Analysis of SwissCovid

Serge Vaudenay[1] and Martin Vuagnoux[2]

2020, June 5

[1] EPFL, Lausanne, Switzerland
[2] base23, Geneva, Switzerland

What follows is the original June 5 version to which we subsequently added an important addendum.

Abstract. We present an analysis of the SwissCovid application which is currently being tested. We observe that the essential part of SwissCovid is under the control of Apple and Google. Outsourcing the heart of SwissCovid to Apple and Google has apparent benefits in terms of security but drawbacks in terms of transparency, flexibility, and sovereignty.
We observe that SwissCovid is far from being open source. The Source code is kept by Microsoft. The protocol is implemented and controlled by Apple and Google. The server is hosted by Amazon. The current information suffers from unclear or incorrect statements.
We confirm some of the threats which had been identified before. Users may be traced or identified by third parties while tracing is on. Diagnosed users who report using SwissCovid have a risk to be identified by a third party. Malicious users may create false encounters and inject false at-risk notifications on targeted phones. They could abuse the system to have vacations paid by authorities by self-injecting false alerts. Diagnosed users could be corrupted to sell a covidcode which would ease those attacks. Malicious apps could collect more information or do the job of SwissCovid outside of any control, and on behalf of a third party, even though SwissCovid is deactivated.

1 Introduction

Contact tracing is a way to reduce the spread of a pandemic. Automated contact tracing is being implemented and deployed to improve the accuracy and efficiency of manual contact tracing. Many countries developed automated contact tracing using several technology. At this time, the prominent technology is based on Bluetooth. In Switzerland, the DP3T project announced its development on April 1st, 2020. Its design convinced Apple and Google[3], as well as many governments including Switzerland.

One difficulty of using Bluetooth in phones is that the operating system does not allow apps running in the background to use the Bluetooth advertising system. Hence, to do so, either the app must stay in the foreground, which drains the battery a lot, or the operating system must be changed. Apple and Google allied to provide a standard Bluetooth API which would not drain the battery. Hence, automated contact tracing apps must either be compliant with this API or make the user upset about heavy battery usage.

The Epidemics Act gives means to the government to reduce the pandemics. Those means could be to reduce human rights, such as the freedom to move (by locking down) or the right for privacy (by tracing people who could carry the virus). Those means are made possible by

[3] So claim DP3T and the press https://www.reuters.com/article/us-health-coronavirus-apple-alphabet/apple-google-plan-software-to-slow-virus-joining-global-debate-on-tracking-idUSKCN21S1TT. We did not find any Google-Apple document confirming this.

law as long as they are *necessary* and *reasonable*. One main objective of DP3T was to allow accurate and efficient contact tracing with minimal privacy loss.

So far, the direction taken by the Swiss government is to deploy such system and count on the voluntary use by residents. Since a key factor for the effectiveness is the adoption rate, the government is investing on a communication campaign about SwissCovid.[4] However, at the same time, SwissCovid is considered as a medical device hence must follow the regulation which enforces communication about the risks for the user to be complete and accurate. Actually, users need information to give their *aware consent*.

Switzerland released the test version of its automated contact tracing app on May 25th, 2020.[5] In this report, we review security and privacy issues which come with this system. Our work was based on the provided test app, standard phones, and the publicly available documents. We took for granted that SwissCovid would be necessary and reasonable but would require the aware consent of users. Hence, our objective is twofold. We aim at providing feedbacks to the SwissCovid effort. At the same time, we strive to make potential users aware of the risks this technology is creating.

We first describe the internal structure of SwissCovid. Then, we make a few observations.

- It was initially hard to obtain the source code. We regret it is not documented. The fact that it is constantly updated made it impossible to make any real analysis. In general, complete technical specifications are missing. We fear that the entire system will be adopted without any real public security analysis (such as what was made for e-voting).
- There is a misconception on the meaning of *open source*. SwissCovid is not open source and will unlikely be. The SwissCovid app and servers may have and available source code but it will be impossible to compile and run. Furthermore, GAEN, the crucial functioning heart of SwissCovid is implemented outside the app, has no source code, and is not subject to any external audit. The apparent principle of publicly available source code for SwissCovid is twisted by excluding (by law) GAEN from the components of SwissCovid.
  The source code is stored on GitHub which belongs to Microsoft.
- SwissCovid requires users to give personal information to Google-Apple while SwissCovid is forbidden to collect such information from users. The GAEN heart of the system is not subject to any independent audit. Consequently, the decentralized DP3T system has now become a non-transparent centralized one.
- Some SwissCovid servers are hosted by Amazon.
- The metadata which is broadcasted by Bluetooth can be maliciously modified. This may ease false at-risk alert injection attacks.
- Replay attacks are possible during two hours.
- Having the Google-Apple closed solution gives some apparent security benefit, but such risky approaches to security have shown to fail in the past.
- Making SwissCovid interoperable across borders may not be easy.
- The voluntary report after positive diagnosis does not allow the user to exclude some hour windows from his report.
- DP3T is making change suggestions in the protocol which are up to Google-Apple to consider.
- Considering SwissCovid as a medical device has important consequences as for the obligations of the government.

[4] https://www.admin.ch/opc/fr/federal-gazette/2020/4361.pdf

[5] It was meant to be limited to a small population of testers but ended up being widely available on https://play.google.com/store/apps/details?id=ch.admin.bag.dp3t

- The current information about privacy risks is very confuse and may inspire mistrust.
- Some provided information is incomplete or inaccurate.
- SwissCovid gives no proof of at-risk notification but this may be needed for claiming subsidies.
- The distinction between consent and liability should be made clear.

Finally, we describe the threats about security and privacy.

- We list a few previously reported attacks.
- Using the Bluetooth technology in SwissCovid eases anyone to make his own surveillance system. We can register in real time how many people are present in a nearby apartment with cheap equipment. We can track the movements of people in a building with a few sensors. A fraction of phones may be identifiable by active scanning.
- Identifying reported users is easy for someone who has briefly met and identify the person before. Likewise, creating false encounters is easy. There are several ways to inject false at-risk notifications and to send people to quarantine.
- Some malicious apps could take advantage of SwissCovid to determine where and when contagious people have been, identify contact with contagious people even though SwissCovid does not consider it as a significant risk, identify diagnosed persons, and locate devices with localization turned off.
- The network may identify who has been diagnosed (however, we could not verify this).
- Malicious phones can of course learn a lot about the holder, but can also locate the phone even though location was turned off.
- Amazon may maliciously provide wrong content, which would allow to easily inject false at-risk notifications to targeted users. However, we could not verify this. Servers, in general, can see an IP address and a user agent which says something about the querying phones.

2 Overview of SwissCovid

2.1 Architecture

The SwissCovid system is essentially made of

- users,
- the Bluetooth API of the smartphone of users,
- the app installed on the smartphone of users,
- the code management system (covidcode server),
- a server (cloud server),
- the health authority.

Other organizations play an important role:

- the app editor,
- the API editor,
- the communication network.

In the above architecture, it is important to separate the app from the Bluetooth API because they are made by separate organizations. In this report, we refer to the Bluetooth API as the GAEN API (as for Google-Apple Exposure Notification). It is specified in several documents [1–4]. The operating system of the smartphone controls that no other app than the one authorized by Google-Apple has access to the GAEN API. Following the GAEN terms [4], the app is granted access to the GAEN API provided that

- the app is the only app which is approved by the local government public health authorities;
- the app "*is used exclusively for COVID-19 response efforts*";
- the app does not require data that identifies the user nor which can be used to determine their religion/age/gender/nationality/...;
- the app lets the free consent of the user;
- the app minimizes the data collection to what is necessary for the purpose;
- the app does not use location.

The app is supposed to implement the DP3T protocol which is described in two documents [5, 6]. However, due to the constraints to use the GAEN API, very little of DP3T is left in the app itself. Essentially, the app is

- offering a graphic interface to the user;
- handling communication between the server and the GAEN API;
- computing the infection risk from precomputed information from the GAEN API.

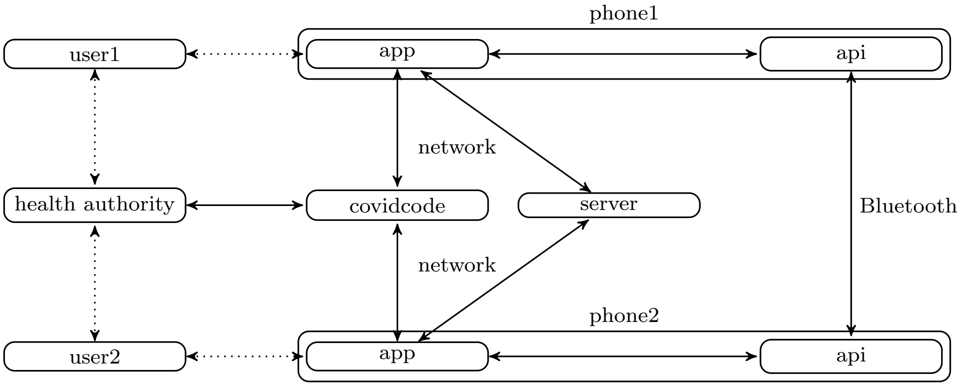


Fig. 1. Architecture of SwissCovid

The communication architecture is represented on Fig. 1. Plain arrows represent automated communication between devices. Dotted arrow involve a human being. Only two users and two phones are represented.

2.2 The GAEN API

On Android, the GAEN API is part of the Google Play Services. Installing it from the Google store requires to have a **gmail** account. Hence, GAEN necessarily associates the device to an account.

