Coronalert: A Distributed Privacy-Friendly Contact Tracing App for Belgium

Corona App Task Force

Version 1.3

17 August 2020

Summary

This document describes the Coronalert contact tracing app that will be deployed in Belgium in September 2020 to help control the next waves of COVID-19 infections. It starts with a motivation for the development of a contact tracing app, followed by a discussion of the requirements. Next, a high-level introduction is presented of the DP-3T architecture (Distributed Privacy Preserving Proximity Tracing) in combination with the Exposure Notification interface provided by Google and Apple. Subsequently the technical details of the protocol are described as well as the international interoperability and the monitoring options.

History

The DP-3T Protocol (Distributed Privacy-Preserving Proximity Tracing) has been created in March 2020 by an international consortium of technologists, legal experts, engineers and epidemiologists, led by EPFL; KU Leuven is part of this consortium. The goal of this team is to bring together wide range of experts who wanted to create an effective proximity tracing technology that does not result in mass surveillance capabilities (https://github.com/DP-3T/documents). Since May 2020, a variant of the DP-3T protocol is supported in Android and iOS by the Exposure Notification Interface developed by Google and Apple. More than 15 countries have deployed or plan to deploy this solution. In April 2020, the work has started on integrating this protocol into the Belgian healthcare context. In May 2020, a task force has been created under supervision of the Interfederal Committee for Tracking and Tracing. This document describes the outcome of these efforts.

Version history

Version	Date	
1.0	July 15, 2020	First version
1.1	July 22, 2020	Section 2 expanded, in particular w.r.t legal dimension. Section 5.4 on integration with manual tracing revised.
1.2	August 5, 2020	Small changes to authorization protocol; corrections w.r.t. eForms. Section 5.6 on monitoring added
1.3	August 17, 2020	Small changes to authorization protocol. Added appendix on updateable parameters and on calculation of error detection.

Contents

List of Abbreviations

AEM	Associated Encrypted Metadata
AES	Advanced Encryption Standard (block cipher)
API	Application Programming Interface
BLE	Bluetooth Low Energy
CDN	Content Distribution Network
EDPB	European Data Protection Board
HMAC	A MAC algorithm based on a hash function
INSZ	Identificatienummer Sociale Zekerheid
NISS	Numéro d'Identification à la Sécurité Sociale
RPI	Rotating Proximity Identifiers (RPIs)
RSSI	Received Signal Strength Indicator
SHA-256	Cryptographic hash function
SHA-512/256	A cryptographic hash function
TEK	Temporary Exposure Key
URL	Uniform Resource Locator

1 Background and motivation: contact tracing

Experts agree that the COVID-19 epidemic can only be brought under control through a combination of testing, contact tracing and quarantine. Contact tracing identifies citizens who have been in contact with an infected person in order to quarantine and/or test these citizens. Contact tracing is an important element of a control strategy, but it is not a panacea. Only about 15% of risky contacts seem to lead to contamination, which means that - even if the contact detection itself works well - 85% of the tests after contact tracing could be negative. In addition, contact detection cannot detect infections through surfaces, through virus-laden droplets or through aerosol transmissions

Large scale manual contact tracing based on interviews has been implemented in Belgium since the beginning of May 2020. This approach is subject to a number of limitations: it requires substantial resources, is rather time consuming, which means that valuable time can be lost. In addition, it is based on the memory of the infected persons and it is not suitable for tracing contacts with strangers (e.g. on public transport or in a restaurant).

There are strong indications that a contact detection app can play a positive role in controlling the COVID-19 epidemic. An app offers significant advantages in speed and efficiency of contact tracking. Such an app should be seen as a solution that is complementary to manual contact detection: both solutions are necessary. For example, an app cannot detect contacts with citizens who do not use the app. The app also does not directly protect the user against contamination, but it can help to slow down the spread.

A large number of technological solutions for contact detection have been proposed since February 2020. These solutions differ in technology used, privacy protection and security features. There is a growing consensus in the scientific world that the approach with the highest success probability is based on the use of a smartphone app based on Bluetooth (available in 95% of smartphones in Belgium). While it is tempting to add other information (such as location data, payment information, video surveillance, and context data) in order to improve accuracy and to better understand the transmission, this would present serious privacy and surveillance risks and risk of abuse. While some countries in Asia have deployed such approaches, this seems to be unacceptable to the European citizens and not compatible with the legal requirements of the GDPR. As it would be very easy to bypass a solution based on an app, it is essential that a contact tracing app is fully trusted by the citizens. This requires that its use is voluntarily, it processes only minimal personal data and can only be used for the main goal: to warn citizens after a risk contact and to encourage them to let themselves be tested and/or quarantined.

For several reasons, it seems recommended to introduce a minimum age for the usage of the app – no decision has been made to date, but 14 seems to be a reasonable choice.

2 Requirements for a contact tracing app

1. The app offers an acceptable level of **precision**. This means that the app only identifies contacts of at least 15 minutes at a distance of 1.5 meters or less. This must be achieved with sufficient accuracy, i.e. with an optimal tradeoff between false positives (registering contacts that do not pose a risk) and false negatives (missed contacts). The accuracy of the app can be improved by offering citizens the possibility to deactivate the app for certain periods (for example, where they are protected by plexiglass).

2. Avoidance of **false or incorrect reporting** of infections: reporting an infection is only possible after a positive test result or a determination by a doctor. This means that self-reporting should not be allowed and that authorization is required to upload data.

3. A responsive (fast) and scalable test infrastructure (COVID19) is in place. The key value add of the app is that it can notify people before they develop initial symptoms, but to be able to do so, any possibly infected person should be able to get a test and the test results sufficiently fast (trigger to test result < 48hrs). Triggers can include development of symptoms, manual contract tracing, or app proximity alert.

4. **International interoperability**. The core functions of the app should work in as many countries as possible and it should be possible to share information about contamination between countries anonymously. However, health infrastructures are still organized at a national or regional level: therefore, each country will provide a customized version of the app that can communicate with its national health infrastructure.

5. **Practical requirements**. In order to deliver a practical contribution, the solution must be scalable to millions of users and available within a reasonable time frame. This means that the app has to work on the majority of current smart phones. The app must be easy to install and use with optimal features for disabled users. Finally, it should not interfere with normal use of the phone, and in particular, it should not consume too much battery.

6. The app must guarantee the **privacy** of the users, which means that the design must be based on “data minimization” and “privacy by design”, as indicated in the recommendations of the EDPB (European Data Protection Board).[1] The privacy requirements include the following (see below for further details on the data protection requirements):

- No location information may be processed (such information can only be pseudonymized or anonymized by aggregation, which is not possible when making risk calculations for individual users);
- No information is provided about who is infected by whom, where and when, and no information is collected about users who are not infected;
- As little information as possible is centrally stored which prevents abuse of data;

[1] https://edpb.europa.eu/our-work-tools/our-documents/usmernenia/guidelines-042020-use-location-data-and-contact-tracing_en

- The data will disappear automatically 14 days after the last reported infection and the system can be turned off and all stored information can be removed at the end of the epidemic;
- It is not possible to use the system or data for other purposes (e.g. sanctioning those who have not followed the quarantine rules).

These privacy requirements will be validated in a detailed Data Protection Impact Assessment.[2]

7. The use of the app should remain **voluntary** and the use (or lack of use) should certainly not give rise to discrimination or coercion. Even after a citizen receives a positive test result, he or she may decide not to use the app to notify other citizens.

8. Maximum **transparency** should be pursued in terms of both technology and the democratic safeguards taken. More specifically, this concerns the architecture of the system and the operation of the app and backend. The source code of the app and backend must be freely available (open source). In addition, the Data Protection Impact Assessment must be published. Finally, there must be transparency about the policy process behind the deployment and evaluation of the system.

9. The implementation of the app needs to be **inclusive** and should not lead to greater inequality or disadvantages.

10. The functioning of the system and its impact on individuals, communities and society must be **monitored**, evaluated and adjusted on a regular basis by an interdisciplinary and independent **oversight** body to ensure that it achieves its goals. The body should include academic and non-academic experts, representatives of vulnerable communities and representatives of civil society. If it turns out that it remains ineffective in spite of further improvements, the system must be stopped.

11. User-friendliness. Even if the user is not infected and the status of the user remains unchanged, it should be **attractive** to keep using the app. This can be achieved for example by providing daily updates on the progress of the epidemic and on the status of travel advisories for EU countries and regions.

After all, a contact tracing app can only be successful if the general public has enough confidence in the solution and understands the value of the app. The conditions stated above are essential for this.

It is very difficult to determine in advance what percentage of use is needed. A study of Oxford has been frequently misquoted as a proof that 60% adoption is needed before an app can be effective. The authors of this study have published a clarifying statement that shows that any adoption rate has positive benefits; they also indicate that 60% is the result of an unrealistic simulation in which no other measures are taken to control the epidemic. In particular, if a substantial fraction of users in a large organization or community would use the app in combination with effective manual tracing, such an app can be highly effective. A more recent study by Barrat et al. confirms the benefit of the combination of an app in combination with manual tracing.

[2] An example of the analysis for the German Corona-Warn-App can be found here https://www.coronawarn.app/assets/documents/cwa-datenschutz-folgenabschaetzung.pdf.

Limitations on the privacy of contact tracing systems

It is important to understand that contact tracing is inherently intrusive. This is obvious for manual contact tracing, where one is interviewed and asked about all personal contacts in the past days. It may seem contradictory, but with an app it is possible to be less privacy invasive and still get the health benefits.

