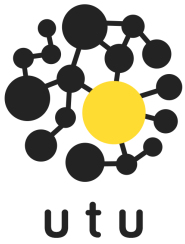

Privacy-Preserving COVID-19 (Non-Tracking) Contact Tracing

Working Doc

Dr. Bastian Blankenburg, UTU Technologies Limited, bastian@utu.io

Introduction

Many governments have introduced or are considering COVID-19 tracking apps for their citizens to use (mandatorily in some places). The efficacy of such apps has recently been studied and confirmed by a team of the Univ. of Oxford.

Currently there exist a number of surveillance-style apps which track each user including personal identifying as well as location information, and which are therefore very open for misuse by governments, agencies or third parties which gain illicit access to such systems.

To counter this, some proposals for decentralised apps which keep sensitive user information such as identifying and location information local to each user's phone, while still allowing them to be warned should they have come close to a person which was (later) diagnosed with COVID-19.

Such apps generally work by utilising Bluetooth Low Energy technology, which allows for continuous advertising of and scanning for small bits of data, such as generated and anonymous IDs. We now describe how such apps work in general.

Bluetooth Low Energy-based Decentralised Tracking Apps

All of the existing proposals decentralised tracking apps are based on Bluetooth Low Energy technology, because it has the following favourable properties:

- It allows for energy efficient continuous broadcasting of small bits of data.
- It allows for energy efficient continuous scanning for others' advertisements.
- It is available in virtually every smartphone made in recent years.
- It supports distance estimation to other devices using RSSI.

Given this, such apps then generally work as follows:

1. The app generates some ID randomly.
2. Each user's app starts advertising this ID to nearby devices via Bluetooth Low Energy, and starts scanning for other such advertisements at the same time.

3. If the app recognises the presence of another app's id, it stores this information locally on the user's phone, together with a timestamp and possibly a location reading.
4. If one of the users is diagnosed positive with COVID-19, this is recorded in the user's app, which will then invoke some mechanism to notify other users about this fact, optimally only those to whom the infected user came close in some recent timeframe.
5. The apps of users who recently came close to a user who was diagnosed to have COVID-19 compute some probability that their user was infected, and possibly warn them.

Existing proposals differ in a number of ways, most prominently in these points:

- How the IDs are generated, and whether they are constant or changed.
- How diagnosed cases are recorded and validated in the system.
- How other users (apps) who came close are identified and notified.
- How the probability of infection is computed.

We review a few of such approaches in some detail in the following sections, but we summarise here that none of the existing approaches offers both maximum privacy-preservation and scalability at the same time. Either the proposed protocols are prone to eavesdropping attacks by well-resourced or colluding parties, or they lead to far too great amounts of computation (and thus battery drain) and of data to be downloaded.

Our proposal will enable both the required scalability as well as maximum privacy preservation including the mentioned eavesdropping attacks.

Further, none of the approaches tackles the situation which is prevalent in the developing world, and specifically most African countries: that large parts of the population don't use smartphones, but feature phones. To this end, we propose to use other, non Bluetooth-based tracking approaches -- which are expected to be less effective, but a best-effort approach is likely still better than none at all -- for these populations, and how such a system could be interfaced with a BT-based one without further compromising the privacy of users of either system.

Finally, apart from Yoshua Bengio's approach (see below), none of the existing ones includes a proposal to use machine learning to improve the prediction of the risk of infection for users of the apps. We show how Bengio's approach can be adopted in our solution without compromising its maximum privacy-preservation.

Existing Proposals for Decentralised Tracking Apps

Yoshua Bengio's Approach

Machine learning expert Yoshua Bengio proposed a decentralised app which supposedly enables people to compute their own probability of infection by using a machine learning approach.

His approach doesn't actually rely on notifying others about users newly being diagnosed with COVID-19, but on an ongoing evaluation of one's own probability to be infected, which is exchanged in real time with other present apps.

For this, some anonymised data sharing with a central server is necessary:

> *upload anonymized information (not your GPS trajectory but the sequence of encounters and the associated risks) to a non-governmental data trust collecting a dataset for training the risk predictor and in return, you would obtain an updated version of the app risk predictor making more accurate predictions.*

Shortcomings

- Because of the missing notification of other users in case of a newly diagnosed COVID-19 infection, this approach doesn't help that much with silent spreading.
- It is not clear what data exactly needs to be uploaded, and what the conditions for anonymisation are, and what the risk of later anonymisation is should more data be learned.
- It is not clear how IDs are managed, whether they are sent to others, and whether they are renewed or kept constant. The problem with unchanged IDs which are continuously advertised is that this allows for eavesdropping: someone (like the government) might use multiple devices in multiple places to track occurances of the same IDs, and therefore, e.g. in combination with video cameras, be able to de-anonymise individual users.
- Diagnosis results are supposed to be reported only by medical personnel; while this might prevent fraudulent misreporting, it would also prevent the app working with (hopefully) upcoming home testing kits.