The API controls how cryptographic keys are defined and used [2]. We use the following keys:

- TEK: Temporary Exposure Key
- RPIK: Rolling Proximity Identifier Key
- AEMK: Associated Encrypted Metadata Key

– RPI: Rolling Proximity Identifier
– AEM: Associated Encrypted Metadata

In this report, we assume the following default parameters (which are suggested in the GAEN documentation):

– a TEK is used for one day;
– a RPI is used for about 10 minutes;[6]
– an encounter is always stored if it lasts 5 minutes;[7]
– an encounter is considered in the risk calculation only if it is more than 15 minutes and at a distance less than 2m.[8]

Every day, a new TEK is selected at random. We measure the time in 10-minute periods based on the UNIX time (i.e. it does not depend on the time zone). If TEK is the valid key during time j, the API defines

$$\begin{aligned}\mathsf{RPIK} &= \mathsf{HKDF1}(\mathsf{TEK})\\ \mathsf{AEMK} &= \mathsf{HKDF2}(\mathsf{TEK})\\ \mathsf{RPI} &= \mathsf{AES}_{\mathsf{RPIK}}(j)\\ \mathsf{AEM} &= \mathsf{Meta} \oplus \mathsf{trunc}(\mathsf{AES}_{\mathsf{AEMK}}(\mathsf{RPI}))\end{aligned}$$

where $\mathsf{HKDF1}$ and $\mathsf{HKDF2}$ are key derivation functions based on HKDF with two different labels as input, Meta contains some metadata, trunc truncates to the length of Meta. Currently, the metadata consists of a version number of the API and the power of the Bluetooth broadcast signal as set up by the API for sending. This power is represented by a byte. It indicates a value between $-127\mathsf{dBm}$ and $+127\mathsf{dBm}$. Two bytes are reserved in the metadata for future use.

During the time period j, the API broadcasts $\mathsf{RPI}\|\mathsf{AEM}$ with the value of the exposure notification service which is `0xFD6F` [1]. Broadcasting is done constantly (several times per second).

In parallel, the API scans for broadcasts from others with the exposure notification service `0xFD6F`. If the encounter is significant, the API stores the broadcasted values $\mathsf{RPI}\|\mathsf{AEM}$ together with a timestamp and the RSSI (Received Signal Strength Indication) [1].

The app can prompt the API to reveal the TEK which have been used recently. If so, the API asks for the authorization to the user.

The API can also receive from the app a list of (TEK, t) pairs based on which the API finds matches with received $\mathsf{RPI}\|\mathsf{AEM}$. By decrypting AEM when a match is found, the distance to the infected person can be estimated. The API returns to the app an *exposure detection summary* which contains the number of matches, the most recent match day, the highest risk score among matches, and the sum of all durations for matches in three different categories of risk scores [3]. If the app finds it significant enough, it may asks for more information. The API asks for the authorization to the user for that and returns the list of day/duration/risk for each match.

[6] We observed that beacons rotate about every 12–14 minutes so it may rather be a bit larger in real.

[7] An answer to the Webinar by DP3T set on 27.5.2020 says that all received beacons are stored, so it could be 0 minutes instead.

[8] This suggests that Coronavirus can fly over 2m in Switzerland, while its flying distance seems to be limited to 1m in France!

2.3 The App

The app is the interface between the API, the user, and the server. It triggers essentially two actions:

– After notification of a positive COVID-19 test, the user gets a 12-digit one-time code (called *covidcode*) from the Health authority and enters it to the app. The app sends it to the covidcode server which verifies the validity of this code. If valid, the app retrieves from the API the list of recently used (TEK, t) (the API prompts the user to authorize), and uploads them to the server.

– Regularly, the app downloads from the server the list of reported (TEK, t) pairs, provides them to the API, get the exposure detection summary, determines if more information is needed to compute the risk, if so, gets it from the API, computes the infection risk and notifies the user accordingly.

The only security-sensitive task of the app is to connect securely to the server. For this, a TLS connection with a pinned certificate is used. The app does not authenticate to the server. Actually, anonymity is a strong privacy requirement in the case of an upload.

In addition to this, the app seems[9] to make some fake requests with covidcode 000000000000 to the code management server in order to hide when a request is not fake. Fake requests are made at random time intervals. The distribution of the random delay is exponential, multiplied by 5 days.[10] This means that the average delay is of 5 days.

On the app, the user can turn on and off tracing. When tracing is turned on, the app regularly complains if Bluetooth is off because it needs it.[11]

The main task of the app is to provide appropriate information to the user.

2.4 The Health Authority

The health authority provides the user with a one-time access code (the covidcode) so that the user can upload his data on the server. For this, the health officer connect to the code management system (covidcode server) which gives a fresh code.

2.5 The Code Management System

One server is called *code management system.* It is used to generate and verify the covidcode given by the health authority. One-time access codes are kept valid for 24 hours.[12]

The server simply generates a fresh code and stores it in a database with its validity period (from creation, valid for 24h). When the app sends the covidcode, the server simply retrieves it from the database and removes it. The server then generates a JSON Web Token (JWT) which is signed using RSA PKCS#1 with SHA2. This token is verified by the other server.

[9] We have not observed any such fake query so far.

[10] https://github.com/DP-3T/dp3t-app-android-ch/blob/develop/app/src/main/java/ch/admin/bag/dp3t/networking/FakeWorker.java

[11] When Bluetooth is off, we confirm that the phone does not broadcast anything but we could not confirm that it does not scan to continue to receive beacons. Actually, it is well known that turning off Bluetooth does not prevent apps from Scanning Bluetooth.

[12] https://www.bag.admin.ch/swisscovid-data-protection-statement-and-conditions-of-use
SwissCovid App : Data Protection Statement, Section 5.

2.6 The Server

The server, called *VA back end*, actually separates two servers for upload and download (each server may be split for load balancing). The upload server is infrequently used, as it is only used by diagnosed users. The download server is quite frequently used by all users to retrieve recent uploaded keys. It is also used to retrieve some configuration parameters.

The VA back end server accepts any TLS connection. Any client can read the stored (TEK, t) pairs on the download server. The upload server accepts new (TEK, t) pairs and stores them after checking a JWT token.

Data on the back end server is erased after 21 days.[13] Access to the servers are logged and the log data is erased after 7 days.[14]

By inspecting the requests from the app to the server https://www.pt.bfs.admin.ch/v1/config we could figure out that three parameters must be added in the URL:

– `appversion` (e.g. `android-1.0pilot`)
– `osversion` (e.g. `android23`)
– `buildnr` (e.g. `1590183410069`)

This request returns a JSON response with some configuration parameters to be given to the GAEN API. For instance, it updates some parameters

$$\texttt{lowerThreshold}, \texttt{higherThreshold}, \texttt{factorLow}, \texttt{factorHigh}, \texttt{triggerThreshold}$$

which are used for the risk calculation. The response header includes a signature which is added, although communication goes through TLS.[15]

To retrieve from the download server, anyone can make a query of type

https://www.pt.bfs.admin.ch/v1/gaen/exposed/⟨date⟩

to get the new keys since a given date. The request returns nothing, which is not surprising because the app has been tested by a few thousand careful people since only 12 days, and most likely no one has reported any exposed key. However, this empty list comes with a header of type

```
HTTP/2 204 No Content
cache-control: max-age=300
date: Fri, 05 Jun 2020 15:57:21 GMT
signature:
eyJhbGciOiJFUzI1NiJ9.eyJjb250ZW50LWhhc2giOiI0N0RFUXBqOEhCU2ErL1RJbVcrNUpDZXV
RZVJrbTVOTXBKV1pHM2hTdUZVPSIsImhhc2gtYWxnIjoic2hhLTI1NiIsImlzcyI6ImRwM3QiLCJ
pYXQiOjE1OTEzNzI2NDEsImV4cCI6MTU5MzE4NzA0MX0.hL7wv3Wmjd_ofME5U2SxJJLzF2gWWFo
b7A0stgDDcw45uC_XbRkeRiek664p2ffiyGBykDm6cAB5yq066QWsJw
strict-transport-security: max-age=16070400; includeSubDomains
x-content-type-options: nosniff x-frame-options: DENY
x-published-until: 1591365600000
```

[13] https://www.bag.admin.ch/swisscovid-data-protection-statement-and-conditions-of-use
SwissCovid App : Data Protection Statement, Section 5.

[14] https://www.bag.admin.ch/swisscovid-data-protection-statement-and-conditions-of-use
SwissCovid App : Data Protection Statement, Section 8.

[15] We think that the signature is added to defeat malicious modification by a content delivery network (CDN).

```
x-vcap-request-id: 7f6fb92f-c411-4b18-7fce-c7fb0b385ebc
x-xss-protection: 1; mode=block
x-cache: Miss from cloudfront
via: 1.1 449f2b51e83bf8ba5fa5e65ce60bc277.cloudfront.net (CloudFront)
x-amz-cf-pop: ZRH50-C1
x-amz-cf-id: H1bvk-G3RItqbFIA3RuhPhOxy4HnVa6wUf_-MBMHexviD2pIJShq5g==
X-Firefox-Spdy: h2
```

How the protocol works to report with those servers is not clear at the moment of writing. One suggestion by Google-Apple[16] is that the exposure request first goes to the code management system which verifies the covidcode (called PIN in the Google-Apple document) and returns an ECDSA signature on the report. The report and signature must then be uploaded on the cloud server. This server checks the signature before storing.

What we understood by inspecting SwissCovid is that the app sends to the upload server the JWT token to with it attaches the keys to report. The upload server then rearrange them for storage.