However, it is important to understand that even for the best possible solution, contact tracing always leaks some information. Consider a case in which a user isolates herself and meets only a single person in the period of two weeks. If she is then identified as being “at risk”, she can immediately deduce that this contact must have infected her. The same conclusion holds if she meets multiple persons, but only one person has an app: if the app informs her of being at risk, she can deduce that this contact must have infected her (note that this assumes that the person knows which of her contacts has an app and which does not). This limitation is independent of the way in which the app works.

There are two main approaches to identify at risk individuals. In centralized systems, all the information collected by the apps (including information on contacts) is uploaded to a central database. The information in this database is then analyzed to identify those users who are “at risk”. In decentralized systems, only minimal information is uploaded in a central database; this information is distributed to the apps that make a local decision in the phone on the risk of the user. In centralized systems, there is an increased risk of re-identification and abuse of the information. For example, the central authority can see how many contacts each user has and can identify the social graph and the contact graph of users. In decentralized systems, such an analysis is not possible. The price paid for this is that decentralized systems are slightly more vulnerable to local attacks, in which a tech-savvy adversary records broadcasted information and at the same time observes users through cameras or in person. However, it should be pointed out that a highly motivated and technologically capable user is always able to collect highly sensitive personal information, even on users who are not using an app.

Some scientists have raised alarms about some of these local attacks on decentralized systems, without properly explaining that these attacks also apply to centralized systems and even to any contact tracing system. Overall, most experts believe that centralizing more information brings more risks, as this highly sensitive information can be abused or leaked through a data breach.

A detailed analysis of the security and privacy risks of contact tracing technologies can be found in the document:
https://github.com/DP-3T/documents/blob/master/Security%20analysis/Privacy%20and%20Security%20Attacks%20on%20Digital%20Proximity%20Tracing%20Systems.pdf.

Respect for data protection requirements

Legal basis. As any processing of personal data (which includes pseudonymised or coded data), the processing of personal data deriving from the use of the app must rely on one of the legal bases listed in the GDPR[3]. The data processing of the app is based on grounds of public interest[4], as it aims to contain the spread of COVID-19 and to protect the population against the COVID-19 epidemic. This processing for reasons of public interest must be based on a legal norm, which provides specific measures to safeguard the rights and freedoms of the data subject. One of these specific safeguards is the fact that the installation of the application is voluntary. As stated earlier, individuals who decide not to or cannot use the app should not suffer from any disadvantage at all. The use of the contact tracing app may not be imposed by employers, public transport or any other third party as a condition of access to a place, or for any other purpose.

Purpose limitation. The purpose of the app is to warn the users in case of potential exposure to the virus. It strictly excludes further processing of data for any unrelated purposes (e.g. law enforcement purposes). Designing the proximity tracing system in view of limiting the possible uses of it is of utmost importance since this is a protection against the risk of "function creep".

Data minimisation – Data quality. The app functions without direct identification of individuals. Only relevant and absolutely necessary information is collected. The app does not require tracking the location of individual users since proximity between persons can be obtained without locating them. Instead, proximity data are used.

In view of minimising the risks for data subjects, the data stored in the central database may not be cross-referenced with other databases.

Data storage and Data localisation. Personal data stored in the central database are located in Belgium. They expire automatically after 14 days; after this period they are removed.

Transparency. Besides what is said about transparency above, information must be communicated to data subjects based on Article 13 GDPR: about the controller, the DPO, the purposes of processing personal data and the legal basis, the recipients of the data, the storage period and the different rights of data subjects. Information provided to data subjects should use clear and simple plain language. Special attention should be paid to information directed to children.

Security, Integrity and confidentiality. Digital proximity tracing systems have to include state-of-the-art encryption, communications security, secure development practices and user authentication to prevent from risks such as access, modification or disclosure of the data of the proximity tracing system.

End of data processing. The Contact Tracing App will be deactivated at the end of epidemic.

[3] Art. 6.1. and Art. 9.2. General Data Protection Regulation (GDPR).

[4] Art. 6.1.(e) and Art. 9.2.(i) GDPR Art. 9.2.

3 The DP-3T Architecture in combination with the Google/Apple Exposure Notification Application Programming Interface

This section describes the DP-3T architecture (Distributed Privacy-Preserving Proximity Tracking, https://github.com/DP-3T/documents) on top of the Google/Apple Exposure Notification API (https://www.google.com/covid19/exposurenotifications/, https://www.apple.com/covid19/contacttracing).

An overview of the system is depicted in Figure 1. The details will be explained step-by-step in the next sections. Overall, the overall system consists of two main components:

- A contact tracing app for users that runs on a smartphone running Android or iOS that supports BLE (Bluetooth Low Energy) – 95% of smartphones in Belgium offer this functionality. This contact tracing apps exchange random anonymous Bluetooth beacons and get anonymous information on infected users from a central server.
- A number of backend servers to collect test results (these form the starting point of the manual contact tracing), to distribute information to apps and to interact with servers in other EU countries. The main server to collect test results is an existing server hosted by Sciensano, the national public health institute of Belgium. Sciensano is a federal scientific institution that operates under the authority of the federal minister of Public Health and the federal minister of Agriculture.

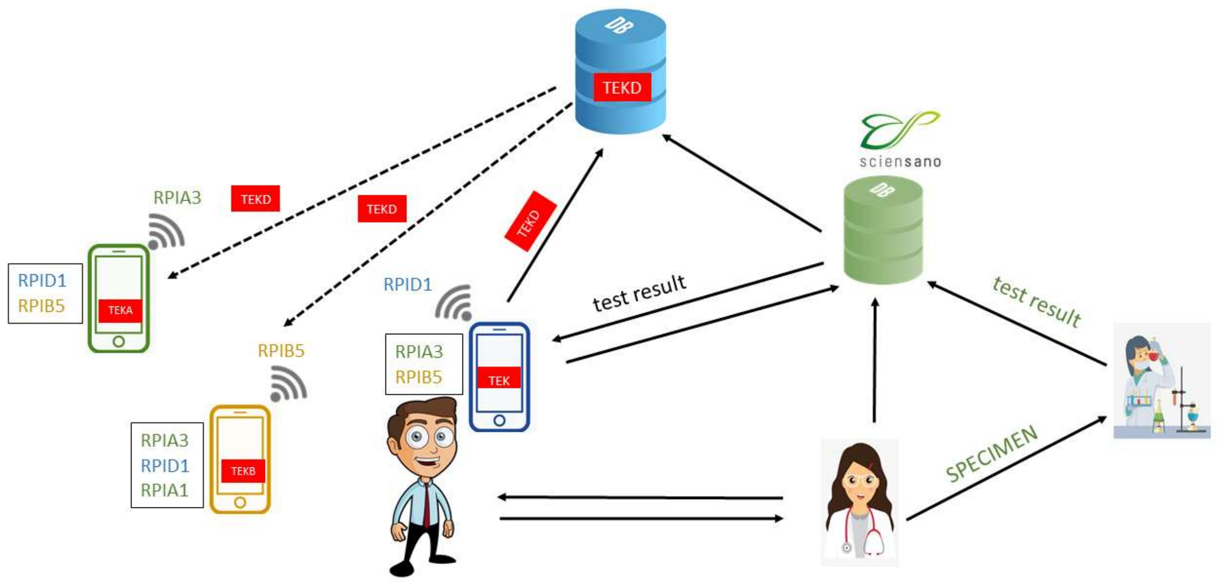

Figure 1: Overview of the contact tracing system

From the point of view of a user, the system as the following phases: installation, operation, infection (and testing), and notification, contact tracing and stopping the system. These are described below.

3.1 Installation

The citizens install the app in the Google or Apple app store. The app generates a new secret key every day. Bluetooth tokens (random numbers of 128 bits) are then generated based on these keys. For technical details on the generation of these tokens, see Section 5.1.

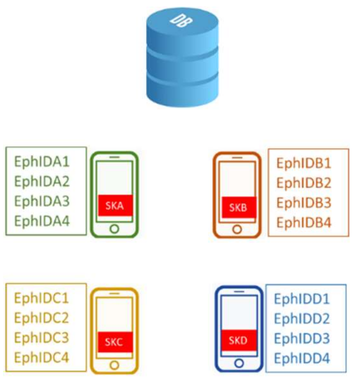

Figure 2: Installation

3.2 Operation

Each smartphone with the contact tracing app sends a Bluetooth token in broadcast mode every 0.5 second. A different token is issued every 10-20 minutes to prevent these tokens from being used to track a user; the change of token is aligned with the change of the Bluetooth MAC address. These tokens are collected by any smartphone nearby. Any smartphone with the contact tracing app also listens for Bluetooth tokens (for 4 seconds every 2.5-5 minutes). It stores the received tokens along with the day and signal strength. These tokens are kept for 14 days and then removed.

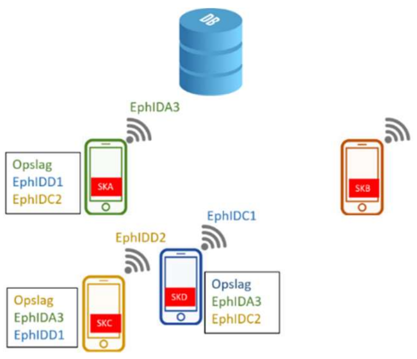

Figure 3: Operation

3.3 Testing and key upload

If a citizen has symptoms or has an infection risk, she will undergo a test for COVID-19 under supervision of a doctor (general practitioner). The citizen communicates her INSZ/NISS[5] and her mobile number to the general practitioner for manual tracing (note that this information is not needed for the app). The doctor also determines the date on which she became contagious and the citizen enters this date in the app. The app also stores the date when the sample is taken. The citizen is informed about the test result via the app. If the test result is positive, the citizen is requested to contact his/her general practitioner and to go in quarantine. The app asks the citizen consent to upload the secret keys of the infectious days with the associated data in a central database. These keys are removed from this database after 14 days. Note that the past keys can be uploaded immediately, but the key used on the current day can only be uploaded on the next day. A simplified protocol is presented in the figure below; several details that are essential for privacy are abstracted away. The full authorization protocol is described in Section 5.2 below.

