IDPT-FP (IDPT, Full Protocol)

Jorge García Vidal, UPC, BSC

Draft version May 25th 2020

This is a proposal for a digital proximity tracing protocol that can operate both in centralized and distributed mode with full interoperability. It is based on the mechanism described for the IDPT protocol[1] for interoperability between ROBERT[2] and DP3T[3] applications.

Assumptions

We assume that in the same geographical area (e.g. the Schengen area) we have a digital proximity tracking application with users who can decide whether the risk score is done on a central server (C-users), or is done on the user's phones (D-users).

It may be that in a country within this geographical area, national public health authorities choose to support only C-users or D-users. Another option is that they give freedom of choice to their citizens, who assess the trade-offs between privacy and security and the effectiveness of the risk score.

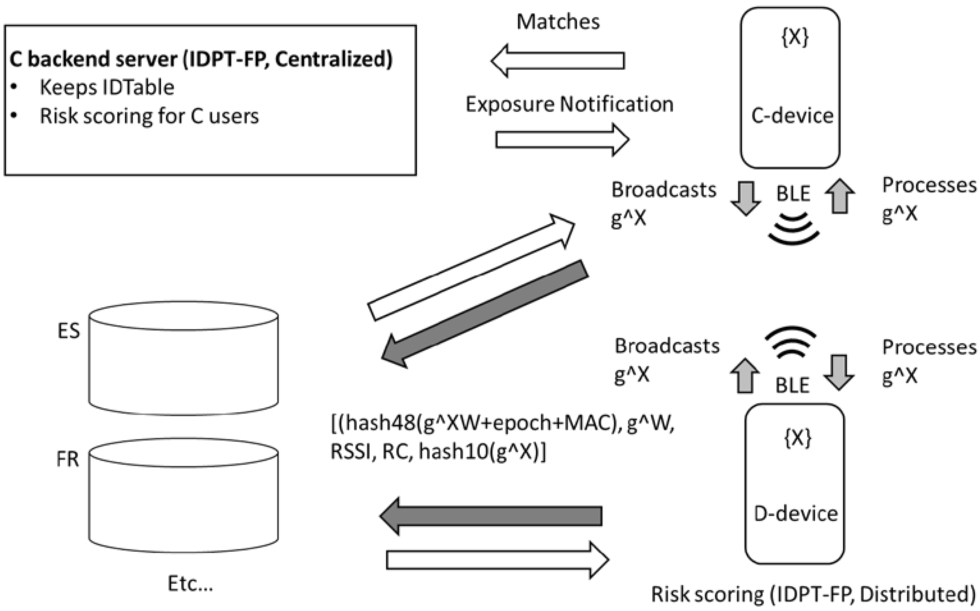

Figure: Functional elements

[1] https://github.com/IDPTdocs/documents/blob/master/IDPT-v2.pdf
[2] https://github.com/ROBERT-proximity-tracing/documents/blob/master/ROBERT-specification-EN-v1_0.pdf
[3] https://github.com/DP-3T/documents/blob/master/DP3T%20White%20Paper.pdf

Beacon broadcast

- All users broadcast BLE ADV_IND packets (which we call "beacons") with a payload equal to g^X, as in a Diffie-Hellman exchange, where X is a secret number that is changed at each epoch (e.g. 15 minutes).
- We assume that g^X is a 16-byte number.

g^X generation

For C and D users, g^X is generated in the devices, which keep the sequence of secret values X.

Periodically during every epoch, the devices insert an special beacon with content equal to “0^{16}+RC+MSB96(g^X)”, that allow the identification of the RC[4] of the transmitting device. Receiving devices use these beacons to assign a RC to a g^X. If no identification is possible, due to the fact that these beacons are not received, a RC=”Schengen” is assigned.

Beacon processing

All users retain the g^X values received, as well as the epoch, the RSSI and the MAC of the received beacon, and the RC assigned to the g^X.

The C-backend server

The risk scoring for C-users is run at a the C-backend server, which keeps an IDTable with similar structure as the IDTable in ROBERT backend servers.