3 Observations

3.1 Availability of the Source Code

The Federal Act relative to the trial of the pilot SwissCovid system by May 13, 2020[17] Art.8 al.4 says:

> *Le code source et les spécifications techniques de tous les composants du SPTS sont accessibles au public.*[18]

Art.5 defines the components of SwissCovid to be the app, the code management system, and the backend server. It does not consider the GAEN API as being a component.

It was not easy to figure out where are the documentations and the source codes until we have been pointed to the following links

https://www.melani.admin.ch/melani/en/home/public-security-test/infos.html
https://www.melani.admin.ch/melani/en/home/public-security-test/scope_and_rules.html

It points to the following links

Android app: https://github.com/DP-3T/dp3t-app-android-ch
iOS app: https://github.com/DP-3T/dp3t-app-ios-ch
app config backend: https://github.com/DP-3T/dp3t-config-backend-ch
app backend: https://github.com/DP-3T/dp3t-sdk-backend
covidcode frontend: https://github.com/admin-ch/CovidCode-UI
covidcode backend: https://github.com/admin-ch/CovidCode-Service

The source codes have nearly no comment at all and we could not find the promised technical specifications. We hope it will code in the future.

[16] https://github.com/google/exposure-notifications-server/blob/master/docs/design/verification_protocol.md
[17] https://www.admin.ch/gov/fr/accueil/documentation/communiques.msg-id-79103.html
[18] The source codes and technical specifications of all components are publicly available.

The repository of the DP3T project contains lots of source codes (in addition to the above ones). Documents [5] describe the DP3T scheme but not the SwissCovid one. Actually, most of the DP3T scheme is replaced by GAEN and the part of the architecture related to the server is nearly not covered at all.

The repositories of the GAEN API contains a sample code for an app and for a server. The GAEN API has no source code available but is fully documented.

The repositories for the covidcode frontend has nearly no documentation.

In addition to this, we stress that, at the time of testing the code, the source codes in the repositories are frequently updated. This means that the current implementation is most likely outdated, compared to the currently available source code. Hence, the significance of making a security test on an always-updating system is very limited. We understand this is the consequence of imposing severe time constraints on the development but **we wonder if the current plan is to deploy a dangerous and controversial system without giving time to a proper public security audit on the final version**. The public security audit was the approach taken to deploy e-voting in 2019 and we remember the system miserably failed to pass.

3.2 Open Source

All developers and politics require SwissCovid to be *open source*, although this term may have several misconceptions. The DP3T repository claims[19]

> *Open source implementations for iOS, Android, and the back-end server are available in other DP-3T repositories. The DP-3T app developed for Switzerland is publicly available Android and iOS and can be used as the basis for other apps.*

The developers wrote a petition[20] which was signed by several hundreds of academics and which made governments change to their solutions, in which they require

> *Any considered solution must be fully transparent. The protocols and their implementations, including any sub-components provided by companies, must be available for public analysis.*

(which does not exactly mean open source). Note that at that time, DP3T was explicitly including "sub-components provided by companies" which we understand is including the GAEN API.

The 20.5.2020 proposal by the Swiss government to modify the law on epidemics says (Art.60a al.4)

> *le code source et les spécifications techniques de tous les composants du système TP sont accessibles au public.*[21]

(which does not exactly mean open source). Depending on how they define the components, this may include the GAEN API or not.

The Social Security and Health Committees of the Swiss Council of States (SSHC-S) praised[22] the *open source approach* and further recommends

[19] https://github.com/DP-3T/documents, accessed 27.5.2020

[20] https://www.esat.kuleuven.be/cosic/sites/contact-tracing-joint-statement/

[21] The source codes and technical specifications of all components are publicly available.

[22] https://www.parlament.ch/press-releases/Pages/mm-sgk-s-2020-05-26.aspx

> *l'application doit être réalisée, de manière avérée, à l'aide du code source accessible au public (art. 60a, al. 4, let. e).*[23]

which means that we should be able to verify that SwissCovid was made from the provided source code. This committee reported to the Council of States on 3.6.2020[24]

> *Positiv bewertet die Kommission, dass das System auf Open Source beruht und damit öffentlich ist. Die Kommission hat dies mit einer Ergänzung von Buchstabe e in Artikel 60a Absatz 4 noch verdeutlicht.*[25]

However, "open source" was not added in Art.60a letter e but instead

> *e. sind öffentlich. Die maschinenlesbaren Programme müssen nachweislich aus diesem Quellcode erstellt worden sein.*[26]

Which requires the app to be provably coming from the available source code.

Being open source does not only mean that the source code should be available. It also means that anyone should be able to take the source code, to modify it, to compile it, and to use it. However, there are aspects making SwissCovid not open source:

- the source code of the app is available but if we compile it ourselves, we cannot use it because only the genuine executable has access to the GAEN API;
- there is currently no way to verify that the provided source code is the one which was compiled in the genuine app (except by reverse engineering);
- the app contains nothing related to the management of the ephemeral identifiers, storage, and matching; nearly everything sensitive is handled by the GAEN API which has no source code available and which we will never be able to compile nor to analyze.

Currently, the last point is strangely addressed by excluding GAEN from the SwissCovid system. Indeed, the Federal Act relative to the trial of the pilot SwissCovid system by May 13, 2020[27] defines the components with this excluding approach. The update proposal on the Epidemics Act does not define the components but requires them to have available source codes. Most likely, a government ordinance to define it will follow. However, this will not change that no one can compile and execute SwissCovid.

The Wikipedia page[28] on the open source definition says that the common definition of open source (after 25 years of experience) is characterized by 10 criteria.

Experts in security often analyze APIs on Linux machines, find vulnerabilities, and fix them. With the API on iOS or Android, this will not be the case.

Therefore, **SwissCovid is far from being open source**. At best, the source codes of the graphic user interface is available but it can neither reproduce the running app nor be modified.

Finally, we observe that the source code is stored on GitHub which belongs to Microsoft.

[23] The application must be verifiably made from the publicly available source code.

[24] https://www.parlament.ch/fr/ratsbetrieb/amtliches-bulletin/amtliches-bulletin-die-verhandlungen?SubjectId=48973

[25] The Commission is positive on that the system is based on open source and available to the public. The Commission completed this by adding point e to Article 60a Paragraph 4.

[26] e. are public. The installed programs must verifiably be created from this source code.

[27] https://www.admin.ch/gov/fr/accueil/documentation/communiques.msg-id-79103.html

[28] https://en.wikipedia.org/wiki/The_Open_Source_Definition

3.3 Apple-Google has the Technical Control on SwissCovid

There are some intriguing signs in the relation to Google-Apple. The DP3T project is now calling his partner Google-Apple through a public post to, at least, open the API to external audit and to update their implementation:[29]

> *We also strongly believe that Apple and Google should adopt our subsequent enhancements, detailed in our white paper, that increase user privacy. We also strongly encourage both companies to allow an external audit of their code to ensure its functionality corresponds to its specification.*

This makes us believe that **DP3T has lost control on SwissCovid**.

SwissCovid cannot work on "deGoogled" Android phones.[30] On Android, the GAEN API is part of Google Play Services which requires the user to register a **gmail** account.

The current situation with Google-Apple actually puts SwissCovid in a strange situation. Indeed, the terms of GAEN [4] prohibits the SwissCovid app to use any personal information on the user or anything which relates to its gender, religion, etc.

However, by registering a **gmail** account, the user is asked for a gender (although this is optional), age, a phone number. It will also register the IMEI number of the phone and many other identifying information. Hence, **users must consent to release their personal information to Google-Apple while SwissCovid is forbidden to do so**.

Similarly, the SwissCovid app is forbidden to use location. However, Google Play Services use access to wearable devices, photos, location, files, calendar, storage, phone, microphone, device ID, camera, contacts, Wi-Fi, device state and history, identity, SMS, and many other privileges.

As iOS is closed, we could not say anything but it is likely to be the same.

There was a controversy about adopting centralized or decentralized systems [7]. We can see now that **the DP3T decentralized system became a non-transparent one which is centralized on Google-Apple services**.

3.4 Location of the Server

We could find some IP addresses with a pinned certificate inside the app:

```
www.pt-d.bfs.admin.ch (sha256/xWdkLqfT40GnyHyZXt9IStltvrshlowMuGZHgp631Tw=)
www.pt1-d.bfs.admin.ch (sha256/Pr8nx8M3Oa8EYefVXYB3D4KJViREDy4ipA1oVyjGoss=)
codegen-service-d.bag.admin.ch (sha256/GLV5pALJpJb02GavguT3NTOmOL57H7K3KhJ59iH6A/Q=)
www.pt-t.bfs.admin.ch (sha256/KM3iZPSceB+hgYuNI+cSg4LRgTiUxCeGjrfXRQAY6Rs=)
www.pt1-t.bfs.admin.ch (sha256/KM3iZPSceB+hgYuNI+cSg4LRgTiUxCeGjrfXRQAY6Rs=)
codegen-service-t.bag.admin.ch (sha256/KM3iZPSceB+hgYuNI+cSg4LRgTiUxCeGjrfXRQAY6Rs=)
www.pt-a.bfs.admin.ch (sha256/KM3iZPSceB+hgYuNI+cSg4LRgTiUxCeGjrfXRQAY6Rs=)
www.pt1-a.bfs.admin.ch (sha256/KM3iZPSceB+hgYuNI+cSg4LRgTiUxCeGjrfXRQAY6Rs=)
codegen-service-a.bag.admin.ch (sha256/KM3iZPSceB+hgYuNI+cSg4LRgTiUxCeGjrfXRQAY6Rs=)
www.pt.bfs.admin.ch (sha256/KM3iZPSceB+hgYuNI+cSg4LRgTiUxCeGjrfXRQAY6Rs=)
www.pt1.bfs.admin.ch (sha256/KM3iZPSceB+hgYuNI+cSg4LRgTiUxCeGjrfXRQAY6Rs=)
codegen-service.bag.admin.ch (sha256/KM3iZPSceB+hgYuNI+cSg4LRgTiUxCeGjrfXRQAY6Rs=)
```

[29] https://github.com/DP-3T/documents, accessed 27.5.2020

[30] https://github.com/DP-3T/dp3t-app-android-ch/issues/59
https://github.com/DP-3T/dp3t-app-android-ch/issues/60

Some of those servers are hosted by Amazon (`www.pt.bfs.admin.ch`, `www.pt-t.bfs.admin.ch`, `www.pt-a.bfs.admin.ch`, `www.pt-d.bfs.admin.ch`).