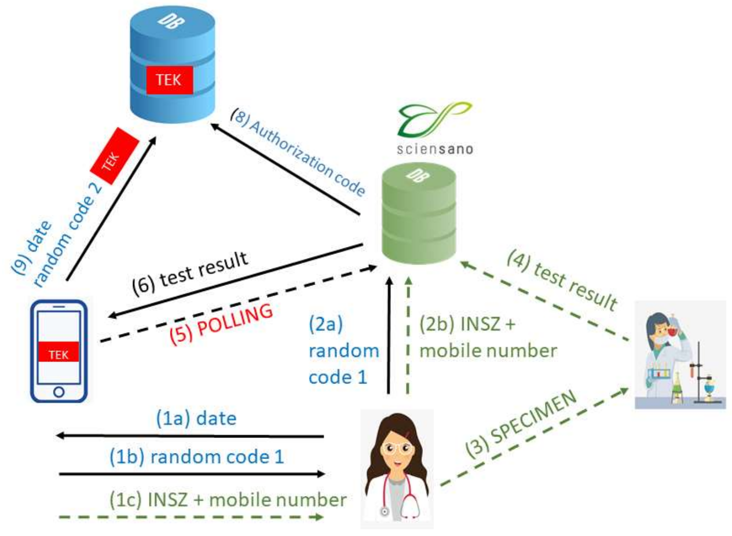

Figure 4: Testing and simplified authorization protocol

[5] Identificatienummer van de Sociale Zekerheid/Numéro d'Identification à la Sécurité Sociale.

3.4 Contact tracing

The app regularly contacts the central database and downloads the keys and associated infectious days from all infected users. This allows the app to check whether the smartphone has been sufficiently long and close to an infected person in the past 14 days. If the app detects a risk, the app informs the user about the actions to be taken (these actions will depend on the status of the epidemic; they include quarantine, contact general practitioner, contact tracing center). Note: this notification does not happen in real time, as it would violate the privacy of the infected person.

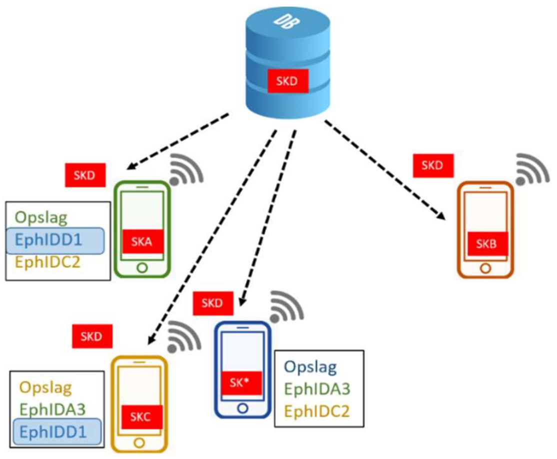

Figure 5: Contact tracing

3.5 Stopping the system

When there are no more infections, no new keys are loaded into the central database. Since keys are deleted after 14 days, this database will automatically be empty after 14 days. When the end of the epidemic is determined, the servers can be dismantled and Google and Apple will no longer support the API. The users will then be asked to uninstall the app.

4 Discussion of the solution

The DP-3T architecture on top of the Google/Apple Exposure Notification API demonstrates that it is possible to fight a pandemic without giving up essential rights to privacy. As indicated earlier, the solution presents a realistic and pragmatic approach that avoids mass surveillance at the cost of some vulnerability to local attacks by tech-savvy adversaries; however, these adversaries could also be effective in violating the user's privacy without a contact tracing app.

The solution has been deployed in several countries; in several countries such as Germany, Switzerland and Uruguay a high level of adoption[6] has been achieved and the system seems to be effective. It is clear however that the full effectivity can only be demonstrated when the solution can help in suppressing a second wave during an extended period.

Some observers have expressed concerns that the solution critically depends on Apple and Google. On the one hand, it is clear that these companies have a strong influence on our smart phones and the usage of iOS or Android requires some level of trust in these companies. On the other hand, the choice of a decentralized system means that no sensitive information needs to be collected by Google and Apple. Moreover, Google and Apple have stated that they "[…] will not receive identifying information about the user, location data, or information about any other devices the user has been in proximity of." They have also indicated that "[…] there will be no monetization from this project by Apple or Google. Consistent with well-established privacy principles, both companies are minimizing data used by the system and relying on users' devices to process information." It is important to insist that the code behind the Google/Apple Exposure Notification API is made open source or is at least audited by independent experts.

There have been some concerns on the integration of the app with the national health infrastructure. It is important to point out that a contact tracing app is complementary to manual tracing. Manual contact tracing on a large scale requires a central database with infected users and their contact information. A detailed analysis has been made how manual contact tracing and contact tracing with an app can collaborate. The design of the backend servers of the contact tracing app has attempted to separate as much as possible the infrastructure from the app from that of the manual contact tracing; while there still exists a connection, it is based on a temporary pseudonym. With proper governance and audit of the databases, the proposed solution will avoid that information from the app is connected to personal health data. In addition, the app only releases pseudonymous identifiers; linking these identifiers to a real user requires either merging databases that have been designed to be separate, or specific attacks in which one spies with a specific local device or app on user's smart phones in order to collect sensitive information that can be correlated to identity information.

[6] On 16 July 2020, the German Corona-Warn-App has more than 16 million downloads (population of Germany is 83.8 million) and the SwissCovid app has more than 1.6 million downloads (population of Switzerland is 8.6 million).

It has been decided to follow the approach of most other countries to not collect epidemiological data through the app. While this is technically feasible in the DP-3T architecture, it does increase the complexity and presents an increased privacy risk. Moreover, due to the “privacy by design” nature of the app, only limited information can be collected (e.g. no location information is available in the app). If it is necessary to analyse epidemiological properties such as the role of “superspreaders” or of specific locations, one has to rely on specific questions asked in the manual tracing process.

It is important that the app is easy and convenient to use. Flanders has sponsored research on the usability that will evaluate how the information can be best provided to the users. Several national apps support the usage of the app by people with disabilities; this functionality is also foreseen for the Belgian app.

For most of the time, most users will see in their apps the message “no infection risk”. In order to make the app more attractive, daily updates will be downloaded from the Sciensano servers on the progress of the epidemic.

Vulnerable groups in society do not use a smartphone. A contact tracing app allows slowing down the epidemic and avoiding a second lockdown; as a result, it has a positive effect on the entire population (including users without a smartphone). However, there is a risk that the positive impact of an app would be much smaller within a number of vulnerable groups, allowing the app to increase inequality and create a cumulative disadvantage for those groups. A possible solution is to develop a small portable Bluetooth device with comparable functionality - the cost of such a device with a battery life of 1 year is estimated at 5-7 EUR and the development time would be 2-3 months. It would be best to cooperate at European level for the development of such a device.

So far only a high-level technical description has been given. However, the devil is in the details: even if the high-level design is fine, any mistake or sloppiness in the detailed design can result in privacy and security vulnerabilities. Section 5 presents the full technical details. Moreover, it is not sufficient to have a high quality specification: one also needs to carefully evaluate the implementation and monitor the usage of the app. It is essential that sufficient attention be paid to these aspects.

5 Detailed technical specification

This section provides the technical details of the application and the backend. Section 5.1 contains the cryptographic specification for the Bluetooth tokens; this section is based on the Google/Apple Exposure Notification API. Section 5.2 describes the authorization protocol that serves to authorize key uploads. The risk exposure computation is explained in Section 5.3. Section 5.4 discusses the interaction with manual contact tracing. The international interoperability is presented in Section 5.5.

5.1 Cryptographic specification for Bluetooth tokens

This section describes the cryptographic functionality implemented by the Google/Apple Exposure Notification API (Application Programming Interface).

Every day the app generates a Temporary Exposure Key (TEK). This is a 128-bit random string that is associated with an index i. We first define the ENINIntervalNumber, a 32-bit (uint32_t) unsigned little-endian value that is the number Unix Epoch Time measured in 10-minute time slots.

ENIntervalNumber(Timestamp) $\leftarrow$ Timestamp/ (60×10) with Timestamp is Unix Epoch Time in seconds

Then the index i is computed as follows (with TEKRollingPeriod = 144 which corresponds to 24 hours):

$i \leftarrow \lfloor$ ENIntervalNumber(Time at Key Generation)/ TEKRollingPeriod $\rfloor \times$ TEKRollingPeriod .

From each TEK_i key two intermediary 128-bit keys are derived: the Rolling Proximity Identifier (RPKI) key RPK_i and the Associated Encrypted Metadata (AEM) key AEM_i. The key derivation function is based on the hash function SHA-2:

Output $\leftarrow$ SHA-2(Key, Salt, Info, OutputLength), with Key=TEK_i, Salt = NULL and OutputLength = 16 .

For RPKI Info = UTF8("EN-RPKI") and for AEM Info = UTF8("EN-AEMK").

From the key RPK_i, for each time slot j 128-bit a Rolling Proximity Identifier $\text{RPI}_{i,j}$ is generated using AES-128 in ECB (Electronic Code Book) mode. The 16-byte (128-bit) plaintext is defined as follows:

PaddedData[0...5] = UTF8("EN-RPI")

PaddedData[6...11] = 0x000000000000

PaddedData[12...15] = ENIN_j

Here $\text{ENIN}_j \leftarrow$ ENIntervalNumber(j).