The matching with users who tested COVID+ is done at the C-devices, though. This avoids a high computational load in the server and minimizes the sharing of information.

Users with tests COVID+

C and D devices choose a secret random number W, different per each received g^X.

Both C and D devices insert the list of tuples [(hash48(g^XW+epoch+MAC), g^W, RSSI, RC, hash8(g^X))] into a global public list, so that the identity of the device remains anonymous. The g^X inserted in the list should fulfil some criteria (e.g. the device has received a min number of beacons with a min value of RSSI for the values g^X) in order to avoid reporting brief contacts with very low risk of transmission.

The tuples for both C-users and D-users are kept in the same public list with a keep-alive time of 1 day. The global list is organized depending on RC values to reduce the download traffic for devices; see section “Complexity”.

[4] Region Code, group a number of users with a size (e.g. 1.000.000) that ensures a reduced size of the lists that must be downloaded by devices when checking for contacts with users with test COVID+. Can be assigned based on geographical criteria (e.g. “Barcelona”) or at random (e.g. “Spain27”), and could be dynamically changed according with the evolution of the pandemics.

Exposure notification

C and D devices periodically reads the public list [(hash48(g^XW+epoch+MAC), g^W, RSSI, RC==”user_region_code” or “Schengen”, hash8(g^X))]. Then they look for intersections of hash((g^W)^X’), for the X’ values stored in the device, with the hash values of the list [(hash48(g^XW+epoch+MAC), g^W, RSSI, RC==”user_region_code” or “Schengen”, hash8(g^X)==hash8(g^X’)].

C-users

The results of these intersections are uploaded to the C-backend server, which updates the IDTable entry for the C-user. A risk scoring algorithm re-evaluates then the risk score per user of the IDTable.

Note that the information sharing with the C-backend server is kept to a minimum. The C-backend server does not know the identity of the users (C or D) who tested COVID+ who were in contact with the C-devices, as the W values are kept secret in the devices.

The C-devices periodically connect to the C-backend server for receiving an Exposure Notification.

D-Users

A Risk Scoring is then run in the D-device that decides whether the D-user is notified as in risk of exposure.

Complexity

Every device must download a list [(hash48(g^XW+epoch+MAC), g^W, RSSI, RC==”user_region_code” or “Schengen”, hash8(g^X))]. Assuming that a user reports 100 significant contacts per day (i.e. 4 per epoch), and for the RC the number of users who test COVID+ per day is 500, the list would have 700 000 entries + # entries with RC==”Schengen”.

The number of computations require for finding intersections is considerably reduced if devices match the values hash8(g^X)==hash8(g^X’) before doing the checks. As the device stores 14*24*4 keys, it means that in average we must perform 5.25 checks per element of the list.

Defences again other attacks

The use of hash(g^XW + epoch + MAC) provides a defence against replication attacks.

Consequences of breaking Diffie-Hellman

If an attacker breaks DH (i.e. obtains W from g^W), she could check if a g^X that has been eavesdropped matches the hash48(g^XW+epoch+MAC) value. This would lead to the conclusion that the user who transmitted g^X was close to a user who reported a COVID+ test. This vulnerability is inherent in all digital proximity tracing protocols, and the attacker can obtain this information in a much simpler way.

Possible implementation issues

- The current Gapple EN API would not support this mode of operation.
- Is 16-byte Diffie-Hellman too weak?. We do not think so, as the information that an attacker can obtain from breaking DH is not worth the effort. However, in case this is considered a risk, using 32 bytes DH and the corresponding consequences on beacon transmissions should be considered.

Privacy properties

D-users avoid re-identification attacks of distributed protocols such as DP3T. C-users share less information with the C-backend server than in the case of ROBERT.

Acknowledgement

The author would like to thank Leonardo Bautista-Gómez, Pedro Martín-Jurado, and Pablo Rodríguez-Rodríguez for useful discussions on the properties of this protocol. The fact that IDPT can be used as a full digital proximity tracing protocol was clear from the beginning, but the author made no effort to work out the details until Pablo pointed out that this protocol would probably not suffer from the re-identification problems of other distributed protocols. However, any errors in the text are the sole responsibility of the author.