To see it, the command `host` gives the corresponding IP addresses.

```
www.pt.bfs.admin.ch is an alias for da6aw9bvreynb.cloudfront.net.
da6aw9bvreynb.cloudfront.net has address 13.224.95.88
da6aw9bvreynb.cloudfront.net has address 13.224.95.73
da6aw9bvreynb.cloudfront.net has address 13.224.95.45
da6aw9bvreynb.cloudfront.net has address 13.224.95.22
```

Then, `whois` gives (among other information)

```
NetRange:       13.224.0.0 - 13.227.255.255
CIDR:           13.224.0.0/14
NetName:        AT-88-Z
NetHandle:      NET-13-224-0-0-1
Parent:         NET13 (NET-13-0-0-0-0)
NetType:        Direct Allocation
OriginAS:
Organization:   Amazon Technologies Inc. (AT-88-Z)
RegDate:        2018-07-11
Updated:        2018-07-11
Ref:            https://rdap.arin.net/registry/ip/13.224.0.0
```

Public geolocation services gives Zurich as the physical city of these servers.

3.5 Unauthenticated Metadata

As correctly mentioned [2, p.9], there is no authentication of Meta. It is encrypted with AES-CTR (hence by a simple XOR operation). So an active adversary can modify AEM without knowing the decryption key. The adversary can maliciously

- upgrade/downgrade the version number
- increase/decrease the claimed power level of the sender (henceforth make the receiver deduce a wrong distance).

It is not clear at this time what changing the version number could be used for because there is a single version at the moment. However, downgrade attacks are very common. In addition to this, **it is useful to make the receiver believe that the sender is closer than it is in order to inject false at-risk alerts**.

3.6 Replay Attacks

The receiver accepts a tolerance of 2h [2, p.9].[31] During this time, a captured Bluetooth signal can be replayed. **Replay attacks work during two hours and could be used to inject false at-risk alerts**.

[31] Presumably, this tolerance to 2h is set because the validity time of a beacon fluctuates in the 12–14min range and the receiver does not know exactly when this validity period is.

3.7 Enclave

The API does not reveal much to the app and it only reveals to the genuine app. The app is further forbidden to request a few privileges. It may be feasible to break those protections, as vulnerabilities of operating systems are often discovered and fixed. If we assume that the smartphone is secure and that the server is secure, the API-app-server chain does not reveal anything but

- the at-risk status of the user (which is the purpose of the app),
- the time and volume of connections to the server(s), and
- the reported (TEK, t) pairs (which is a non-necessary leakage from the protocol design choices).

The remaining attack vector in this case is the Bluetooth access to the phones and the access to the server.

Actually, having the API closed gives some security advantage. Assuming that Google-Apple did this job well and an be trusted, this provides a high level of security and defeats some of the attacks which have been proposed in the past. It is however very risky to count on this type of security as this has been shown to fail in the past.

3.8 Interoperability

Assuming the GAEN API is used in several countries, the traveling user must instruct its app to report to the server of recently-visited countries and also to retrieve reports in recently-visited countries. The GAEN additional terms of use [4] prohibits the app to have access to the location. Hence, the user must provide this information himself.

Adding other countries in the app implies adding more pinned certificates.

3.9 Lack of Report Granularity

Users who want to report selectively can only select which days to report. There is no smaller granularity. If a user wants to — say — remove some overnight contacts, he must anticipate and turn off tracing, or remove the entire days overlapping with the night. DP3T made recommendations to Google-Apple about this as detailed next.

3.10 The DP3T Suggestion

Since most of the system is implemented by the GAEN API, there is not much left from DP3T. We can observe that the latest version of the DP3T white paper [5] suggests to change the period of TEK to the minimal of 10 minutes (hence each TEK derives a unique RPI) *and* that TEK which were used to broadcast are immediately erases if no significant encounter was observed during this time. This change is proposed in order to mitigate the identification attacks. Such change should be done at the GAEN API level (i.e. the app has no control on this).

Such change would have several consequences.

- Storage may increase, because we suddenly have to store more TEK keys.
- Communication complexity may increase, because there would be more TEK to upload or to download.

- The granularity of the reporting functionality would improve.
- The risk for a diagnosed user to be identified would decrease because there are chances that the identifier kept by the adversary would be erased immediately.

Regarding the last point, a passive adversary who collects identifiers and wants to be sure that they are kept by the sender to be reported could run a more active attack: he could send a simulated broadcast with high power for long enough so that the target phone would think there is someone at proximity and keep TEK in memory. Such active attack is easy but could be detected by an inspector with a Bluetooth receiver.

3.11 Medical Device

It is well admitted that SwissCovid falls into the category of *medical devices* as defined by the regulating organizations. This is due to the fact that the purpose of SwissCovid is to inform users of their possible risk of infection. One consequence of being considered as a medical device is that it must comply with the regulation under the control of Swissmedic. This includes *Ordonnance sur les dispositifs médicaux* (ODim).[32] We list here a few consequences.

- (Art.14) The issuer of the system must take measures during the period of use to identify and prevent the risks caused by the device.
  The notion of *risk* is undefined. Privacy issues may be considered as a risk caused by SwissCovid.
- (Art.14) The issuer must operate a *product surveillance system* which collects and monitors information regarding the relevant experience of users and complaints by users, as well as published reports by specialized press about the system.
- (Art.15) Any professional who becomes aware of an incident must report, e.g. through their professional association. This must be done within less than 30 days. (less than two days if the incident is likely to repeat).
  The *incident* is defined (Art.3) as being *"an event associated with a medical device, resulting from a malfunction of the device, a change in significant characteristics, inadequacies in the labeling or instructions for use, and which has led, or could have led, to the death or a serious deterioration in the state of health of patients, users or third parties"*. Unless it causes serious psychological troubles, privacy issues may not fit into this definition. However, incorrect risk notification by SwissCovid could as the consequences of incorrectly put someone in quarantine or incorrectly not put someone in quarantine may have consequences on health.
- (Art.21) *"The claims with regard to the use, performance and efficacy of medical devices for direct dispensing to the general public or for direct use by the general public must be restricted to those contained in the product information only."*
  This implies that communication to encourage using SwissCovid must limit itself to information about using SwissCovid, the performances of SwissCovid, and the utility of SwissCovid.
- (Art.21) *"Misleading statements concerning the efficacy and performance of a medical device are prohibited."*
  We understand this as that incorrect information regarding the privacy risks for the users is prohibited.

[32] https://www.admin.ch/opc/fr/classified-compilation/19995459/

3.12 Consent

The user who installed the app has the opportunity to turn on/off tracing. (Turning on tracing and turning off Bluetooth has the effect of making the app raise notifications all the time.) By turning on tracing, the user receives a proper notification from the API saying that some data will be shared with the app.

The information about reporting in the app gives lot of information about COVID-19 but it is hard to find information about SwissCovid itself. Once we find it in a PDF document, we realize that the FAQ on SwissCovid provides no information about the privacy risks for the user. The official terms of the app about data protection (under section 2 "Collection and processing of personal data") gives a more complete but somehow contradictory information:[33]

> *The entire app system is designed to ensure that the app user is not identifiable. The processing of personal data is kept to a minimum. Data cannot be traced back by technical means to persons, locations or devices. What is collected is not location data, but merely encrypted data concerning proximity (contact) events. This is protected by technical means against misuse. The FOPH cannot draw any conclusions concerning app users. The app protects users' data in such a way that it cannot, at a distance, be connected to specific persons. Connection to a specific person cannot, however, be ruled out altogether. There is a certain likelihood that, when someone is notified of a possible exposure, their recollection of social contacts over recent days may allow them to deduce the identity of the infected individual. The notification contains the information that the user may potentially have been exposed to the coronavirus, the date on which this was last the case, and the behavioral recommendations of the FOPH. As a result of using the app, persons may thus potentially be identified.*
>
> *The app system has two components:*
>
> - *a proximity data management system, comprising software installed by users on their mobile phones and a back end (VA back end);*
> - *a code management system, comprising a web-based front end and a back end.*
>
> *Both of the back ends, as central servers, are under the direct control of the FOPH and are operated technically by the Federal Office of Information Technology, Systems and Telecommunication (FOITT). The code management front ends run on the authorized experts' systems.*
>
> *After coming into proximity (2 meters or less) with another mobile phone, the app stores the following data for a total of 21 days:*
>
> - *the identification codes broadcast by the other device*
> - *the signal strength*
> - *the date and the estimated duration of proximity.*
>
> *In the event of an infection being confirmed in a user, the following data is recorded in the code management system:*
>
> - *the release code*
> - *the date on which the first symptoms appeared, or — if the infected individual is asymptomatic — the date of testing*
> - *the time at which this data is to be destroyed.*
>
> *The VA back end contains a list with the following data:*

[33] https://www.bag.admin.ch/bag/en/home/krankheiten/ausbrueche-epidemien-pandemien/aktuelle-ausbrueche-epidemien/novel-cov/situation-schweiz-und-international/datenschutzerklaerung-nutzungsbedingungen.html

- *the private keys of infected users which were current in the period during which infection of other persons is likely to have occurred*
- *the date of each key.*

It seems that the only displayed message to warn the user who reports is[34]

> *Although no personal data relating to you is sent out, it may well be that someone remembers their encounter with you from the date.*

It would be more transparent and inspire more trust to add a clear warning such as

> *Anyone who captured one of your Bluetooth signals recently and who made the association with you can figure out that you reported.*

We believe that **providing unbiased and complete information about the risks for the user would inspire more trust and adoption than providing overly biased selling arguments**.