The app encrypt relevant Metadata (the type of phone and the transmission level that are used for calibration) in AES-128 CTR (Counter) mode with as IV the value of the Rolling Proximity Identifier $\text{RPI}_{i,j}$; this results in a 128-bit ciphertext $C_{i,j}$.

Each phone will broadcast 256 bits consisting of the Rolling Proximity Identifier Identifier $RPI_{i,j}$ and the 128-bit ciphertext $C_{i,j}$. Note that $C_{i,j}$ cannot be decrypted with the knowledge of $RPI_{i,j}$ only or even with that of RPK_i: one needs to have TEK_i to compute AEM_i.

For more details on the cryptography of the Exposure Notification API, see the specifications of Apple and Google:
https://blog.google/documents/69/Exposure_Notification_-_Cryptography_Specification_v1.2.1.pdf.

The detailed Bluetooth specifications are described in:
https://blog.google/documents/70/Exposure_Notification_-_Bluetooth_Specification_v1.2.2.pdf.

5.2 Authorization protocol

Objective

The goal of this protocol is to authorize an infected user (the index case) to upload her Temporary Exposure Keys (TEKs, cf. Section 5.1) from the days she was infectious in the central database as well as the countries she has visited on those days. The days on which she was infectious have to be decided in concertation with a medical professional based on several factors including the date of the first symptoms (if any), the date of the test and the date on which the user was informed of a potential risk. This decision should be aligned with the same decision for manual contact tracing as specified by Sciensano.

A related authorization protocol together with its analysis is protocol 3 described in the following DP-3T document:
https://github.com/DP-3T/documents/blob/master/DP3T%20-%20Upload%20Authorisation%20Analysis%20and%20Guidelines.pdf

Actors

- User with smartphone and app: install app, request test, receive test result, and receive notification of possible risk.
- Doctor: take samples and request test for the user (samples can also be taken in a hospital or triage center).
- Reference lab: perform tests and report results.
- Sciensano: manage three databases and connects to the EU federation gateway for international interoperability (cf. Section 5.5).
    - Database for epidemiological data and manual contact tracing with INSZ/NISS number, mobile number of the user, date of test, test result, date on which the user has become contagious, random code generated by the app (Database I in KB 44).
    - Central database (DB1) with keys of infected users, date of each key, country/countries where key was used: receives keys and date from app and distributes this information to all apps (Database~V in KB 44).
    - Pseudonymous polling database (DB2) for app with test result, date when user became contagious, random code generated by the app. This server also generates authorization codes.

Functional requirements:

- When a user applies for a test, the app registers the date of the test and the date on which the user may have become infected. The app is linked to the test result via a pseudonym.
- Minimum overhead for doctor: the doctor requests a test by prescription with INSZ/NISS number and date. He/she registers at that time in the Sciensano medical database the following data:

INSZ/NISS number, user's mobile number, date of test, date on which the user became contagious (based on symptoms), random code generated by the app. Note that the app only requires the last three data items; the other data are used for manual contact tracing.

- Test result goes from the lab to the database for manual contact tracing at Sciensano.
- Sciensano transfers the test result to the pseudonymous database.
- After a test, the app will regularly poll the pseudonymous database based on the random code of the app and the date on which the user became infectious to inquire about the test result.
- A user who receives a positive test result is asked whether she wants to upload her keys and whether she wants to indicate which countries have been visited. In addition, the user is asked to self-quarantine, contact her doctor and perhaps call the tracing center.
- Only keys with authorization code provided by the pseudonymous database can be uploaded in the central database.

Security requirements

- Only users with a positive test can upload information (keys, date, and countries) to the central database.
- There is a check on the integrity of the data (keys, data) that is loaded in the database: only correct information can be uploaded from the app of the person who tests positive (period cannot be adjusted, or the key of another app cannot be uploaded).
- Authentication of pseudonymous polling database and central server apps.

Privacy requirements

- Information in the app cannot be linked to any identity or telephone number.
- No one can link the names of the users of the app to the keys in the central database (this assumes that the different databases managed by Sciensano are separated, that is, the information between these databases is not correlated).
- The communication between computer of the doctor and Sciensano and between all the databases is properly protected.
- It is not possible to deduce from a contact between the app and the polling database or an app and the central database that a user of an app is infected.
- The central database and polling database do not learn the IP addresses of the apps.

Additional assumptions:

a. A user is contagious for up to 14 days (some sources report 21 days - this number should be a parameter).
b. The app can regularly contact the polling database to check whether a test result is available.
c. The app knows the web address (URL) and the public keys of the polling database and the central database. The integrity of these public keys is guaranteed.
d. The central database has the public key of the polling database. The integrity of this public key is guaranteed.

e. The polling database and the central database are protected against DDoS (Distributed Denial of Service) attacks.

Authorization Protocol: polling

An overview of the authorization protocol is presented in Figure 6 below. The step-by-step details are explained in this section.

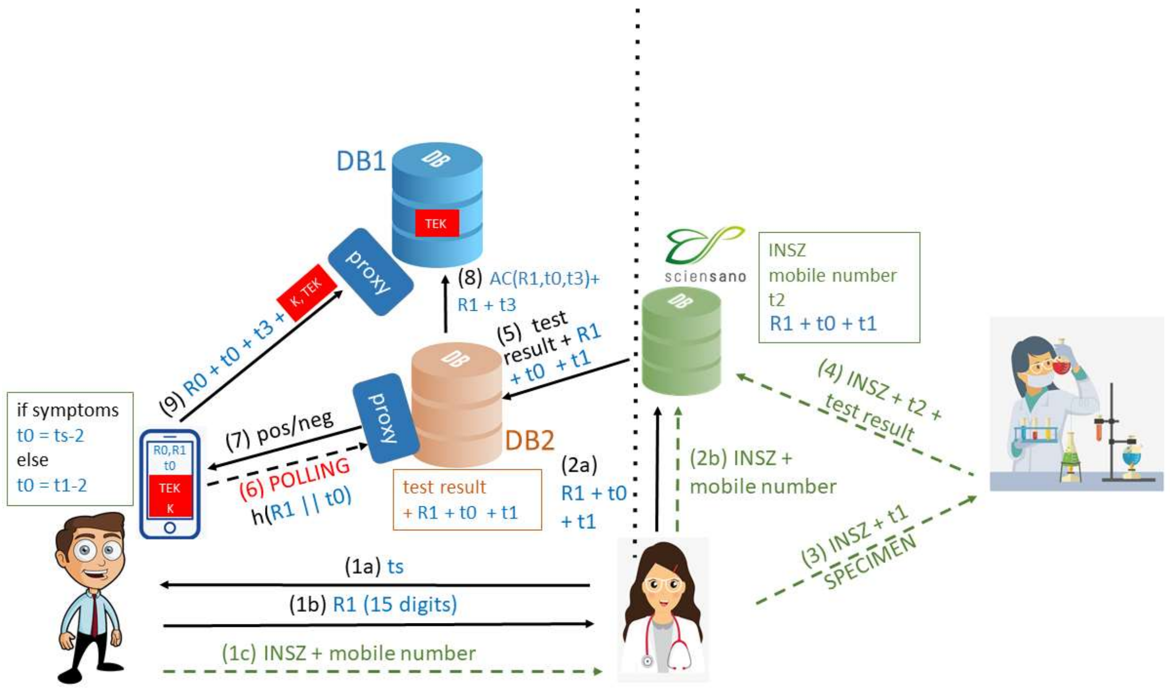
DB1
TEK
proxy
(8) AC(R1,t0,t3)+ R1 + t3
(9) R0 + t0 + t3 + K, TEK
(5) test result + R1 + t0 + t1
proxy
DB2
(7) pos/neg
(6) POLLING h(R1 || t0)
test result + R1 + t0 + t1
(2a) R1 + t0 + t1
if symptoms t0 = ts-2 else t0 = t1-2
R0, R1 t0
TEK K
sciensano
INSZ mobile number t2 R1 + t0 + t1
(4) INSZ + t2 + test result
(2b) INSZ + mobile number
(3) INSZ + t1 SPECIMEN
(1a) ts
(1b) R1 (15 digits)
(1c) INSZ + mobile number

Figure 6: Authorization protocol. The random code is computed as R1 = H(K || R0 || t0), where R0 is a 128-bit random string and K is a 128-bit secret key. The authorization code AC is a digital signature on R1 || t0 || t3. The dates are defined as follows: ts is the date of onset of symptoms, t0 is the date on which user becomes infectious, t1 is the date the sample is taken, t2 is the date of the test, and t3 is the date on which the test result was communicated.

1) (Step 1a in Figure 6). A user - with the contact tracing app on her phone – has indications that she may be infected so she visits a doctor (or a triage center). The doctor determines (based on a conversation with the user and on the Sciensano guidelines) the date ts of onset of symptoms if any and communicates this to the user. The doctor enters the date of onset of symptoms or the absence of symptoms in his computer.

2) (Step 1b and 1c in Figure 6). The user clicks the "**COVID-19 test**" button in the app and enters the date of onset of symptoms or checks a box to indicate that she has no symptoms. Next the app computes an estimate for t0, the date when the user became contagious (in the case of a positive test). If the

user has symptoms, t0 is computed as t0=ts-2, otherwise t0= t1-2 with t1 the current date. Note that in both cases, the "-2" depends on an epidemiological decision, so it should be possible to update this as a parameter of the app. The app generates a key K (a 128-bit AES key) and a code R0 (128-bit string) and calculates a 15-digit code R1 (random code generated by the app). The code R1 is derived from a 128-bit cryptographic hash of the secret key K, a 128-bit random string R0, the date t0 and the string "test request".

