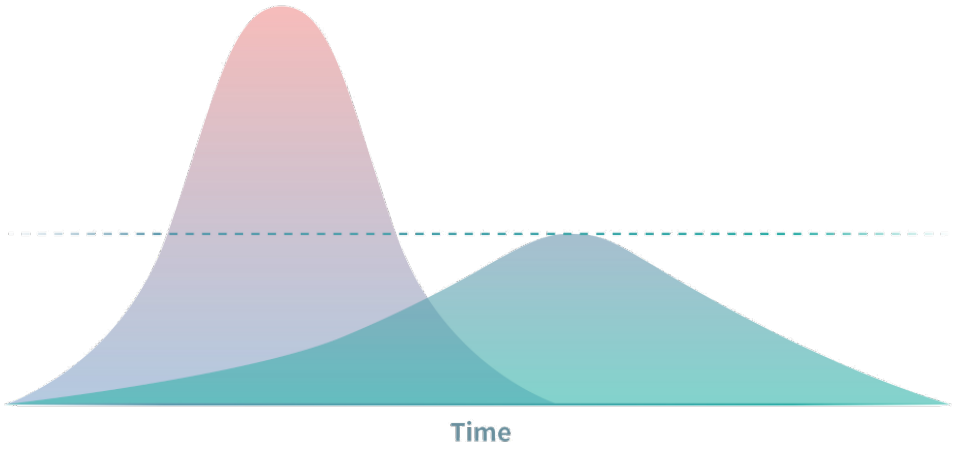
Introduction

This document describes the contact tracing algorithm used by the social distancing application form Taiwan AI Labs. The social distancing APP is a community project aiming to reduce the spread of infectious diseases.

Goals

As the SARS-Cov-2 outbreak spread across the world, people try to contain the disease with modern technologies such as cell phone location tracking. However, a simplistic understanding of cell phone tracking technology will massively hinder the protection of an individual's privacy without producing benefits.

The contact tracing algorithm we implemented is based on public algorithms such as Decentralized Privacy-Preserving Proximity Tracing from EPFL, Privacy-Preserving Contact Tracing from Google and Apple, and BlueTrace from Singapore Government Technology Agency. We believe that the challenge posed by COVID-19 requires global coordination and collaboration between governments, technologists, and the general public. This document is not just a public announcement but also an invitation to collaborate.



The goal of the social distancing application(the "APP") is to reduce reproduction number, or R0, of the SARS-Cov-2 by alerting individuals who have a high risk of infection by infected patients. To do so, the app exchanges anonymous hashed ID via Bluetooth Low Energy broadcast. Each APP compares the public infected ID list with its local contact history and alerts the user if a high-risk contact history is found.

Data Policy

The collection of an individual's contact history is a sensitive topic. The following paragraphs show what data is collected in the process, who can access these data, how is the data used, and the life cycle of the data.

To protect the privacy of individuals, no personally identifiable information, or PII, is collected in the process. The app only records anonymous hashed ID received, and a rough timestamp is saved in the local device(the cell phone). The data retention period of local data is 28 days. The collected data, also known as the contact history, will never leave the device.

When a user is confirmed as infected by public health organizations, they can provide the anonymous IDs they had broadcast within the last 28 days to the organization. For each day, the public health organizations will publish a list of IDs broadcast by known infected users. These IDs cannot be linked back to an individual and will be destroyed after 7 days.

At a defined interval, the app will download a new list of anonymous IDs from public health organizations. The app will try to match its local contact history and alert user when a high-risk contact history is found.

Requirements

On the basis of protecting individual privacy, the algorithm should also:

- Notify people at risk and give guidance on the next steps quickly and effortlessly.
- Minimize data collected in the process

- prevents abuse of data
- prevents tracking of non-infected users
- graceful dismantle itself after the outbreak is over.

How the Contact Tracing Algorithm Works

Registration

The algorithm requires no registration step before start tracking a user's contact history. All data collected by the app is generated by itself and cannot be linked back to the user. However, Additional personal information might be needed when a user is contacted by public health organizations under the condition of infected or other emergencies.

Generating Anonymous IDs

To provide better contact tracing, it's necessary that the app can observe and record as many IDs as possible. Since the algorithm relies on the Bluetooth Low Energy protocol, the length of an anonymous ID is limited by the protocol's payload size. Also, the app must record every single anonymous ID received. The length of an anonymous ID should not overburden the local storage of the device.

To avoid location tracking via broadcasted IDs, the app must frequently change the anonymous ID it's using. By default, the app will rotate to a new anonymous ID every 15 minutes. The app will broadcast the ID it's currently using via the BLE broadcast.

Each day, the app will generate 96 unique, secure, anonymous ID with the cryptography library provided by the operating system. The length of each key is 128-bits. The length based on previously stated limitations. The short rotation interval is designed to prevent large-scale location tracking and avoid linking anonymous IDs back to an individual based on other real-world information.

The app store all IDs it used in the local device storage. However, IDs older than 28 days will be discarded to keep storage usage at a reasonable level.

Preserving Contact History

The app locally store each ID it received via BLE broadcast. The coarse timestamp (e.g. 'April 14") is also stored.

Each record requires 24 bytes (16bytes ID and 8 bytes timestamp). Assumes 10k ID is received daily, about 6.4MB is required to save 28 days worth of data.

Report when Infected

When a user is tested infected, first they pass the IDs they've broadcast in the last 28 days through a crypto-secure hash function (we choose blake2b). After provides basic contact information to the public health organization, they upload the list of hashed IDs to the backend server. The server aggregate all newly infected hashed IDs, remove out-dated IDs, then publishes a new list for the app to download.