Triggering the report functionality on the app requires to enter the one-time access code at the very beginning. This is the covidcode. Verifying this code requires internet connection. This means the app is querying the server for verification before asking the GAEN API to prepare a report. If the server rejects the code, the process cannot continue. Hence, we could not see what choice the user has when reporting.

Another interesting question is whether the app is still scanning although tracing is on but Bluetooth is off. It is well known that turning off Bluetooth only turns off the sending functionality in Bluetooth. Bluetooth scanning can still be done by apps when Bluetooth is off. We could not verify if this is the case for the GAEN API too.

3.13 Accuracy of Information

The information provided by SwissCovid[35] suffers from inaccurate statements. For instance

> *Data cannot be traced back by technical means to persons, locations or devices. What is collected is not location data, but merely encrypted data concerning proximity (contact) events. This is protected by technical means against misuse.*

is not accurate. Assuming an adversary who has access only to the data collected on the server and to nothing else, the statement is correct. However, the *additional information* given by this data on the top of other data helps to identify a person, his device, and to locate them. Encryption is irrelevant here. There is no encrypted data at all. The cryptographic technique which is used is a simple key derivation mechanism which allows to compress a list of beacons to a single key in a report. It is also hard to admit that there are technical protection against misuse since data is freely available on the server.

In general, the term *encryption* is often misuse, as well as the term *anonymity.* If a SwissCovid device continuously broadcasts a random number, which anyone could store, and if this random number can later on be posted on a bulletin board, saying that this random number is *anonymous* is misleading. It is a *pseudonym* which can be associated to the person by anyone

[34] https://github.com/DP-3T/dp3t-app-android-ch/blob/develop/app/src/main/res/values/strings.xml String `inform_code_intro_text`

[35] https://www.bag.admin.ch/swisscovid-data-protection-statement-and-conditions-of-use

who has noticed the matching or who have received this information by someone who did. A pseudonym is considered as a *personal information* and is subject to regulation on data protection. Hence the difference between anonymity and pseudonymity has important legal consequences. The user has no control how the information about the pseudonym-identity matching spreads. This is a privacy concern which must be clearly explained.

The *Data Protection Statement* of SwissCovid under the section 7 *Rights of data subjects* states

> *With regard to your data, you have the right to information, rectification, erasure or disclosure. You also have the right to restrict or object to data processing. In addition, you have the right to withdraw your consent, without this affecting the lawfulness of processing based on consent before its withdrawal. These rights are applicable insofar as personal data is present; this is prevented to the greatest possible extent by the technical anonymization system underlying the app system. For this reason, it is not possible for the FOPH, for example, to provide information on the proximity events logged for a specific person or to correct this data in the system. The FOPH cannot connect this data to specific individuals.*

Given that by reporting, a user gives his pseudonyms to be posted, their right of erasure should be applicable. The fact that FOPH does not know how to connect a pseudonym to an identity does not mean that everybody else does not. However, this right for erasure is impossible to manage reliably. Currently, the user has no way to prove that a pseudonym is his. **It would be more transparent to ask the user for the explicit unrevocable consent to publish their pseudonym for a period of 21 days.**

SwissCovid also provides a FAQ list with approximative information[36]

> *4. How does the app ensure that my location is not disclosed via Bluetooth?*
> *The SwissCovid app does not use any satellite-based positioning technology. This means it is impossible to use the app to find where a person or a mobile phone is. Bluetooth Low Energy (BLE) simply allows one device to determine that it is close to another device. As Bluetooth has three levels of proximity recognition, the detection of contacts can be limited to the critical range of around two meters in open spaces.*

The point is that the Bluetooth beacon broadcast could be used by a close-by receiver who knows the location and reports it. Another point is that, independently of SwissCovid, the same Bluetooth technology is already used by Apple (find-my) and Google (nearby) to locate devices using Bluetooth. The non-use of GPS does not imply the impossibility to locate a phone.

> *7. Where are the servers for the SwissCovid-system located?*
> *The servers are located in Swiss government data centers in Switzerland and are hosted by the Federal Administration. The list of anonymous keys relating to persons who have been infected may however be passed on to third parties, which make them available to other users.*

This is a self contradiction. Either all servers are hosted by federal authorities in Switzerland or they are not. Why not explicitly saying that the server is hosted by Amazon?

[36] https://www.bag.admin.ch/dam/bag/en/dokumente/cc/kom/covid-19-faq-tracing-app.pdf.download.pdf/200513_FOPH_FAQ_SwissCovid_App.pdf

> *8. Could this app be used for other purposes?*
> *The app has been developed solely to contain the spread of the coronavirus and it will be discontinued as soon as it is no longer required for this purpose.*

This does not say that no third party could take advantage of people using the app for other purposes.

> *10. How long must someone be in my proximity before the SwissCovid app records a contact?*
> *People have to be less than two meters distance from each other for a short time. Only encrypted IDs, so-called checksums are exchanged via Bluetooth between the mobile phones. These distance measurements via Bluetooth will be continuously recalibrated during the test and pilot phases in order to improve their accuracy. The API announced by Google and Apple should further increase the accuracy of the measurement.*

There is neither encryption, nor ID, nor checksum. These are pseudorandom numbers derived from a random key by means of a one-way process. The term "encrypted ID" appears at several places in the document. It gives a false feeling of security (by using the term of encryption) and a false feeling of a risk (by using the term ID). The main problem behind this number is that it can be openly used several times: many times during 12–14 minutes then again if the user reports. Privacy issues come from correlating events about using this number. Claiming that the pseudorandom solely reveals nothing and cannot be connected to an identity is a misleading truth.

> *16. The app from the Federal Institutes of Technology in Zurich (ETHZ) and Lausanne (EPFL) works with its own protocol. This will then be replaced by the Apple or Google protocol. Will data security still be guaranteed?*
> *The Apple/Google application interface (API) is not a mobile phone app. It is a standard proposed by Apple/Google in order to achieve a more accurate estimate of the distance between two mobile phones via Bluetooth, and to reduce electricity consumption by Bluetooth Low Energy. Data security remains guaranteed — the Apple/Google standard is also based on the DP-3T concept drawn up by the Federal Institutes of Technology in Zurich and Lausanne.*

This does not say what guaranteed data security. This also hides that the app is now mostly a graphic interface. It would be more accurate to explicitly say that the protocol in SwissCovid was developed by the ETHs then modified and implemented by Apple/Google in phones.

3.14 Proof of At-Risk Status

One suggestion from the Swiss SSHC-S committee is that someone who is notified at risk and decides to stay home for a few days would not be paid by his employer but would have subsidies from authorities for being a responsible resident.

One question is how to prove that he received the at-risk notification to claim for subsidies. The app does not offer any such proof at the moment. A screen capture could easily be forged.

If such a proof was added in future version, an interesting new attack model would be the one of someone trying to maliciously make his phone believe that he is at risk to be infected, in order to have extra paid vacations at home.

3.15 Consent and Liability

Anyone who contaminates another one by being careless may be subject to criminal liability. However, the at-risk notification is not clearly meant to induce any obligation. **Liability in the case of ignoring the notification is unclear**. (Specially, if a proof of at-risk status was implemented, the app could provide evidence of lack of care.) It is of utmost importance to make it clear before asking the consent to users.

4 Threats

4.1 Effectiveness of Existing Attacks

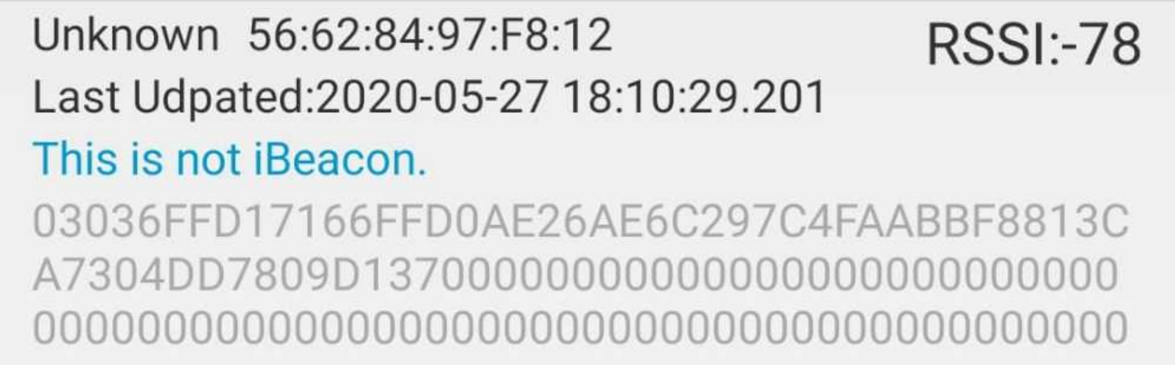
Some previous report [7] presents possible attacks against the DP3T protocol. Some of them are discussed in a specific section. We review here some other ones.