Output ← SHA-2(Key, Salt, Info, OutputLength), with Key=K, Salt = NULL, and OutputLength =7 .

Here Info = R0 || t0 || UTF8("TEST REQUEST").

Note that the code R1 is computed from Output by converting the 56 output bits to a 15-digit integer. This can be achieved with large integer arithmetic; an easier way to achieve the same goal with a slightly higher but acceptable bias: first, one constructs an integer from the first 20 bits and reduces this integer modulo 10^6; similar for the next 20 bits; the next 10 bits are reduced modulo 10^3; the last 6 bits are ignored). **If a value of R1 equal to all 0s is obtained, a new value of R0 has to be selected** (this is expected to happen only very rarely).

This code R1 is shown on the screen (as a number or as a QR code) of the smart phone; the doctor registers this code (manually or with an app on his/her phone). The app stores K, R0, R1, t0 and t1, the date on which the button is clicked. Note that 15 digits are required since at peak moments one may expect that 100,000 tests per day are performed, 50% of these for a person with an app; in that case the probability of colliding values of R0 on a single day is upper bounded by 1 in 800,000. Since R1 (15 digits) and t0 (6 digits yymmdd) may be entered manually, 2 check digits are added to make sure that the resulting 23-digit number is a multiple of 97. This allows detecting errors when the number is typed in. The details of the calculation are explained in Annex A.

3) (Step 2a, 2b and 3 in Figure 6). The doctor requests the INSZ/NISS number and mobile number of the user (the mobile number is only needed for manual contact tracing; it is not used in the app). The doctor sends the following data to the database Sciensano for manual testing and contact tracing: INSZ/NISS number, mobile number of the user, t1 (date of sample), t0 (computed from ts or t1), R1. This information is sent over a secure connection in an eForm ("Melding en labo-aanvraag bij vermoeden van besmetting SARS-CoV-2"). The Sciensano database for epidemiological data and manual contact tracing stores these values pending the test results. The doctor sends the sample to the reference lab together with the INSZ/NISS number and the date t1 of the sample.

Note that the doctor can also declare a COVID-19 infection without waiting for a test result. In that case the eForm "Directe aanvraag contactopvolging bij zeer sterk vermoeden van besmetting COVID-19, onafhankelijk van het testresultaat" is used. In that case the variable "declared by doctor" is set to 1 (d=1), action 5) below will be taken immediately; negative test results will be ignored in that case. This is described in the following Sciensano document:

https://covid-19.sciensano.be/sites/default/files/Covid19/COVID-19_procedure_GP_NL.pdf
https://covid-19.sciensano.be/sites/default/files/Covid19/COVID-19_procedure_GP_FR.pdf

Note: if a user forgot her phone, the data exchange under 3) could in principle be performed afterwards remotely. However, this brings some risks, as it requires that the client is trusted to use her phone (and not another phone) and it assumes that the doctor can verify her identity remotely.

4) (Step 4 in Figure 6). The reference lab performs the PCR test and sends the result to Sciensano together with INSZ/NISS number, the date of the sample t1 and the date of the test t2.

5) (Step 5 in Figure 6). The Sciensano database for testing and manual contact detection sends the following values to the pseudonymous polling database: R1, t0, t1, test result, declared by doctor (d=0 or d=1). **It deletes R1 from its database.** At this step, the manual tracing procedure starts at Sciensano; the description of this process is outside the scope of this document. Note that citizens with an app may have had contacts with citizens without an app, hence manual contact tracing is still necessary. For more details, see Section 5.4. Note that in exceptional circumstances, R1, t0 and t1 arrive after the test in the Sciensano database; in that case, the test result is stored up to 5 days until these values arrive.

6) (Step 6 and 7 in Figure 6). The user's app contacts the Sciensano polling database every 1-2 hours with the values SHA-256(R1 || t0). There are three possible answers:

a) No test result yet.

b) Negative test and d=0: the app communicates this to the user. Twenty-four (24) hours after the test result is shown to the user, the test result, R1, R0, t1, t0 and d are removed from the app.

c) Positive test or[7] d=1: the app communicates this to the user and provides the user with health advice: the user is advised to isolate herself, contact her general practitioner and perhaps call a central number (manual contact tracing). The app asks the user to give explicit permission to upload the keys. If the user refuses the permission, no upload takes place. The user can change her mind and still perform an upload in the next 24 hours[8]; after that delay, the test result, R1, R0, t1, t0 and d are removed from the app. This means that a key upload will only be possible after a new positive test. If the user agrees, the steps below are followed.

Once the test result corresponding to R1 and t0 has been downloaded, the values R1, t0, t1 and the test result shall be deleted from the polling database. These values shall be removed on the date t0+14 (where 14 is the maximum number of days a patient is infectious; this is also an epidemiological parameter that can be adjusted later). If the app has not received a test result after t0+14 days, it should delete the values K, R1, R0, t1, t0.

Note that it is assumed that a user can have only one active test; if a user wants to take a new test, she first has to delete the values K, R1, R0, t1, t0 corresponding to the previous test and only then can

[7] Inclusive or: to avoid all misunderstanding: if d=1 the test result is ignored; if d=0, the test result is kept as is.

[8] Note that the indication of a positive status of an app user inside the app brings the risk that the app can be used as immunity/infection passport, which is highly undesirable; 24 hour after showing the test result seems a reasonable compromise.

she click the "**COVID-19 test"** button in the app. Therefore, the result of the previous test can no longer be retrieved with the app; however, it could still be obtained by the doctor.

7) (Step 8 and 9 in Figure 6) (positive test result and user allows to upload keys) The polling server computes a digital signature on the following data: R1, t0, t3 (t3 is the date of the communication of the result), d. This digital signature is an authorization code that is sent to the central server (batch processing, once per hour – this is important to protect the privacy of the user) together withR1 and t3. Next, the app asks the user if she has been in other countries during the period t0-t3 (per day). The app will requests the TEK keys from the Google/Apple Exposure Notification API for the period t0-t3 and sends them to the central server together with the dates t0, t3, the countries for each key as well as the key K, R0 and the value d. Note that for privacy reasons the app will never send the key of the current day: it will send the keys of the period until the day before t3. From version 1.5 of the Google/Apple Exposure Notification Interface (cf. Section 5.1), a new key will be generated at the time of the upload and the key of the current day can also be sent. After uploading the keys, all the data in the app should be reset and the user should be recommended to remove the app (since the app should not be used to verify whether the user complies with isolation measures).

8) The central database processes the received keys every 1-2 hours. The server recalculates R1 using K, R0 and t0 and checks if in the list of recent (last 48h) authorization codes an authorization code for R1 is present. An authorization code is a digital signature on R1, t0 and t3 (t3 can be used to check that the code is recent). If there is a valid authorization code, then the TEK keys of the period t0-t3 together with data and relevant countries are included in the central database. Note that an authorization code can be used only once.

Note 1. It is foreseen that users without symptoms but with an elevated infection risk (e.g. returning from a "red zone" abroad, notification of being at risk via manual tracing) will get a special code so that they can get tested without visiting a doctor. For this purpose, they will fill in a web form with this special code and their personal details. If these users have an app, they will press the "**COVID-19 test"** button in the app while filling in the web form; as they have no symptoms, they will indicate that in the app and t0 will be computed as follows: t0=t1-2. Then R1 will be generated by the app and entered in the web form. This information will directly go to the testing center where it will be forwarded in an eForm to the Sciensano database. Note that the actual date of the test can be a few days later than t1 used in the app; if that would be necessary, the values of t0 and t1 could be corrected in the test center, that knows the exact date the sample was taken. This offset could be communicated to DB2 and could be taken into account when selecting the keys that are included in the database.

Note 2: Doctors can decide interpret a negative test result as a positive test result. In that case the doctor will call the patient and ask her to remove the old test values and press the "**COVID-19 test"** button in the app; the values of ts/t1 will be used to determine t0 in the app and the app will generate a new value of R1 that can be communicated over the phone. Subsequently the doctor will enter the values R1/t0/t1 in an eForm "Request contact tracing in case of negative test result". This form will result in a positive test result being communicated to the polling database and the steps 6) through 9) above can be followed.

Implementation details

All the test results are sent to Sciensano, hence the health status of infected citizens is known to Sciensano. Moreover, Sciensano needs a database with the contact information of all patients in order to perform manual tracing; this database contains no specific information about the app except the fact that a user has an app (to avoid that users who have an app are contacted for manual tracing when that is not necessary).

However, the authorization protocol is designed in such a way that the additional information provided by the app (e.g. the IP address) and the link between the test result and the uploaded keys are hidden from Sciensano. The only link between the app and the test result is the random identifier R1: this identifier is a short-term pseudonym used by the app that is stored in the Sciensano only during the time the lab is testing the sample. Organizational and logical measures need to be deployed to ensure that the value R1 is indeed erased and that the content of the three databases is not correlated.

Polling database: test results

The apps contact this database via a proxy to hide their IP addresses.

The traffic to this database does not require dummy traffic because most tests (95%+) are negative hence it is not necessary to hide that a user has undergone a test. However, apps that perform dummy key uploads to the central database (cf. infra) will first perform a number of dummy requests to make this dummy upload look like other uploads. These requests will have R1 equal to all 0.

The interaction with the polling database should not leak test result: the result shall be encrypted and regardless of the response (result ready or not, value of the test result) and the value of R1, the message timing and length should be the same.