DP3T

DP3T is an approach developed by some academic institutions led by EPFL (it had temporarily been adopted officially by EU-endorsed PEPP-PT, but not anymore). Its main feature is the notification of other users who came close to newly diagnosed cases and are likely to have contracted COVID-19 themselves. Quoting from their Overview of Data Protection and Security document, it works like this:

> *1. **Installation**: the app is installed, generates a secret seed (SK), and derives Ephemeral Bluetooth IDs (EphIDs) from it*
>
> *2. **Normal operation**: each app broadcasts EphIDs via bluetooth, and records EphIDs that are broadcast by other apps in the vicinity.*
>
> *3. **Handling infected patients**: after patients are diagnosed, and only with their consent and with authorization from a health authority, they upload specific data from their phone to the backend server. From this data, the identity of the patient cannot be derived by the server or by the apps of other users (see below), it is nearly anonymous. Before this point, no data other than the broadcast EphIDs leaves the phone.*
>
> *4. **Decentralized contact tracing**: each app can use the data from the backend to locally compute whether the app's user was in physical proximity of an infected person*

> *and potentially at risk of infection. If they were, the app can inform the user to take action.*

Therefore the eavesdropping problem mentioned in the previous section is supposedly solved by frequent ID regeneration from a secret seed.

Shortcomings

Unfortunately though, this secret seed is not really secret at all. As is described in the more detailed sections of the above-linked document,

> *The app opens an encrypted TLS connection to the server and sends the authorization code and its seed SK, a compact representation of the EphIDs it has broadcast during the infectious window, to the backend*.

And then even:

> *Several times a day, the backend server sends newly received EphID seeds of infected patients to all installed proximity tracing apps via the notification service.*

So this means that ultimately, every user's app will learn *all of all* infected users' IDs, again enabling eavesdropping and de-anonymisation as described in the Shortcomings sub-section of Yoshua Bengio's Approach.

They counter this in the "Eavesdropper" section under "Thread model" on page 21 of the whitepaper thus:

> *It should be noted however that in many instances, for individuals or companies to use data in this way, such as to collect data about passers-by to try and estimate their infection status based on the announced identifiers, will fall foul of a range of existing national and European laws around data protection, ePrivacy and computer misuse.*

Apart from inconsistency of relying on laws and regulations for a crucial security aspect of a system whose goal is otherwise to implement privacy-preservation on a technical level, this limits the protocol's applicability to the EU and jurisdictions with similarly strict jurisdictions (i.e. very few). Moreover, some kinds of larger scale eavesdropping attacks would be very hard to detect or prove, such as the eavesdropping by security companies we describe later.

Also, and again similarly to the former approach, the requirement of authorization from a health authority again limits the app to lab-diagnosed cases, and prevents the app from working with (hopefully) upcoming home testing kits.

Apple/Google Privacy-Preserving Contact Tracing

Apple and Google announced a joint contact tracing approach which they will make available as an app from mid-May, later as part of iOS and Android themselves. It's architecture is generally similar to the PEPP-PT approach, but they use additional daily keys, which are derived from the original secret key, and which are sent to the server in case of infection instead of the secret key itself.

By sending only a fixed number of daily keys, this allows them to limit the set of rolling keys which become known to other users to only those which were generated in this time period.

Shortcomings

The approach has the same eavesdropping and de-anonymisation potential as the the PEPP-PT one, even if it is limited to a fixed time period (this will be configured to 14 days) before the infection is reported.

Additionally, they also mention

> *Second, in the coming months, Apple and Google will work to enable a broader Bluetooth-based contact tracing platform by building this functionality into the underlying platforms. This is a more robust solution than an API and would allow more individuals to participate, if they choose to opt in, as well as enable interaction with a broader ecosystem of apps and government health authorities.*

Though they do not specify which data might be shared with “with a broader ecosystem of apps and government health authorities”, how this will be permissioned, or any other details on this.

MIT

Some days prior to Apple and Google, MIT proposed a very similar approach. It doesn’t use the daily keys, but uploads the ephemeral ids, here called “chirp ids”, of the last 14 days directly to the central server.

Shortcomings

The approach has the same eavesdropping and de-anonymisation potential as the Google/Apple one.