- Linking beacons derived from the same key (i.e. by the same phone during the same day) is possible after the user reports.
- Reading the used or received beacons by coercion is made harder by making it protected by the API. (We did not investigate how hard.)

We discuss in a separate section the attacks which identify reporting users and the attacks which create false alerts.

4.2 Inherent Threats Related to Bluetooth

Watching beacons. Currently, with existing phones we can use Bluetooth apps to watch the broadcasts by other phones which use SwissCovid. For instance, using the app iBeaconDetector,[37] we can clearly see what follows:



```
Unknown  56:62:84:97:F8:12                    RSSI:-78
Last Udpated:2020-05-27 18:10:29.201
This is not iBeacon.
03036FFD17166FFD0AE26AE6C297C4FAABBF8813C
A7304DD7809D13700000000000000000000000000
000000000000000000000000000000000000000000
```

We clearly see the full message with `0xFD6F` number (in little endian) corresponding to the Exposure Notification service. It was broadcasted by an unknown device with MAC address `56:62:84:97:F8:12`. This address is random and changes regularly.

On the phone we tested, the beacon is repeated nearly 3.5 times per second (i.e. every 285ms) until it changes. The exact rotation period fluctuates a lot between 12 and 14 minutes.

It may be possible in the future that Androids and iphones filter beacons with `0xFD6F` service number so that no Bluetooth app can see them nor send any, but any other device can be used to do so. We tested it with an (old) laptop and also with the ubertooth USB dongle (about 125 CHF).[38] The old laptop easily catches the beacons at a 4m distance through a

[37] https://play.google.com/store/apps/details?id=youten.redo.ble.ibeacondetector
https://f-droid.org/en/packages/youten.redo.ble.ibeacondetector/

[38] https://greatscottgadgets.com/ubertoothone/

wall. The ubertooth with a small 2.4GHz antenna could easily capture SwissCovid beacons through two walls at a 15m distance. Experts further say that capturing at several 100m distance is easy.

We used standard Bluez system GNU/Linux commands to receive and send BLE beacons.

We also demonstrated the (unsurprising) possibility to capture, replay and/or forge advertising broadcasts with cheap Bluetooth USB dongle, integrated Bluetooth in laptops and cheap embedded devices such as Raspberry Pi/Zero W/etc. On Fig. 2 we show a picture of a Raspberry Pi/Zero W and of the ubertooth dongle.

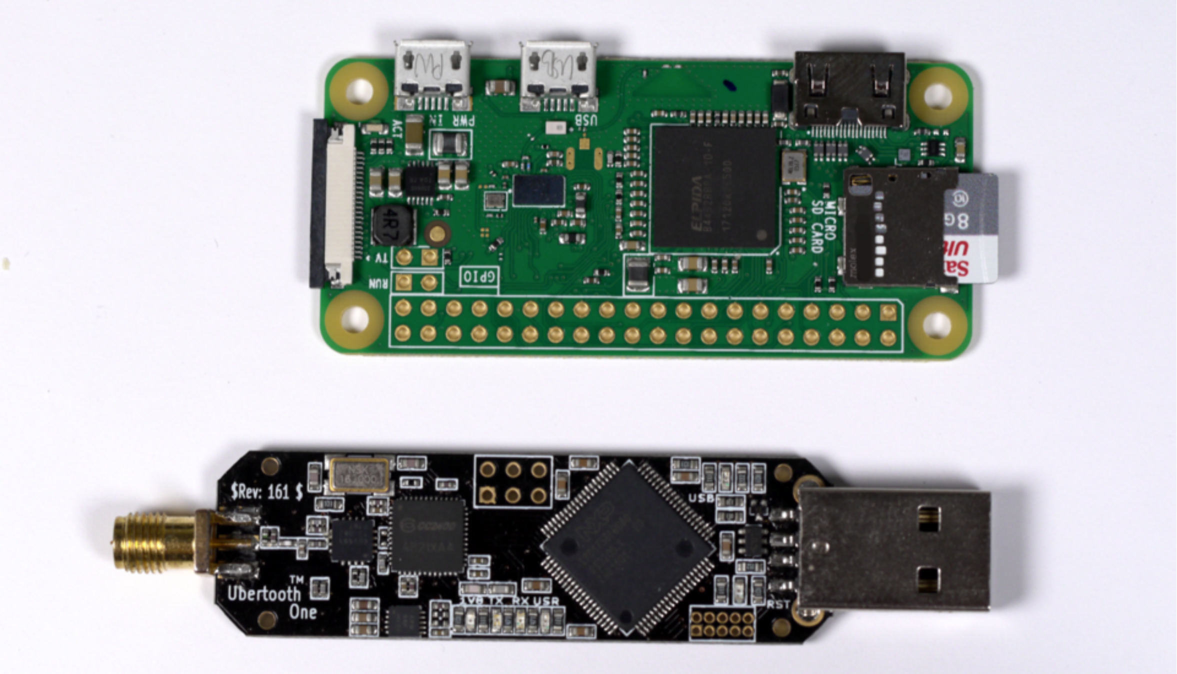


Fig. 2. Raspberry Pi/Zero W (up) and ubertooth dongle (down)

We could see that every time the phone captures a new beacon or receive a beacon that we replayed, GAEN modifies an internal file and the length of this file enlarges.

We further stress that we wasted a lot of time trying to understand the behavior of a given phone without seeing anything working as expected. Trying on another phone succeeded. Then, we were able to understand that the first one, although able to broadcast beacons normally, was silently making the Bluetooth stack crash every time it was receiving a beacon. Hence, **it is likely that some phones will look like working normally while they will be unable to spot if the user encountered a person at risk** because they receive nothing from other phones.

Tracking phones. Each beacon is repeated 3.5 times per second during 12–14 minutes. Hence, each beacon is repeated roughly 2 300 times. A passive Bluetooth sensor sees fragments of repeated beacons. Two overlapping fragments necessarily come from two different active SwissCovid phones. Therefore, **a passive Bluetooth sensor can easily keep track on how many active SwissCovid phones are present in its covering cell in real time**. For instance, we can monitor how many people (more precisely, a lower bound on the number of phones) are present in the neighbor apartment at every moment of the day.

A second step is to figure out if phones at two different times are the same. The above observation allows to infer fragments from different phones. The MAC address rotates as well. Interestingly, the MAC address and the beacon do not always rotate synchronously. This means it is trivial to link two consecutive different beacons as coming from the same phone. Even though rotation was done synchronously, since two consecutive fragments are highly likely to come from the same device, it is easy to infer fragments from identical phones. This way, an adversary with a single sensor can get a partial view on the equality and inequality relations between fragments. This information can be pooled with several sensors. Actually, **by disseminating sensors in a building, we can locate a phone in the intersection of cells and track moving phones from cell to cell**. The attack is passive.

If, occasionally, a beacon belongs to a reporting user, the reported key allows to link all beacons that this user sent during the day. Hence, some more equality relations can be deduced.

We can make a cheap sensor (less than 10 CHF) with a Raspberry Pi0. It can catch beacons sent in a cell of about 10m of radius. These cheap sensors may be exploited to create a large network of surveillance for both active and passive attacks. (Active attacks are discussed next.)

Identifying phones. Letting Bluetooth on also allows to do active attacks to get a fixed MAC address and the name of the phone. Using standard BR/EDR Bluetooth, in addition to the SwissCovid beacon, we sometimes receive the fixed MAC address and the identity of the phone, depending on which apps are installed on the phone. We did not investigate further but **it is likely that a good fraction of SwissCovid users are identifiable by Bluetooth**.

Locating phones. Some expensive devices (direction finding) allow to locate a phone which is broadcasting with Bluetooth. Hence, **we can see from far away the movement of phones**. This surveillance system requires no access to any SwissCovid component. This technology is accessible by law enforcement. (We did not test it.)

4.3 Inherent Threats Related to the Protocol

Identifying reporting users. The risk to identify users who reported have been documented several times [7]. We recall here the possible scenarios.

- *Paparazzi attack.* A paparazzi with appropriate equipment captures the beacon of celebrities then checks on the public server if celebrities reported.
- *Nerd attack.* A malicious person develop his own tool to scan beacons and record additional information about the sender. Checking on the public server allows to identify reported people. This can be done with regular phones or other portable equipment. If beacons become filtered by phones, this may not work any more on phones.
- *Militia attack.* A group of people use the previous attack and share their information.
- *Attack by organization.* An organization such as a hotel, a shop, or a company, captures a Bluetooth beacon at the same time it identifies its visitor, customer, or employee, then learns on the public server if this person was diagnosed.
- *Surveillance systems.* A system of surveillance (e.g. video-surveillance) is coupled with a Bluetooth sniffer. Beacons which are reported on the server can be linked to a record from the surveillance system.

We discuss attacks with malicious app or phone in a separate section. Hence, **identifying reporting people is essentially easy for anyone/anything who has seen those people before**.

We could query the download server and retrieve the (currently empty) list. Access to the download server is actually open by design. Hence, those attacks are easy to make. They only concern people who report, but those people deserve to be aware of it.