Central database: uploading and storing keys

The apps contact this database via a proxy to hide their IP addresses.

In order to hide the infection status of a user, dummy traffic is required. Every user will perform a dummy upload to the central server every five days; more precisely, dummy uploads are performed with a Poisson distribution with expected value λ=0.2. Dummy requests contain random keys. As they do not have a corresponding authorization code, they will not be accepted. Before performing the dummy upload, the app has to send a few polls (typically 3-4) to the polling database to inquire about test results. Note that this should be performed with a value of R1 equal to all 0 so that the polling database knows that this is a dummy request.

A network level attacker could observe the incoming communications to the central database and could download the list of keys. This may leak information on how many of those communications corresponded to the keys of infected users. It is thus recommended that if there are very few key uploads, the central database adds each day a limited number of dummy keys so that the total keys per day is at least 10 (this has been implemented in the German Corona Warn app). These keys will be downloaded to all apps, but will never lead to an exposure risk.

DDoS attacks

Any system exposed to the internet risks to be flooded with malicious requests. A simple solution is to filter these requests in the proxies. Indeed, the apps will not contact any of the databases more than once per 2 hours, hence IP addresses that send too many requests can be blocked.

Note that if that would not be sufficient, DDoS attacks could be stopped with a simple proof of work. The polling database or the central database first sends a random number x of 128 bits to the app and waits for a 128-bit number y so that SHA-512/256 (date || x || y) ends with a number of 0 bits (e.g. 20 or 24). This defense may not be need initially but could be activated if DDoS attacks turn out to be a problem.

Cryptography

The following cryptographic algorithms shall be used for the authorization protocol:

- The calculation of R1 from R0 shall use SHA-256 as specified above; an alternative solution is to use HMAC-SHA-256.
- For the digital signature for authorization, the recommended algorithm is EdDSA with the curve edwards25519 (RFC 8032); an alternative solution is curve41417 (Goldilocks). If none of these are possible, ECDSA with SHA-256 is acceptable with a standardized elliptic curve of at least 256 bits (e.g. NIST P-256).
- The hash function for the small proof of work shall be SHA512/256 (truncated version of SHA-256 - intentionally different function than Bitcoin).
- TLS 1.3 and certificate pinning of the proxy servers shall be used whenever possible; if TLS 1.3 is not available, TLS 1.2 shall be used.

5.3 Calculation of the exposure risk

Before the exposure risk can be calculated, all apps need to download the TEK keys from infected users. Subsequently these keys are used to compute the exposure risk in the app.

Distribution of keys

If a user is infected, she can give consent to upload her secret TEK keys together with the days and visited countries into the central database as described in Section 5.2. At least once per day, all the apps download these TEKS keys and perform a calculation to evaluate whether a user is at risk. The file with this data is digitally signed by the central database DB1; this means that the operator of the server has to create a public/private key pair and to include the public key pair in the app. The detailed format for the key downloads is specified in the following document:

https://developers.google.com/android/exposure-notifications/exposure-key-file-format

In several countries it has been decided to only consider keys from the past 10 days; this seems a sensible decision to follow (it can be a parameter in the app). Note that the Google/Apple Exposure Notification API allows going back up to 14 days.

The number of TEK keys per infected users is thus at most 14, where each key requires 16 bytes, resulting in a maximum size of 224 bytes per infected user. If there are a few thousand infections per day (assume a second wave in Belgium) the total size of the daily download could easily be 1 Mbyte. This information has to be sent to 1 million phones, which presents a substantial network load. For this purpose, a Content Distribution Network (CDN) will be used. Note that the information distributed by this network is public information that has to be sent to all apps: it consists of all the keys of infected users and the days on which these keys have been used. This information does not contain identities or location. Internet players use a similar distribution mechanism to distribute videos or software to users.

Calculation of the exposure risk

The next step is to determine the exposure risk. Epidemiological research indicates that the risk of exposure is related to distance and to duration of a risk (there are other factors such as wearing a mouth mask, indoor versus outdoor contact, activity during contact; however, these are not taken into account in the app). There exist protocols to measure the distance using BLE with a resolution of 30 cm: however, they require changes to the radios and a pairwise interaction between two phones. The first is clearly not feasible; adding an interaction would make the protocol unsuitable for crowded places and would increase battery consumption. Therefore, the decision has been made to approximate the distance by the attenuation value. The attenuation value (expressed in dB) is calculated as the difference between the transmission power of the sending smartphone and the power of the signal registered at a receiving device (RSSI). Note that extensive measurements are necessary to calibrate these values between different phone models.

This is an acceptable approximation to compute exposure risk because contact tracing itself is inherently noisy:

- Distance itself gives only an approximate indication of the transmission risk. While some countries use guidelines of 1.5m, others use 6 feet (1.83m) and yet others 2m; clearly, it is not the case that there is a strict threshold. Moreover, manual tracing efforts are not based on exact distances either.
- In manual contact tracing, attack rates (percentage of people identified through contact tracing who end up having been infected), tend to be rather low: typical values are 15% or less.

Figure 7 shows an example of a measurement performed by the DP-3T consortium. These measurements confirm that attenuation value is not a good indicator for distance (this was well understood from the literature). However, if one considers the relevant range below 3m, there is a clear correlation between distance and attenuation. Hence, we can analyze which attenuation thresholds both indicate that the devices are within a given distance and capture the majority of the devices that are within that distance.

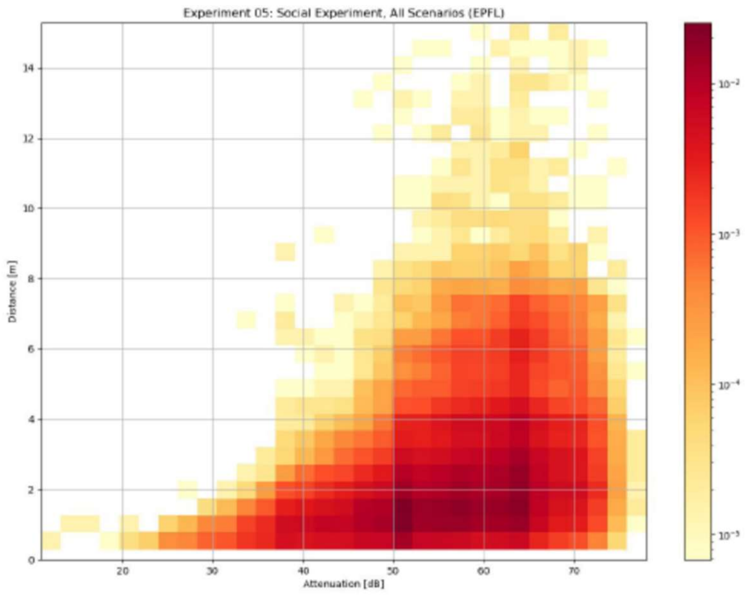

Figure 7: Heat map displaying the signal attenuation and physical distance between two Smart phones. Each square represents the number of signals received at an attenuation-distance pair (red is more frequent). Source: DP-3T Consortium

The first aspect is covered by the concept of precision: **precision** is the fraction of beacons for which an attenuation threshold correctly identifies that the phone is within a given distance. The second aspect corresponds to recall: **recall** denotes the fraction of beacons from phones within that distance that have attenuation equal or smaller than the threshold.

An extensive set of measurements performed by the SwissCovid team has led to the results described in Figure 8 that shows the tradeoff between precision and recall. We cite here from the DP-3T report:

> *For example, on the blue curve (1.5m), around 55% of the beacons with an attenuation level of 55 dB or less came from the phones that were within 1.5m, and 60% of the beacons from phones that were physically within 1.5m were attenuated by 55dB or less (i.e., precision≈0.55, recall≈0.6). For the same 55 dB threshold, the precision at 2m and 3m increases to around 0.75 and 0.9, respectively (highlighted by the red dotted line). This means that most of the beacons that were received with attenuation below 55 dB and that did not originate from within 1.5m, came from devices that were between 1.5m and 3m away.*

This leads to the following conclusion: if one aims to collect information from a distance of about 1.5m, but one accepts false positives up to 3m, a very high recall can be obtained.

The Google/Apple Exposure Notification API does not give the developer of the app or the user of the phone access to the received beacons nor to the received RSSI or attenuation. On the one hand, this limits the options for a developer to evaluate the exposure risk. On the other hand, it strengthens the privacy protection offered by the implementation, as this sensitive information is not available outside the API.

The Google/Apple ExposureSummary API takes as input a set of TEK keys corresponding to users with a positive test and two thresholds, tr1 and tr2 (tr1 < tr2), that partition the range of attenuation values into three buckets:

- 0-tr1: bucket 1: high exposure risk;
- tr1-tr2: bucket 2: moderate exposure risk;
- tr2-…: bucket 3: very low exposure risk,

The third bucket will be ignored in the calculation of the exposure risk.

The API returns only aggregate information that is relevant to compute the risk: for each of the three buckets, namely the value "duration_at_attenuation": this is the joint duration of exposure to all users with positives tests whose keys were in the input set. Note that this means that it is not possible to relate an exposure to a single individual; moreover, a single individual can have contributed to risk in multiple buckets. Each of these values is limited to a maximum of 30 minutes.