WeTrace

WeTrace is an open source proposal and proof of concept from members of the AirGap team. On a high level, its architecture is quite similar to the PEPP-PT approach. However, it employs concepts used in privacy-first blockchain projects such as the ZCash project to implement a secure notification mechanism, thereby overcoming the privacy shortcomings of PEPP-PT. The process works as follows (quoting from their whitepaper):

> *1) Every device that installs the WeTrace app generates an asymmetric key pair using elliptic curve cryptography. For this example’s sake PKA stands for public key of Device A and SKA stands for secret key of Device A. So in this step we generated PKA, PKB, PKC and respectively SKA, SKB, SKC.*
>
> *2) Every device starts broadcasting to its surrounding devices their PK. This is also their unique identifier.*

3) When now 2 devices (i.e. A and B) meet in close contact, Device A knows PKB and Device B knows PKA. Both devices store from the contact besides the PK. Also: a timestamp, the geolocation where the encounter happened.

4) When User A is now infected and wants to report it, the Device will go through the list of close Contacts and encrypt a message with the public key of every contact. In our case, it will be encrypted once with PKB because this was our only contact. All those messages will be sent to the central backend that will relay them to all devices. The messages will contain the data that User A chose to share, so either only the fact that an infection happened, or additionally when or even where it happened. Important: Only the reporting user decides if he/she wants to share this information.

5) Device B and Device C receive from the backend a no- tification telling them that new reports have happened. Device B will then try to decrypt every message with SKB and will eventually find out that a message was directed at him/her. Device C will do the same, however because no message was encrypted with P KC , no message can be decrypted.

So the deciding difference to PEPP-PT is that using asymmetric cryptography (specifically, elliptic curve cryptography) to encrypt messages with keys to which a newly diagnosed user came close to with their public keys, the secret seed or previously used public keys never have to be shared with other users. This removes the possibility for an eavesdropping attack as outlined above.

Shortcomings

Even with the secure communication which removes the de-anonymising eavesdropping possibility, WeTrace still employs a central server as a communication relay. This could be a bottleneck which might be open to censorship or (D)DOS attacks. But it also means that without further provisions, the server knows which messages come from which IP address, and therefore the same user. Note that this linking activity is then not only available to the server operators and network hosting company, but also to state agencies or hackers gaining access to the system.

And even if we limit the readability of advertised rolling keys to users which have actually come close to an infected user, there is still potential for eavesdropping: one might place a number of devices running the app in different locations, and therefore be notified with all their advertised keys that they've been near the diagnosed person. If one gets multiple notifications at the same time (i.e. these messages have been written to the smart contract at around the same time), one might assume with some probability that they concern the same person.

This is not such an unlikely scenario; for instance, many publicly accessible places in Nairobi, such as malls, use the same small set of available security companies. These usually do a security check of every person entering the location, which implies their staff comes very close to every visitor. If the security company installs the app on all their staff's phones, it will then be easy for them to collate all their seen visitor ids and so trace persons which later are reported as

infected through all their secured locations. They might also collude with their competitors to get an even more all-encompassing picture.

One way to remedy this might be to delay sending the report to each of the recipients with some randomised time. It's then a trade-off: the smaller the time (on average), the more vulnerable to a linkage attack it is. The greater the time, the more average delay before the recipients get the message and can self-isolate.

However, to also prevent linkability by parties with access to the server, it is necessary to not only decouple messages time-wise, but also IP-address-wise. Therefore, additionally to stretching out messages over time, they'd also needed to be sent from different IP addresses in networks different than the one the server is in, e.g. using different VPN connections with changing exit nodes over time, or using the TOR network.

Also, while the self-reporting of users allows for home testing kits, it opens up the possibility of fraudulent false reporting of cases. For example, a person could go to some place where they know to find some other person they want to sabotage, come close to them, and then self-report a positive diagnosis. To mitigate this, Alessandro De Carli of AirGap has suggested in a call that a proof of work mechanism could be added as a requirement to report, which would make reporting somewhat costly and thus prevent more rational actors from defrauding (but not irrational ones who would still pay the price for sabotaging someone else).

TCN Protocol

The TCN Coalition proposes TCN Protocol, based on Temporary Contact Numbers, which are basically the same as the rolling ids in the PEPP-PT/Apple-Google/MIT approaches. Similarly to those other protocols, TCNs are generated from some seed. When a user is diagnosed positive, a report including the used seed is sent. This is similar to the PEPP-PT approach on first glance, but their protocol allows for creating new seeds and then submitting multiple partial reports rather than a single report for their entire history. Since reports are not linkable, this prevents associating TCNs across reports to the same user.

Shortcomings

When new seeds are not generated very often and thus not split into many parts, this approach might have similar drawbacks as the PEPP-PT/Apple-Google/MIT approaches.