Creating false encounters. In the so-called lazy student attack, the student who wants his exam to be cancelled bombards his professor with beacons which are replayed from people who are very likely to be diagnosed soon. Those beacons can be captured, for instance, in hospitals, or places where people make COVID-19 infection tests. Replaying a beacon is feasible during 2h. Additionally, if the student is more than 2m away from his professor, he can modify the AEM value to make his phone believe he is really close to him. The student can use his own phone to replay the beacon with an appropriate app (as long as those beacons are not filtered). He can also use other portable equipments. There can be a black market of at-risk beacons to replay [7].

The GAEN API has already mitigated this attack by limiting the validity period of the beacon but it is still possible during 2h. Hence, **making someone's phone believe that it met the sender of a given beacon is easy during 2h**.

Another attack can be done by a group of activists/terrorists who would use their own broadcasting tool mimicking the GAEN API, and synchronize on using the exact same beacons. They would go around the city and meet as many people as possible until one member of the group becomes sick and reports. Then, a wide proportion of the population will receive an alert and go to self-quarantine. Hence, **making many phones believe that they met the sender of a given beacon is easy**.

The SSHC-S committee of the Swiss States Council recommended[39] to financially help users who would self-isolate after their app notifies a risk of infection and before any sickness symptom appears. Some malicious users may abuse this to take free vacations. Hence, **a malicious user to take advantage of making his own phone believe that it met the sender of a given beacon.**

Creating false at-risk notification. The previous attacks creating a false encounter are useful when they create a false at-risk notification.

A more direct way to create false at-risk notifications could be to directly upload on the server the keys which generate an encountered beacon. We note that the enclave with the GAEN API does not allow to report a sent beacon without being diagnosed. This means that to do such attack, either we should break the enclave (we did not look at how hard this would be) or we should simulate the GAEN API with another implementation and steal a valid covidcode.

The attack based on corrupting the server are discussed in a separate section. We wonder here how to get the credentials to upload without being diagnosed.

Uploading a report is done by a user who holds a covidcode which is valid for 24 hours. One threat is that people sell their code which would maliciously be used. The FAQ list of SwissCovid[40]

[39] https://www.parlament.ch/press-releases/Pages/mm-sgk-s-2020-05-26.aspx

[40] https://www.bag.admin.ch/dam/bag/en/dokumente/cc/kom/covid-19-faq-tracing-app.pdf.download.pdf/200513_FOPH_FAQ_SwissCovid_App.pdf

> *25. If I get infected, am I obliged to enter the covidcode into the SwissCovid app or can I decide to keep the news to myself?*
> *The covidcode remains valid for 24 hours after its generation by the contact management staff. Whether and when you enter it into the SwissCovid app is your decision. You are under no legal obligation to do so. Whether you enter the code or not, you will remain anonymous, i.e. no one can find out subsequently whether you entered the covidcode or not.*

This implies that someone who is given this code could give it to someone else without being noticed. Hence, **diagnosed people could be corrupted to sell their covidcode** to malicious services offering to inject false alerts.

4.4 Threats Related to Malicious Apps

Bluetooth surveillance app. A malicious app which has access to location, when running in the foreground, can scan Bluetooth beacons, isolate those with the `0xFD6F` SwissCovid identifier, and report them to the server of the app together with the location and time of reception. The server would hence collect records of beacons with location and time.

If this app is popular and often running in the foreground, this can implement an easy distributed Bluetooth-based surveillance system. Such surveillance system collects records and recognizes some which correspond to reported users by checking the server. **It would see where and when a contagious person have been.**

If the app has no access to location (for instance because the user turned this permission off on purpose), it could be the case in a crowded place such as a terrace or public transport that some other phone with location on can see the same beacon and report with the same app. **Crossing information allows the server to locate the user who turned localization off**. Relying on another device which can locate itself is actually the basic Bluetooth-based technique for locating a device when location is off. Systems such as nearby and find-my are based on this method.

Simple identifying app. If the previous malicious app has access to some form of identity of the phone (such as the phone number, account information for the app, address book), then such app can be used to identify people who have been in contact with a reported diagnosed user. **Such app could learn if the phone holder is at risk even though SwissCovid did not take this encounter as a risk or was not active at the time of encounter**.

Malicious social app. In general, the app of the sender and the app of the receiver may disclose different types of information which, put together, would reveal private information. For instance, if the two apps reveal the identity, location, and received beacons, the server can match beacons with identities and improve the surveillance system.

Apps such as Facebook, Whatsapp, or Twitter are often in the foreground. The first two already have a good idea about the social graph and make identification easier.

4.5 Threats Related to Malicious Network

A malicious network which routes a connection request from a phone to the code management system and who knows the identity of the user will deduce that the user entered a covidcode. This can be done by the telecommunication operator, by a Wi-Fi hotspot, or by other routers.

Most of those connections may be fake. However, there may still be ways for the network to figure out if the request is fake or not by side channel information. For instance, the length of queries and responses as well as the time when they are made and the delay of the response could leak. The request by the app following the response from the server may also leak. Since this process is not documented and since we had no non-fake covidcode to try, we could not analyze the possible leakage. Therefore, **the network may identify the reporting device after positive diagnosis**.

4.6 Threats Related to Malicious Phones

We treat here threats coming from a malicious operating system, API, or hardware. Those entities can see everything what the phone is doing and report it to a third party. Actually, they could run a contact tracing by themselves even though there is no SwissCovid app installed. However, doing this is not very discrete because Bluetooth broadcasts are visible by any Bluetooth receiver. Using SwissCovid does not change this threat but makes it easier to be unnoticed.

Location based on Bluetooth was already in use by Apple (in the find-my application) and Google (in the nearby library). Those services enable locating a phone even though localization is off. SwissCovid could help malicious phones to continue to trace people and their encounters by using the same technology. Actually, the separation between GAEN and nearby is not clear within the GMS library. It is possible that SwissCovid beacons help GMS to locate the phone. Likewise, SwissCovid may help find-my to find a device.

4.7 Threats Related to the Security of the Server

False at-risk notification injection. A malicious server can simulate reports by himself, hence performs the attacks described in a previous section. Given that the server is hosted by a third party **the current SwissCovid system may be giving power to this third party to inject false alerts on selected phones**. As this part is not documented, we could not check if this attack is possible.

Identification of reporting users. In general, servers can see an IP address and a user agent, which reveals something about the querying phone. The code management system and the upload server keeps connection logs of those information which may reveal something about the diagnosed person. The IP address allows to identify from which area this person is reporting. Cross-relating the IP address and time with other network logs may allow to fully identify the person.

5 Conclusion

We have shown that SwissCovid creates critical security and privacy threats. Shall they be reduced or not, we believe that they must be communicated. More importantly,

- the available information is insufficient,
- there are misconceptions about anonymity and open source,
- there seems to be no room for public security test in the agenda,
- and the developers of SwissCovid are bound to Google-Apple decisions.

References

1. Exposure Notification — Bluetooth Specification. Version 1.2. Apple and Google. https://blog.google/documents/70/Exposure_Notification_-_Bluetooth_Specification_v1.2.2.pdf
2. Exposure Notification — Cryptography Specification. Version 1.2. Apple and Google. https://blog.google/documents/69/Exposure_Notification_-_Cryptography_Specification_v1.2.1.pdf
3. Exposure Notification — Android API Documentation. Version 1.3.2. Apple and Google. https://www.google.com/covid19/exposurenotifications/pdfs/Android-Exposure-Notification-API-documentation-v1.3.2.pdf
4. Google COVID-19 Exposure Notifications Service Additional Terms. Version: May 4, 2020. Google. https://blog.google/documents/72/Exposure_Notifications_Service_Additional_Terms.pdf
5. Carmela Troncoso, Mathias Payer, Jean-Pierre Hubaux, Marcel Salathé, James Larus, Edouard Bugnion, Wouter Lueks, Theresa Stadler, Apostolos Pyrgelis, Daniele Antonioli, Ludovic Barman, Sylvain Chatel, Kenneth Paterson, Srdjan Capkun, David Basin, Jan Beutel, Dennis Jackson, Marc Roeschlin, Patrick Leu, Bart Preneel, Nigel Smart, Aysajan Abidin, Seda Guerses, Michael Veale, Cas Cremers, Michael Backes, Nils Ole Tippenhauer, Reuben Binns, Ciro Cattuto, Alain Barrat, Dario Fiore, Manuel Barbosa, Rui Oliveira, José Pereira. Decentralized Privacy-Preserving Proximity Tracing. Version: 25th May 2020. DP-3T Project. https://github.com/DP-3T/documents
   Also: Preprint arXiv:2005.12273 [cs.CR], 2020. https://arxiv.org/abs/2005.12273
6. DP3T Team. Decentralized Proximity Tracing — Interoperability Specification. Version 0.1. https://github.com/DP-3T/documents
7. Serge Vaudenay. Centralized or Decentralized? The Contact Tracing Dilemna. Cryptology ePrint Archive: Report 2020/531. IACR. http://eprint.iacr.org/2020/531

A Addendum, June 16 2020

We submitted our report on June 5 which was subject to a responsible disclosure rule in the frame of the test of the pilot system.[41] The administration has made our summary public while the report was under analysis and not public. Nevertheless, it has been cited and criticized in Le Temps (11.6.2020)[42] by EPFL:

> *Certaines affirmations contenues dans ce rapport sont jugées incorrectes par l'EPFL. Ainsi, il y est écrit que "le serveur est hébergé par Amazon", ce qui est clairement démenti par la vice-présidence de l'école.*[43]

Besides, one author of this report was suggested by EPFL to correct some "incorrect" statements in the (unpublished at that time) report. For transparency, we decided to let the report as is and to augment it with this addendum. We try to clarify some statements below on which we are in disagreement.