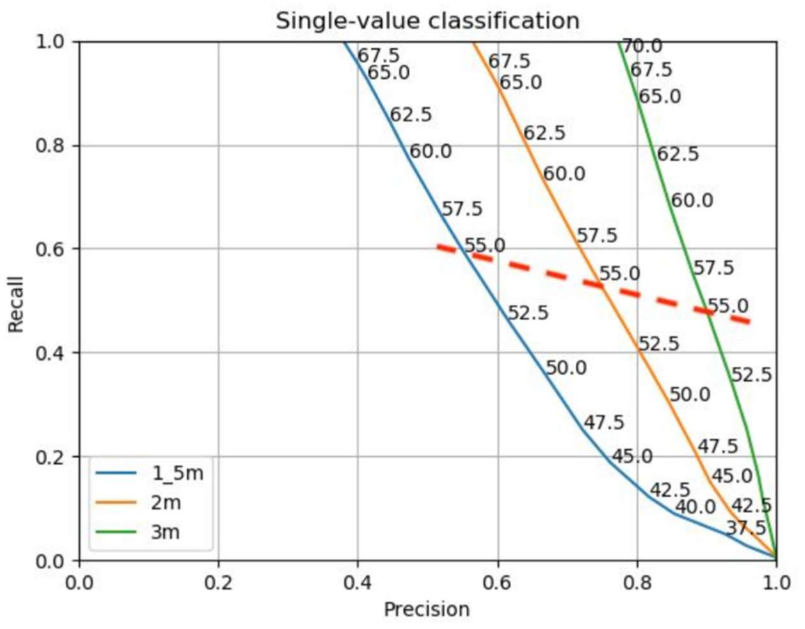

Figure 8: Precision and recall from experiments at three distances. The numbers along the curves are the attenuation values in dB. Source: DP-3T Consortium

The Exposure Score (ES) is estimated by taking a weighted sum of the content of the three buckets:

ES = w1*B1 + w2*B2 + w3*B3 .

Here B1, B2, and B3 are the durations of exposure in attenuation bucket1, bucket2, and bucket3 and w1, w2, and w3 are the weights associated with each bucket. For the Belgian app, the following values are proposed:

tr1=55 dB, tr2=63 dB, w1=1, w2=0.5, w3=0 .

The threshold values are identical to those currently used by the German Corona Warn app. The weight w2 of 0.5 means that midrange contacts have less weight in determining the exposure risk.

Risks are computed per calendar day; on any day, a user is determined to be at risk if ES ≥ 15 minutes over this calendar day. If that is not feasible, they can be computed in an aggregated way over the last 14 days as is performed in the German Corona Warn app.

Note that the values of tr1, tr2, w1, w2, and w3 are parameter values of the app; it shall be possible to update these if new epidemiological insights would justify doing so. Moreover, it is also conceivable that a complete new calculation method is introduced later on by providing an update of the app.

More details on the Swiss risk calculation can be found in https://github.com/admin-ch/PT-System-Documents/blob/master/SwissCovid-ExposureScore.pdf.

5.4 Integration with manual tracing

As explained in Section 1, we consider the app and manual tracing as complementary components that are both part of a full solution to fight COVID-19. This also implies that we need to understand how the two work together. We explain this in more detail from a functional and from a technical perspective.

Functional overview of the integration

We consider several scenarios of how manual and app-based tracing integrates.

5.4.1.1 Persona A with app infects persona B with app

When a COVID-19-confirmed persona A with an app infects a persona B with an app, the normal flow is that persona B is informed via the app that he/she is at risk. As a result, persona B can start self-quarantine and initiate a test. Even in this normal case, manual tracing will be started to understand from persona A if other potential contacts (with or without app) are at risk. Non-COVID-19 confirmed personas at risk with an app that initiate a test timely[9] should not be contacted by manual tracing until a positive test result is registered.

The main exception to this normal flow is the following: Persona A can choose not to disclose his positive status via the app. As a result, no keys will be added to the Central Database and no other app users can benefit to identify the potential risk. While we could design a system to monitor unused authorization keys, and start tracking these users, we prefer not to do so for reasons of privacy, as this would require storing information that can link the user for a longer period of time.

A detailed schematic overview of this scenario is available at
https://www.websequencediagrams.com/files/render?link=sYERflIGyNmMC10CKKEPtE3IhZTrCXxpPIOYd8oLVM498GxJSULwbXLUYb8NUtrj

5.4.1.2 Persona A with app infects persona B without app

When a COVID-19-confirmed persona A with an app infects a persona B without an app, the app cannot inform persona B of the potential infection. We see two options of how persona B can be notified:

- Through manual tracing, we might learn that persona B is at risk and manual tracing will then contact persona B (and provide him with a 12-digit code to initiate a test);
- Persona A can directly contact persona B with the message that the latter might be at risk. Persona B can then initiate a test (without a 12-digit code).

[9] Exact time bound needs to be determined.

A detailed schematic overview of this scenario is available at
https://www.websequencediagrams.com/files/render?link=EDhjuTjbIGZsJrHB47V9H0Y0xuFckvGV1dWGxuBKFMDxWkyb0OcKuOtZmYK64Cph

5.4.1.3 Persona A without app infects persona B with app

When a COVID-19-confirmed persona A without an app infects a persona B with an app, the only way persona B can be informed is either through manual tracing or through direct update from persona A. In the former case, a 12-digit code may be provided to A to initiate the test, while in the latter case no code will be used to initiate the test.

A detailed schematic overview of this scenario is available at
https://www.websequencediagrams.com/files/render?link=r5CY1ir2r8ph9sV5ewDPJGpFWPFJ6zg964LSTCyPWNn3jNhDPHbN9j5hZmQBwXg0

Technical implications of the integration

To support the scenarios explained above, it is important to ensure the current systems at Sciensano support the following:

- The Sciensano Database for epidemiological data and manual contact tracing contains a 'motivation' field that registers the motivation of the test and can have these options: manual contact tracing person at risk, app proximity tracing person at risk, symptoms. The "app proximity tracing person at risk" indication can help to monitor the effectivity of the app.
- For persons at risk who have been informed via the app and who initiated the test (timely), manual tracing should not occur (at least not before COVID-19 is confirmed). The Sciensano Database can already keep track of this, by flagging which at risk persons are currently being tested and by excluding these people from contact tracing (test in progress). If the test is positive, they should be contacted as described in Section 5.4.1.1; if the test is negative, there is no need to further contact them.
- The current tracing systems shall be able to support the information required for the app, in particular R1. Note that t0 and t2 are also required in the app scheme, but they are already used in the current system. All other information, such as INSZ and mobile phone number, are only necessary for the manual tracing system, and are not required for a correction functioning of the app.
- The current manual tracing system uses a 12-digit code to initiate a test. While we have considered to use and/or extension of this 12-digit code within the context of the app, we consider the code not useful for the correct functioning of the app given all the previously described flows and actions.

5.5 International interoperability

On August 12, 2020, contact tracing apps based on DP-3T have been deployed in Austria, Croatia, Czech Republic, Denmark, Germany, Ireland, Italy, Latvia, Poland, Switzerland, and Uruguay. For the next months, deployments are planned for Belgium, Cyprus, Ecuador, Estonia, Finland, Hungary, Japan, Lithuania, Malta, The Netherlands, Portugal, Saudi Arabia, Spain, and UK. From the beginning, the DP-3T consortium has considered an interoperability approach when developing the protocol. The main advantage of the DP-3T protocol is that only TEK keys have to be exchanged between countries: these keys are random values that give no indication on location or contacts. Sharing his information is much less sensitive than sharing contact information for manual tracing or contact information that is shared in centralized systems.

While these apps are national apps, several of them can be downloaded outside their countries of origin. This means that foreign users traveling in this country would be able to get “at risk” notifications. If users would be tested positive in the country of the app, they may be able to upload their keys to the server of that country, depending on the national health infrastructure and authorization scheme of the country of the app. However, if they test positive in their home country, it is unlikely that they can upload the keys of the app of another country. In particular, for the EU, where people change countries frequently, this approach seems rather inconvenient: users would be forced to manage a substantial number of apps and received information on infections or risk would be limited to a single country. Moreover, Google and Apple allow only a single app to use the API at the same time, which restricts the options of a traveling user.

In June 2020, the EU has published an interoperability architecture based on an EU federation gateway that connects national databases: all countries can use this gateway to upload their keys from their national database and can download the keys from the other national databases. This gateway will become operational in mid September 2020; 11 EU countries will be involved in the first phase. It is expected that keys can be exchanged with non-EU countries based on bilateral agreements. The Belgium Coronalert app will join the EU gateway in October 2020.

Consider now a Belgian citizen who has installed the Belgian app and who is traveling abroad to country X that also uses the DP-3T protocol based on the Google/Apple Exposure Notification API. As the exchange of Bluetooth tokens has been standardized, the app will operate normally, in the sense that it will broadcast and receive Bluetooth tokens. We distinguish the following three scenarios.

- If the citizen wants to evaluate her exposure risk, she has to indicate in his app that she has travelled to country X. Her app will then download the TEK keys of country X from the Belgian central server. (Note that a Belgian app will not know the internet address of the server of country X, hence a direct download is not an option.)
- If the citizen undergoes a test in Belgium and the outcome is positive, she will be requested to upload her TEK keys to the central server of Belgium. During the upload, she will be asked to indicate which countries she has visited during the last days. If she indicates that she has visited

country X on a particular day, her TEK key from that day will be forward by the Belgian national server to the EU federation gateway, so that the users of the app in Country X can be informed about their exposure risk.

- If the citizen undergoes a test in Country X, it will not be possible to upload her keys directly to the server of country X. Moreover, it is unlikely that the healthcare infrastructure of Country X will support the Belgian authorization protocol, hence an upload to the Belgian server is also not possible. However, ad hoc procedures could be developed between national health authorities to transfer test results that may involve a dedicated authorization process. The alternative is to perform only manual contact tracing.

