Privacy Issues for Apple-Google Exposure Notification Mechanism

Adam Bobowski, Jacek Cichoń, Mirosław Kutyłowski

Politechnika Wrocławska

Anty-CIVID, PAN, 23.6.2020

Idea

Infection chain

- Find who could be infected by a COVID-19 positive person: isolate and prevent further infections
- the time and effectiveness is critical:
  - manual processing bottleneck in case of a serious outbreak
  - a suspect person already could infect further people

BLE signalling

- the potential infection exposures deduced from proximity of smart phones of the people concerned
- contact detection by receiving identifiers sent over BLE – low energy and short range signalling

Contract tracing

Mechanism

- all identifiers captured by the smart phones uploaded to a central server (e.g. Robert-Koch-Institut)
- in case of a positive test diagnosis instant derivation of exposed people

Privacy

no privacy protection, all kinds of misuse possible

Social acceptance

possible if most people unconditionally trust the authorities regarding their honesty and competence

UK, June 18: *UK gives up on centralized coronavirus contacts-tracing app – will 'likely' switch to model backed by Apple and Google*
Norway: forbidden by data protection authorities, system shut down

Exposure notification

Mechanism

- a **central entity**
  - processes anonymized data obtained from smart phones of COVID-positive users,
  - creates data for downloading – for recomputing the identifiers corresponding to infectious persons
- **app of an end user** may download the data, make computations and warn:

  "you have been exposed to COVID-19"
- **after notification:** it depends ... (user's decision or automatic upload by the app)

Properties

- no instant identification of exposed people
- privacy taken very seriously – better chances for social acceptance while a high number of users is critical for success

Architecture proposed

App

- creates and broadcasts pseudorandom identifiers
- receives and stores pseudorandom identifiers from the smart phone's proximity
- downloads the data (blacklists) from Diagnosis Server and checks against stored identifiers
- in case of a match notifies about infection exposure:
  - the user
  - Diagnosis Server ??
  - the health authorities??

Diagnosis Server

- collects data from exposed users
- prepares data for downloading

Implementation

Google-Apple for App:

implemented:

- key functionalities in the operating systems
- API for creating apps

The app itself should be created by a third party.

Diagnosis Server

To be created and run by a third party.

- Flexible building blocks rather than a fixed system.
- Interoperability easy to achieve.

Time, Temporary Exposure Key

- 10 minute time periods
- period ID: 32-bit index ***ENIntervalNumber*** based on UNIX Epoch time:

$$\mathit{ENIntervalNumber}(\mathit{timestamp}) = \mathit{timestamp}/(60 \cdot 10)$$

- ***TEKRollingPeriod*** consists of 144 periods (= 24 hours)
- For the *i*th *TEKRollingPeriod* the smart phone generates a 16-bit *Temporary Exposure Key*:

$$tek_i = \mathrm{CRNG}(16)$$

(CRNG = Cryptographic Random Number Generator)

Keys for a single day

that is, for a *TEKRollingPeriod*

2 keys associated to the ith *TEKRollingPeriod*:

- ***Rolling Proximity Identifier Key***:

$$RPIK_i = \mathrm{HKDF}(tek_i, \mathrm{NULL}, \mathrm{UTF8}(\textit{EN-RPIK}), 16)$$

  where HKDF is a hash key derivation function.

- ***Associated Encrypted Metadata Key***:

$$AEMK_i = \mathrm{HKDF}(tek_i, \mathrm{NULL}, \mathrm{UTF8}(\textit{EN-AEMK}), 16)$$

Identifiers for 10 minute periods
broadcasting activity

- randomized BLE MAC address used
- pseudorandom ***Rolling Proximity Identifier*** for the *TEKRollingPeriod* i and a period j:

$$RPI_{i,j} = \mathrm{AES128}(RPIK_i, PaddedData_j)$$

where $PaddedData_j$ is the following 16-byte string:

$$PaddedData_j = (..., ENIntervalNumber(j))$$

- ***Associated Encrypted Metadata (AEM)*** – an AES Counter Mode ciphertext:

$$\mathrm{AES128\text{-}CTR}_{AEMK_i}(RPI_{i,j}, Metadata)$$

Positive diagnosis

The app of infected person

sends to the Diagnosis Server tek_i keys for chosen days

Diagnosis Server

Collects all *tek* keys and puts them on a blacklist of diagnosis keys

Notification

A user's app:

- periodically downloads the current list of diagnosis keys
- recomputes the corresponding rolling proximity identifiers and check for their presence in own list of anonymous contacts
- if positive, derives *AEMK* key and decrypts associated encrypted metadata

Linking

Remark: all smart phones change the identifiers (and BT MACs) synchronously

goal

find out if two rolling proximity identifiers belong to the same person

situation

unless HKDF broken, the attacker has the same chances as in case of random rolling proximity identifiers chosen independently at random each 10 minutes

Remaining risk: e.g. only if a persons *A* in range, then no change of identifier can help against linking

Random BLE Signalling

goal

flood with fake rolling proximity identifiers

attack

broadcast messages that have the same format as those generated by legitimate apps:

- a receiver cannot see any difference and would store them
- uploading diagnosis keys impossible if the fake rolling proximity identifiers created at random

attack impact

DDoS on phones, impossible to eliminate fake data

Sending to quarantine

goal

send a target group of people to quarantine, e.g.

- to prevent anti-government demonstrations
- slow down competition projects by eliminating their key staff

attack

create a fake app that disseminates RPI and then declares itself as infected

- anonymity prevents checking infection declaration

attack impact

unlimited if

- Diagnosis Server under control of adversary, or
- Diagnosis Server honest but no authentication of the apps

security of an app is not enough!

Replay Attack

goal

increase the number of of exposure notifications

attack

record RPIs and AEMs and replay them elsewhere at a different time, e.g.

- collect data in a hospital or any high risk environment
- replay them at a place with a high congestion of people

attack impact

does not work with Google-Apple: an RPI invalid after the 10 minutes slot

Relay Attack

goal

increase the number of of exposure notifications

attack

relay the RPIs and AEMs into a different location and broadcast them immediately

attack impact

it should not work with Google-Apple if metadata contain location checking mechanism:

- the metadata are encrypted, so it should be infeasible to manipulate the ciphertext
- . . . however AES counter mode is used!
  the worst choice when concerning resilience to manipulations

an authenticated encryption mode should be chosen

CPRNG

goal

break anonymity

attack

implement CRNG so that its output can be predicted. E.g.:

- kleptographic CRNG
- the CRNG seed retained by the manufacturer

attack impact

all RPIs can be recomputed by the attacker and compared with the RPIs sent by the smart phones

absolutly no privacy for attacked users

CPRNG
user's watchdog

Goal

the user must be allowed to change *tek* keys so that

1. the final *tek* keys are unpredictable for the adversary controlling CPRNG
2. the user cannot enforce any particular form of *tek* keys (the user may attack own app)
3. the user can check that the presented method has been really implemented

CPRNG - Example protection mechanism

Setup (or Reset)

- The user inputs a random seed u (maybe generated by another app or device).
- u retained outside for control purposes.

Controlled operation

- apart from tek_i the smart phone computes also tek_i^{ctrl}
$$tek_i^{ctrl} := \mathrm{HKDF}(u, i)$$
- apart from $RPI_{i,j}$ the smart phone computes
$$RPI_{i,j}^{ctrl} := \mathrm{HKDF}(tek_i^{ctrl}, j)$$
- the modified RPIs are broadcast:
$$RPI_{i,j}^{mod} := \mathrm{Hash}(RPI_{i,j}, RPI_{i,j}^{ctrl})$$
- the app presents $RPI_{i,j}$, $RPI_{i,j}^{ctrl}$ to the user

CPRNG - Example protection mechanism

Control

- the user recomputes $RPI^{mod}_{i,j}$ as $\mathrm{Hash}(RPI_{i,j}, RPI^{ctrl}_{i,j})$ and compares with the value from the BLE channel

Uploading to Diagnosis Server

data to be sent:

- the keys tek_i from critical days
- the corresponding tek^{ctrl}_i keys

Exposure checking

as before, but RPI's recomputed according to the modified formulas thank you for your attention!