A.1 On Amazon-Hosted Servers

The app sends queries to the TLS server `www.pt.bfs.admin.ch`. As explained in Section 3.4, this name is resolved in Switzerland by a DNS as an alias for a Cloudfront infrastructure with several IP addresses including 13.224.102.111. This address has a reversed name

```
server-13-224-102-111.zrh50.r.cloudfront.net
```

which suggests it is an Amazon *server* located in Zürich. We stress that this machine is called *server* by its name. The same experiment from other places gives other machines around the world. For instance, from France, `www.pt.bfs.admin.ch` resolves in an Amazon server located in Amsterdam. Probably, it resolves in other Amazon servers all around the world from other countries.

The Amazon servers act as CDNs, which may imply additional security risks. Since there is no public description of this part of the infrastructure, the security of CDNs has not been evaluated.

The TLS query from the app is answered with a certificate belonging to *Bundesamt für Informatik und Telekommunikation BIT* which is signed by QuoVadis in Bermuda (as we can read in the certificate). It is quite common to give the secret key of this certificate to the servers in the CDN architecture, but it depends on the adopted architecture. If this is the case, it means that there are many Amazon servers all around the world which share this TLS secret. This could imply vulnerabilities. However, the observation about CDN and extra signature that we already made in Section 2.6 of this report is supporting this assumption and even suggests that the risk of sharing this key may have been addressed. Hosting servers in a third party may have important impact on which we could not properly investigate, due to a lack of specification and to the fact that the information was actually hidden.

How about EPFL denying that any server is hosted by Amazon? It seems that there is a misunderstanding about the term "server". Since the beginning of the project, DP3T mentions one unique server which now turns into being a constellation of machines, some of them being

[41] https://www.melani.admin.ch/melani/en/home/public-security-test/scope_and_rules.html

[42] https://www.letemps.ch/economie/swisscovid-feu-critiques-quatre-questions-comprendre

[43] *Some statements in this report are considered to be incorrect by EPFL. For instance, it says "the server is hosted by Amazon" which is clearly denied by the vice-president of the school.*

called “servers” and some others not. Machines which are called servers are claimed to be hosted by the Federal administration. Some others which everyone but SwissCovid designers call servers (like the one whose reversed name includes `server` and which could be the actual TLS server) are hosted by Amazon. Denying that servers are hosted by Amazon is just playing on words. It is irrelevant.

Nevertheless, we admit that having servers hosted by Amazon or not is an unnecessary controversy from a technical viewpoint. The overall architecture is not documented yet but does not seem to suffer from an evident security problem from what we could see. We hope other groups have carefully investigated the impact of this architecture.

A.2 On GAEN Having no Source Code

The SwissCovid components are required to have publicly available source codes. GAEN, which is maintained by Apple and Google has no available source code.

It seems that there is a misunderstanding about the term ”component”. Actually, the components are legally defined in the ordinance about the pilot test by excluding GAEN, as mentioned in Section 3.1. Critics to our observation sometimes take GAEN as part of the *phone*, which takes care of the Bluetooth input/output communication, and sometimes as part of the *operating system*, which regulates accesses and rights among all apps and services. One comment from the Federal Data Protection and Information Commissioner (FDPIC) states[44]

> *The FDPIC is aware of the widespread criticism of Google and Apple's failure to disclose the API (application programming interface) for the SwissCovid app. The situation, however, is not new. The data protection impact assessment of 1 May 2020 and the report from the National Cyber Security Centre (NCSC) of 28 May 2020, which is published today, also make reference to this lack of disclosure.*
> *The SwissCovid app is based on globally standardised interfaces and their underlying operating systems. The source code for the operating systems and the interfaces is partially available in some cases or not available at all. This issue is known and is not specific to the SwissCovid app.*

The DP3T protocol is described in a White Paper [5] with ways to generate ephemeral identifiers, schedule their usage, collect received ones, compare with some reported on the server, and identify risky encounters. The prestandard version (which is not based on GAEN) was implementing all these. For instance, the source code

> https://github.com/DP-3T/dp3t-sdk-android/blob/prestandard/dp3t-sdk/sdk/src/main/java/org/dpppt/android/sdk/internal/crypto/CryptoModule.java

implements the `EphID` generation, rotation, and comparison. In the latest version, all of these operations are implemented in GAEN. There is no available source code showing how these operations are done, no way to verify it is properly done, no way to modify it if need be. This part of SwissCovid is up to Apple and Google to change in the future. Actually, we identified a specific protocol problem related to GAEN in Section 3.5 of this report which Federal offices have no way to fix, may it be necessary or not. The app is the interface between GAEN, the

[44] Update Proximity Tracing App, 12.6.2020 on https://www.edoeb.admin.ch/edoeb/en/home/latest-news/aktuell_news.html

server(s), and the user. This is why we chose to write our report by including GAEN as a major component of the SwissCovid system. We hereby maintain GAEN must be considered as such.

GAEN is neither the part of the *phone* which regulates Bluetooth communications nor a part of the operating system. On Android, GAEN is part of the Google Play Services which regulate the Google-specific services (like Maps, Cast, Fit, Analytics, Drive, Location, etc). Having Google Play Services installed is virtually necessary to use Google apps such as Gmail, Maps, or Play Store. However, Android phones can be "de-Googled" and work without Google Play Services, yet use Bluetooth. This proves GAEN is neither part of communication drivers nor part of the operating system, contrarily to the common excuse for the non disclosure of GAEN which is repeatedly relayed by the press. Actually, the prestandard version does not depend on GAEN.[45] We verified it is working well on a phone which has no Google Play Services, still has the Android operating system.

We are convinced that the legal definition of *components* in an ordinance is a trick to bypass the law on the source code availability. In RS 818.101.25 Art.5 (*Ordonnance COVID-19 essai pilote traçage de proximité*), the Federal Council is playing with redefining what is common sense in order to fit the law. **We urge independent legal experts to make a statement on this controversy to determine whether GAEN should be considered as a component of SwissCovid and therefore be subject to the law requiring an available source code.**

A.3 On the App Being Open Source

The SwissCovid app (excluding GAEN) has available source codes. This source code is under the Mozilla MPL2 license which allows developers to mix open source code with proprietary code. This type of license was preferred to purest open source licenses such as GPL2 precisely to allow using the GAEN library which has no available source code.

To be qualified as open source, anyone should be able to take the source code, modify it, compile it, install it, and run it. In Section 3.2, we said we cannot run it, which is incorrect. We actually can. However, running the obtained app (even without modifying the code) terminates on the following error

Failed to enable COVID-19 Notifications (Unauthorized package signature)

Hence, it runs but it does not work. To acquire the authorization to make the app work requires to sign a non-disclosure agreement.[46] We do not find it compatible with the notion of open source. We invite legal experts to make a statement on this point.

However, the adopted law does not require SwissCovid to be open source *per se*, so this question is of purely academic interest. The law requires to have publicly available source codes and specifications, and to be able to verify that the installed app was built from the available source code. We acknowledge that source codes are available (except for GAEN, as previously mentioned). Complete specifications are still missing. We have been able to see that verifiability was on a way to be solved.

[45] https://github.com/DP-3T/dp3t-sdk-android/tree/prestandard

[46] We also tried to sign the app with the available keystore and the indicated password, which evidently failed. https://github.com/DP-3T/dp3t-app-android-ch/blob/testversion1-2020-04-29/app/build.gradle

A.4 On Reported Attacks

None of the attacks we discussed are essentially new or surprising. We already published some of them on April 8, 2020.[47] SwissCovid designers claim that they have already documented them. The documents we found are however always minimizing the impact or defending that other systems would be worse. For instance, the response to the April 8 analysis[48] answers as follows.

- On Bluetooth vulnerabilities, the document points to another document (GR5, PSRE) which sorts the case in 6 lines.
- Deanonymizing known reported users is said to be inherent to all contact tracing methods, without saying that such attack is much easier in SwissCovid.
- Replay attacks is kind-of acknowledged as a weakness (we nevertheless observe that it is still possible 2 hours after the epoch of the replayed beacon).

We have been aware that several attacks have already been empirically tested and reported on June 10.[49]

Our main point is that volunteer users should be aware of those attacks. They may be considered as minor for most of them but of critical importance of some. So far, the documentation provided to the user is silent on this.

[47] Serge Vaudenay.
Analysis of DP3T.
Cryptology ePrint Archive: Report 2020/399. IACR.
http://eprint.iacr.org/2020/399

[48] https://github.com/DP-3T/documents/blob/master/Security%20analysis/Response%20to%20'Analysis%20of%20DP3T'.pdf

[49] Lars Baumgärtner, Alexandra Dmitrienko, Bernd Freisleben, Alexander Gruler, Jonas Höchst, Joshua Kühlberg, Mira Mezini, Markus Miettinen, Anel Muhamedagic, Thien Duc Nguyen, Alvar Penning, Dermot Frederik Pustelnik, Filipp Roos, Ahmad-Reza Sadeghi, Michael Schwarz, Christian Uhl.
Mind the GAP: Security & Privacy Risks of Contact Tracing Apps.
Preprint arXiv:2006.05914 [cs.CR], 2020.
https://arxiv.org/abs/2006.05914

B Errata (State: June 16, 2020)

- Abstract. “The server is hosted by Amazon” should be changed to “Some servers are hosted by Amazon”.
- Section 3.2, p.10. “no one can compile and execute SwissCovid” should be changed to “if anyone compiles SwissCovid, executing it will fail on an error”.
- Section 3.3, p.11. “[Google Play Services] which requires the user to register a gmail account” should be changed to “Installation from the Google Play Store requires the user to register a gmail account”.