The backend is simple storage for infected IDs with no additional processing. Its job is to simply aggregate and serve the infected ID list to each app. Since no processing is required, the list can be efficiently shared via CDN or distributed servers, which lower the bandwidth cost of the backend.

The server will also remove infected IDs older than 7 days to reduce resource usage and prevent abuse of data, while still being valuable about controlling the outbreak.

To further anonymize the data when the infection case number is small, we continue to explore potential solutions such as adding spurious tokens or introducing mixing servers[5].

Download and Notification with Risk

Each day, the app will download a new infected hashed ID list from the server. It will find matches between the list and the local contact history. If a match is found, the user's risk score is calculated based on estimated distance and contact duration. The user will be notified if the calculated risk score is high. The app will also provide health care information to the user.

Security and Privacy Considerations

In the following paragraphs, we follow roughly the same framework as DP-3T to discuss our threat model, security consideration, and privacy concerns. First, we defined the potential attackers. Then we discuss their capabilities and the risk they pose to the system. Finally, we analyze the privacy and security of the system with respect to these attackers.

Threat Model

- Regular user: A typical user of the system who is able to install and use the app via the designed user interface.
- Tech-savvy user (Blackhat/Whitehat hacker, NGOs, Researchers, ...etc): This type of user can access the device's system and setup additional equipment to intercept the communication, decompile or modify the app.
- Eavesdropper (Internet Service Provider, Local System Administrators, Bluetooth sniffer): They can observe the network communications and BLE broadcast messages.
- Backend: Can access all data send to/aggregate on the backend server. They also have access to the backend's various access log entries such as IP address or Bluetooth mac address.
- State-level adversary (Law enforcement, Intelligence agencies): Has the combined capabilities of the tech-savvy user and eavesdropper. They also have access to nearly unlimited budget to perform country-scale or global-scale attack to the system.

Security Concerns

Fake Contact Events or Replay Attack

Due to the nature of the BLE broadcast, adversaries can always record received messages and replay them back to the public. An attacker can always record a received anonymous ID and replay it before it's rotated. If the owner of the ID is diagnosed as infected, the

replayed message will create multiple false-positives to other users. However, such an online attack requires a large scale of device deployment to try to record infected IDs before they're published, thus requires a larger budget which is not feasible to smaller-scale adversaries.

Suppressing At-risk Contacts

The user can always hide their contact history via removing the app, refuse to share their past IDs, disable BLE broadcast whenever they want, or just ignore the alert send by the application.

Deny of Service

Any system based on BLE broadcast is susceptible to jamming attack by active adversaries. It's an inherent problem of this approach.

Adversaries can also perform DDoS to stop the backend from publishing the infected ID list. However, since the list is a simple static file which updates daily. It can be distributed via multiple channels such as geo-distributed servers, content delivery networks, or P2P file-sharing system. Completely stop all possible channels to distribute the file is extremely hard.

Privacy Concerns

Global Interaction Graph

The global interaction graph reflects the social relationships of all users in the system. Our contact tracing application and algorithm does not reveal or process this kind of knowledge. The interaction between two users is only revealed to the two users themselves. Infected users will not share their contact history with authority. They only need to share their past anonymous IDs to the public health organization. No party can learn the global interaction graph from the system. However, a partial, relevant subset of the graph (below) might be inferred from the system.

Proximity Graph

The proximity graph encodes contacts between infected users and other individuals. This graph is essential to the contact tracing application. However, only a relevant subset (i.e. related to infected users) of the proximity graph is revealed to each user. Users will not share their subgraph with other users or authorities.

Location Traceability

The anonymous IDs in our design are unlinkable. Only the device that generated them knows the relationship between the keys. When a user is confirmed infected, the anonymous IDs they've used in the last 28 days will be shared with the public. The user's device will rotate to a new set of IDs after the user is diagnosed. Hence, published infected IDs might be linkable but still can not be linked back to the user.

At-risk Individual

At-risk individuals or other users' data will not leave their device at all. Therefore no information about them is leaked to any party.

Infected Individuals

When a user is diagnosed as infected, their IDs will be published through the backend server. Under normal user interface, users cannot directly access these IDs. Regular users cannot learn who is infected.

Close individuals such as family, friends, or colleagues might be able to identify a small set of potentially infected individuals based on their daily routines. However, they should already be notified by the health care organization about the potential infection and perform the further test or goes isolation. Therefore, no extra information is revealed by the system.

A tech-savvy adversary can attempt to re-identify infected individuals based on proximity with a large number of devices at multiple locations. Due to the nature of the BLE broadcast, this adversary can identify the infector regardless of the design of the contact tracing system.

A more frequently rotated ID scheme will increase the difficulty of re-identification. However, increase the frequency will also increase the power and storage requirements of individual's devices. It's a trade-off which that be decided by the implementor.

Conclusion

In this document, we describe the contact tracing algorithm used by the social distancing application form Taiwan AI Labs. We discussed not just the design of the algorithm but also its limitations and potential security and privacy concerns.

Our decentralized design fully respect an individual's decision and privacy. It also minimizes the data required to perform such tracing. The design gives users fine-grained control over the information they shared and all data sharing happens under the user's explicit permission.

References

1. Decentralized Privacy-Preserving Proximity Tracing
2. Bluetrace: Privacy-Preserving Cross-Border Contact Tracing
3. Privacy-Preserving Contact Tracing by Apple/Google
4. COMMISSION RECOMMENDATION (EU) 2020/518
5. Contact Tracing Mobile Apps for COVID-19: Privacy Considerations and Related Trade-offs
6. ACLU WHITE PAPER: THE LIMITS OF LOCATION TRACKING IN AN EPIDEMIC