OTOH if new seeds are generated often and reports split accordingly, it might be closer to WeTrace's approach in terms of resiliency against eavesdropping.

The motivation to use seeds instead of single TCNs is given as scalability, though no concrete analysis of expected report numbers is given.

The TCN documents further includes a list of attacks.

We note that a "Linkage Attack" is in principle possible with all tracking apps. The simplest case to see this is if a user came close to only one person over a number of weeks, and then gets a warning that they got close to an ifected user, it is clear that it could only have been that one other person.

Fully Decentralised Approach with Maximum Privacy

Here, we propose to combine the best ideas of the existing approaches, as well as a few additions with the purpose of arriving at a still more secure and privacy-preserving protocol.

We consider the WeTrace approach as the best so far as it is the only one which can effectively protect against eavesdropping using the public/private key cryptography-based notification scheme.

Privacy vs Scalability

The superior privacy properties of the WeTrace approach comes at the cost of number of messages, and so en- and decryptions that have to be done by each app: for each reported case, a message is encrypted and sent for every other public key this person's app came near to, and then each app needs to examine (download + try to decrypt) all these messages.

This is why other approaches like TCN, DP3T and Apple/Google send seed keys instead of the full set of advertised ids over the last 14 days. But none of these approaches present an analysis of the actual expected number of messages or number of people which have to be notified, and so whether scalability is an actual problem, and not just an imagined one.

Let's try to get some idea of actual numbers to expect. Even if our proposed solution is applicable world-wide, we're developing this from our own context, which is East Africa, generally with African smart phone users in mind. A study "Characteristics of human encounters and social mixing patterns relevant to infectious diseases spread by close contact: a survey in Southwest Uganda" find that each person has about 100 contacts on average and about 350 maximum in 14 days. We believe this maximum could serve as an upper bound for our setting: in urban places like Nairobi, many people of the lower middle class have smartphones, but still use matatus (public transport). But the upper middle and upper classes mostly use taxis and private vehicles, therefore should have less longer-lasting close contact.

Therefore, each positively reported case has to encrypt and send max. 350 messages. Since reporting a newly positively tested result only happens once (or very few times) for a user, the encryption and sending part is not an issue, even if it turns out to be many more messages for some users.

However, the number of messages to download and try to decrypt depends on the number of reported cases per day. Currently, the max. number of new cases per day worldwide was just below 100,000. So if we (naively) assume that our app will be deployed worldwide using one and the same server, this would mean that each user has to download and try to decrypt 35,000,000 messages per day. Apart from using a lot of computation and thus power and battery drain, this could amount to more than 1.6GB downloads per day, also clearly too much. Additionally, the number of reported cases per day is expected to still grow in many countries, particularly in Africa, and so these numbers might be expected to further grow.

To mitigate, we propose to not use one global server, but instead use multiple servers for different geographical locations. Each such server would correspond to a specific region, and a

reported case would need to upload each message only to that server which corresponds to the region that the contact happened in.

Similarly, each app would only download new messages from servers whose corresponding area they have visited in the last 14 days.

Depending on the granularity of the geographic subdivisions, with current movement restrictions, this would likely amount to very few contacts that each user if interacting with. The more granular, the higher the number of servers, but also the smaller the number of messages per server.

Note that this would reduce the degree of anonymity from a k-anonymity with a k = number of worldwide users for the global approach to k = the number of users in the same area (i.e. reporting to the same server). But assuming that each such area would contain at least some thousands of users, this shouldn't be a problem.

Using Smart Contracts instead of Servers

Using a central server as proposed by WeTrace and other approaches has some drawbacks, as we mentioned earlier in the "Shortcomings" section there. We propose to prevent some of these problems by using smart contracts running on a public blockchain instead of public servers.

Then diagnosed patients could make the app announce their own newly-diagnosed infection by calling a suitable function on the smart contract, uploading its own public keys encrypted with the public keys of those other apps to which it came near at around the same times. Basically exactly the same as in the WeTrace protocol, but using the smart contract instead of a central server.

By using different (pseudonymous) accounts (or "wallets") on the blockchain for sending reports to each close contact, and spreading out sending of the messages over some time, linkability of the messages can be disrupted. Account creation and management purely for this purpose can be managed within the app.

The app would then regularly check for any new uploaded messages on the smart contract, and try to decrypt each one with its secret key. It would thus be able to decrypt only those for whom it is the supposed receiver.

Report validation

We mentioned earlier that report validation is problematic when users are allowed to self-report without further requirements. We are so far not aware of complete solutions to this problem.