5.6 Monitoring the operation of the system

The strong "privacy by design" nature of the Coronalert app makes it non-trivial to monitor the correct operation of the system. The following information can be collected without interfering with the privacy features of the app:

In the app stores, one can monitor the number of downloads and the number of uninstalls of the app; one can also analyze the rating the app receives from the users.

In the database for epidemiological data and manual contact tracing one can monitor the following elements:

- The number of PCR tests applied for with an app (that is the tests for which a code R1 is submitted).
- The number of tests applied for because users were informed by their app of a high-risk contact; this number may help to understand which high-risk contacts can be identified by the app but not by manual tracing. This assumes that doctors report this number.
- The number of users contacted for manual tracing who indicate that they were informed by their app of a high risk contact. This assumes that the scripts for manual contact tracing are updated to request this information.

In the Central database (DB1) one can monitor the following elements:

- The number of uploaded diagnosis keys (TEKs).
- The number of uploaded authorization codes.
- The number of active apps (how many apps download diagnosis keys and new parameters).

In the Polling server (DB2) one can monitor the number of downloaded test results (positive and negative).

If the international interoperability scheme is active, the number of diagnosis keys sent and received from each country can be monitored.

In order to develop a more in-depth understanding, one could perform a study with a group of users of the app and collect their experiences.

6 Web links and documents

DP-3T

https://github.com/DP-3T/documents

Google/Apple

https://www.google.com/covid19/exposurenotifications/
https://www.apple.com/covid19/contacttracing/

Wikipedia

https://en.wikipedia.org/wiki/COVID-19_apps

Public health landscape

https://landscape.lfph.io/

Belgian overview page on contact tracing

https://www.corona-tracking.info/

Belgian public consultation on Coronalert app

https://www.esat.kuleuven.be/cosic/sites/corona-app/

Sciensano document:

https://covid-19.sciensano.be/sites/default/files/Covid19/COVID-19_procedure_GP_NL.pdf,
https://covid-19.sciensano.be/sites/default/files/Covid19/COVID-19_procedure_GP_FR.pdf

Belgium legal documents

- Arrêté royal n° 44 concernant le traitement conjoint de données par Sciensano et les centres de contact désignés par les autorités régionales compétentes ou par les agences compétentes, par les inspections sanitaires et par les équipes mobiles dans le cadre d'un suivi des contacts auprès

des personnes (présumées) infectées par le coronavirus COVID-19 sur la base d'une base de données auprès de Sciensano (26 June 2020) https://www.corona-tracking.info/wp-content/uploads/2020/06/KB-nr-44.pdf;

- APD-GBA Avis 64/2020: Demande d'avis concernant un projet d'accord de coopération entre l'État fédéral, la Communauté flamande, la Région wallonne, la Communauté germanophone et la Commission communautaire commune, concernant le traitement conjoint de données par Sciensano et les centres de contact désignés par les autorités régionales compétentes ou par les agences compétentes, par les inspections sanitaires et par les équipes mobiles dans le cadre d'un suivi des contacts auprès des personnes (présumées) infectées par le coronavirus COVID-19 sur la base d'une base de données auprès de Sciensano (CO-A-2020-076) https://www.autoriteprotectiondonnees.be/publications/avis-n-64-2020.pdf

EU documents

- Commission Recommendation (EU) 2020/518 of 8 April 2020 on a common Union toolbox for the use of technology and data to combat and exit from the COVID-19 crisis, in particular concerning mobile applications and the use of anonymised mobility data https://op.europa.eu/en/publication-detail/-/publication/1e8b1520-7e0c-11ea-aea8-01aa75ed71a1/language-en;
- Common EU Toolbox for Member States of the eHealth Network (15 April 2020) https://ec.europa.eu/health/sites/health/files/ehealth/docs/covid-19_apps_en.pdf;
- Communication from the Commission Guidance on Apps supporting the fight against COVID 19 pandemic in relation to data protection 2020/C 124 I/01 https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=OJ:C:2020:124I:TOC;
- Resolution of the European Parliament of 17 April 2020 on EU coordinated action to combat the COVID-19 pandemic and its consequences https://www.europarl.europa.eu/doceo/document/TA-9-2020-0054_EN.html;
- Guidelines 04/2020 of the European Data Protection Board of 21 April 2020 'on the use of location data and contact tracing tools in the context of the COVID-19 outbreak' https://edpb.europa.eu/sites/edpb/files/files/file1/edpb_guidelines_20200420_contact_tracing_covid_with_annex_en.pdf;
- Technical specifications for interoperability of contact tracing apps - eHealth Network Guidelines to the EU Member States and the European Commission on Interoperability specifications for cross-border transmission chains between approved apps https://ec.europa.eu/health/sites/health/files/ehealth/docs/mobileapps_interoperabilityspecs_en.pdf

Denmark

https://smittestop.dk/

Germany

https://www.coronawarn.app/en/

https://www.rki.de/DE/Content/InfAZ/N/Neuartiges_Coronavirus/WarnApp/Warn_App.html

Ireland

https://covidtracker.gov.ie/

Italy

https://www.immuni.italia.it/

Latvia

https://www.apturicovid.lv/#en

The Netherlands

https://coronamelder.nl/

https://www.rijksoverheid.nl/onderwerpen/coronavirus-app

Switzerland

https://www.bag.admin.ch/bag/en/home/krankheiten/ausbrueche-epidemien-pandemien/aktuelle-ausbrueche-epidemien/novel-cov/swisscovid-app-und-contact-tracing.html

Selected epidemiology papers

R. Hinch W. Probert, A. Nurtay, M. Kendall, C. Wymant, M. Hall, K. Lythgoe, A. Bulas Cruz. L. Zhao, ,A. Stewart, L. Ferretti, M. Parker, A. Meroueh, B. Mathias, S. Stevenson, D. Montero, J. Warren, N.K. Mather, A. Finkelstein, L. Abeler-Dörner, D. Bonsall, C. Fraser, Effective Configurations of a Digital Contact Tracing App: A report to NHSX, 16 April 2020, https://045.medsci.ox.ac.uk/files/files/report-effective-app-configurations.pdf

https://www.research.ox.ac.uk/Article/2020-04-16-digital-contact-tracing-can-slow-or-even-stop-coronavirus-transmission-and-ease-us-out-of-lockdown https://www.technologyreview.com/2020/06/05/1002775/covid-apps-effective-at-less-than-60-percent-download/

A. Barrat, C. Cattuto, M. Kivelä, S. Lehmann, J. Saramäki, Effect of manual and digital contact tracing on COVID-19 outbreaks: a study on empirical contact data, July 25, 2020.
https://www.medrxiv.org/content/10.1101/2020.07.24.20159947v1

A. Updatable parameters of the app

This appendix lists the parameters of the app that should be updatable on a daily basis as signed parameters that are distributed together with the TEK keys. In version 1.0, these parameters will be hard coded, but from version 1.1 onwards these parameters can be securely updated remotely.

- Formula to compute the date t0 on which the patient became infectious from ts, the date of onset of symptoms or t1 the date the sample is taken. The current formulas are t0 = ts – 2 and if there are no symptoms t0 = t1 – 2.
- Maximum number of days a patient is infectious. Currently the value of this parameter is 14, which means that test results are removed from the polling server DB2 after t0+14 days.
- The values tr1, tr2, w1, w2, and w3 that are used to estimate the exposure risk. The current values are tr1=55 dB, tr2=63 dB, w1=1, w2=0.5, w3=0 (cf. Section 5.3).

B. Computation of the error detection code modulo 97

Section 5.2 mentions the use of an error detection code modulo 97 to detect errors in R1 and t0. This appendix describes how this error detection code can be computed efficiently.

The date t0 is presented as 6 digits in the format year (2 digits) month (2 digits) day (2 digits): example:

t0 = 200813 (August 13, 2020)

R1 = 123456789012345

The two check digits are then calculated on

200813123456789012345 .

We find these two check digits c as follows (where % 97 stands for modulo 97):

c = (97 - (200813 123456789012345 * 100) % 97) .

The result is then represented by 23 digits:

t0 (6 digit date)

R1 (15 digits)

c (2 digits)

Example 1 (with R1 of only 5 digits)

t0 = 191205 (December 5 , 2019)

R1 = 27182

integer = 19120527182

(19,120,527,182 * 100) % 97 = ((19,120,527,182 % 97) * (100 % 97))% 97 = (90 * 3)% 97 = 76

Note that 19,120,527,182 = 19 * 10^9 + 120,527,182 hence
19,120,527,182 % 97 = (19 * 10^9) % 97 + (120,527,182) % 97)% 97 =
(19 * 34 + 26) % 97 = 672 % 97 = 90.

97 - 76 = 21

resulting value:

1912042718221 with 1912042718221 modulo 97 = 0.

Example 2 (with R1 of 15 digits)

t0 = 200813 (August 13, 2020)

R1 = 123456789012345

integer = 200,813,123,456,789,012,345

(200813123456789012345 * 100) modulo 97 = 37 (see detail below)

97 - 37 = 60

resulting value: 20081312345678901234560

Split as

t0: 200813

R1 || c = 20081312345678901234560

How do I calculate with 23 decimal digits without a special library?

20,081,312,345,678,901,234,500 = 20,081 * 10^{18} + 312,345,678 * 10^9 + 901,234,500.

You can now write 10^{18} as 10^9* 10^9; now every term is less than 10^9 so you can compute the value % 97 with single precision (9 digits or 30 bits). This gives the following steps:

10^9 % 97 = 34 and 10^{18} % 97 = (34 * 34) % 97 = 89

20,081 % 97 = 2; 312,345,678 % 97 = 52; 901,234,500 % 97 = 31

(2 * 89 + 52 * 34 + 31) % 97 = 37.