For this reason, at least at this point in time, until home test kits are widely available, it seems best to us to require some sort of medical personnel authorisation. In the fully decentralised protocol, this could be done by medical institutions signing the messages sent to the smart contract. Each authorised institution would obtain a signing key, whose own authenticity could perhaps be validated by keeping the corresponding public keys also in the smart contract.

A diagnosed user's app would then first compose the messages and send them to the medical staff's app (e.g. via QR code). The latter would then be allowed to sign the messages with the

institution's key, and send them to the smart contract together with the signature. The receiving users could then validate the signature using the above mentioned authorised signing key list.

However note that in this case, the staff who is signing the transaction would then need to do the time-decoupled sending of the messages, and also they themselves could link them. Likely there exist still better mechanisms which do not have these constraints, so we shall do some further research here to find the best solution.

Another idea (originally given by Leah Stuhltrager of æternity) is that home test kit manufacturers could include one-time use tokens with each kit. This way, it could be ensured that only one result pet kit could be reported. However, a fraudulent user could still obtain a test kit and mis-report the result in this case.

Catering for Non-Smartphone Users

One aspect of the proposed Bluetooth-based system is that it caters only for smartphone users. However, this is actually only a minority in most African countries. Other kinds of systems are developed for such populations, such as performing interviews with infected persons which typically includes questions about their travel histories.

One might consider not only asking for travel histories, but generally for specific locations such as concrete matatus (public transport vehicles) and shops. Then, people who had been in the same locations, might be warned via some mechanism that ideally also preserves the privacy of involved persons as much as possible. For example, public bulletins containing a list of locations for each day where and when an infected person has visited.

Additionally, such locations could have someone with a smartphone — or a stationary smartphone — which has the BT-tracing app, and notify other app users who had been there. In this case, only that stationary app would reveal itself to be at that location, but this should be considered public knowledge anyway.

For the other way around, i.e. when an app user visits a shop with such a stationary app, and later gets infected, is more tricky if we don't want the user to reveal that they've been there. Because the stationary app would have a timestamp of the contact, they could easily de-anonymise that user if e.g. it was the only one in the shop during the lifetime of the stationary app's id. However, we could allow users to withhold reporting to users (public keys) when they've not been close to other apps around the same time/space.

Improving Risk Prediction Accuracy with Machine Learning

App users could additionally share information with a machine learning service as proposed by Yoshua Bengio, to get a better estimation of their own likelihood to have contracted the disease. One way to do this while fully preserving privacy is to let this system know only about unlinkable datapoints about each report. Or employ some other existing zero-knowledge ML approach.

Further advice might be found e.g. in https://blog.openmined.org/covid-app-privacy-advice/

Voluntary Sharing of Additional Anonymised Information with Trusted Institutions

Similarly, such information might be voluntarily shared with third parties in an anonymised way, such as trusted (by the user) medical or government institutions. Each user should be able to specify their minimum k-anonymity requirements, as is commonly used for medical studies. The anonymisation mechanism shall then be designed in such a way that each user's minimum k-anonymity is preserved.

Having such anonymised data, we can then build dashboard and other high-level analytic tools on top of this, which can be used by government and other agencies without compromising citizen's rights and privacy.

Impact Objectives

le Polain de Waroux et al. (BMC Infectious Diseases (2018) 18:172) report that the mean number of human contacts per day in Uganda is 7 and the maximum is 25. Therefore, in the previous 14 days an infected patient may have come into contact with 100-350 people. Contact tracing has been shown to be key in achieving epidemic suppression e.g. in China and South Korea. However Ferretti et al., Science 10.1126/science.abb6936 (2020) show that it is essential to minimise delay in contact tracing due to the fast transmission rate of Covid-19. Using their analysis we estimate epidemic suppression could be achieved by 60% usage of our digital contact tracking system. However even lower rates than this will save many lives. Based on known data of Covid-19, 5% of infections require intensive care hospitalisation (Ferretti et al., Science 10.1126/science.abb6936 (2020)). In the context of insufficient or overloaded healthcare systems in Africa most of these 5% of infections may be fatal. Reducing the reproduction rate (the number of people an infected person infects) from R0=1.7 to R0=1.2 would halve the number of deaths. Assuming this worst case scenario of 5% mortality rate, reducing R0 in this way could save 25 million lives (see figure).

Additional Information:

GitHub - https://github.com/AfricavsVirus/Privacy-preserving-COVID19-non-tracking-contact-tracing

UX Prototype - https://balsamiq.cloud/slf74my/pduq61p/r2278

User Journey -
https://docs.google.com/document/d/11vslQqFVRDMI6grfgXE7v9P-GdKY1UNAij1QtStfAFg/edit