Digital Contact Tracing Solutions: Promises, Pitfalls and Challenges

Thien Duc Nguyen[1], Markus Miettinen[1], Alexandra Dmitrienko[2], Ahmad-Reza Sadeghi[1], and Ivan Visconti[3]

[1]*Technical University of Darmstadt, Germany - {ducthien.nguyen, markus.miettinen, ahmad.sadeghi}@trust.tu-darmstadt.de*

[2]*JMU Würzburg, Germany - alexandra.dmitrienko@uni-wuerzburg.de*

[3]*University of Salerno, Italy - visconti@unisa.it*

***Abstract*—The COVID-19 pandemic has caused many countries to deploy novel digital contact tracing (DCT) systems to boost the efficiency of manual tracing of infection chains. In this paper, we systematically analyze DCT solutions and categorize them based on their design approaches and architectures. We analyze them with regard to effectiveness, security, privacy and ethical aspects and compare prominent solutions with regard to these requirements. In particular, we discuss shortcomings of the Google and Apple Exposure Notification API (GAEN) that is currently widely adopted all over the world. We find that the security and privacy of GAEN has considerable deficiencies as it can be compromised by severe large-scale attacks.**

We also discuss other proposed approaches for contact tracing, including our proposal TRACECORONA, that are based on Diffie-Hellman (DH) key exchange and aim at tackling shortcomings of existing solutions. Our extensive analysis shows that TRACECORONA fulfills the above security requirements better than deployed state-of-the-art approaches. We have implemented TRACECORONA and its beta test version has been used by more than 2000 users without any major functional problems[1], demonstrating that there are no technical reasons requiring to make compromises with regard to the requirements of DCT approaches.

***Index Terms*—digital contact tracing, privacy, security**

I. INTRODUCTION

The pandemic caused by the SARS-CoV-2 corona virus has still the world in its grip since it was officially announced by the World Health Organization (WTO) on March 11, 2020. At the time of writing, we have been witnessing the surge of several infection waves all around the world. Reliable and efficient contact tracing for containing the spread of infections has therefore become more important than ever. In many countries, digital contact tracing apps on smartphones have already been rolled out to support manual contact tracing with the hope of significantly improving its effectiveness in breaking infection chains and preventing the virus from spreading further. Particularly automatic notifications can be faster and can reach at-risk individuals in situations related to random encounters between strangers as they occur, e.g., in public transportation, shops, and other indoor activities. There also exist contrary opinions[2] on the generic usefulness of digital contact tracing. In this paper, however, we do not seek to resolve this controversy, but focus instead on analyzing how theoretical results of epidemiologists (e.g., [1]) are taken into account in current proposals for identifying at-risk contacts in the presence of technological errors, data pollution attacks and privacy and ethics regulations. Initially we analyze deployed solutions, as many countries are currently actively employing them and millions of users are affected by such systems. Regardless of the potential usefulness of digital contact tracing or a lack thereof, contact tracing apps have become a reality in many countries. At the time of writing, 49 countries around the world (including, e.g., most European countries, Australia, China, Singapore) and 27 states in the USA have deployed contact tracing apps [3]. Many of these systems in use today were designed, implemented and rolled out in great haste with the goal of containing the spread of the pandemic as quickly as possible. It is therefore ever more important to take a step back and try to obtain a critical view of the benefits and disadvantages of individual approaches.

In this context, *effectiveness*, *security*, *privacy* and *ethics* are key aspects that need to be considered thoroughly: (i) the system should be *effective*, i.e., able to provide acceptable detection accuracy (high true positive and low false positive rate), (ii) it should be *secure* so that malicious adversaries cannot manipulate the system to trigger false alarms, (iii) it should protect *privacy* to increase users' trust in the DCT system, and (iv) it should consider ethical aspects as it should be transparent and based on voluntary use. Ensuring all above properties is necessary to achieve high adoption rates to then significantly contain the spread of the virus. Users need to be confident that the system is reliable in identifying at-risk contacts, and minimizes the use of users' data and provides strong security and privacy measures to protect them from malicious institutional operators as well as external attackers who may seek to track or to damage users. Otherwise, users will not be willing to use contact tracing apps, negatively impacting their adoption rate that would be crucial for their effectiveness in practice (ideally higher than 60%) [2].

While the first countries (predominantly in Asia) that deployed tracing apps adopted centralized approaches, and extensively collected sensitive user information (e.g., names, addresses, mobile phone numbers, location), a widespread and

[1]https://tracecorona.net/download-tracecorona/

[2]Contact Tracing in the Real World, https://tinyurl.com/2p9765ra

[3]MIT Covid Tracing Tracker, https://tinyurl.com/3ey44r5c

heated debate on user privacy broke out in Europe and the USA[4]. In this turmoil of evolving contact tracing approaches, somewhat surprisingly, Google and Apple established an unprecedented collaboration and provided their special application programming interface for decentralized contact tracing called *Exposure Notification API (GAEN)* [3] which they rapidly integrated into their mobile operating systems. Google and Apple give in each country access to this interface only to one organization that is authorized by the local government. GAEN runs an almost complete contact tracing solution as a part of the underlying mobile operating systems, so that the role of national organizations is reduced to developing a user interface to GAEN through a smartphone app and providing the backend server infrastructure required for acquiring and distributing information about at-risk contacts. Further, although Apple and Google initially promised not to get directly involved in contact tracing by developing their own backend server and app, later they did so by providing the GAEN Express solution that is used in several US states, e.g., Maryland and Utah [5]. Unfortunately, as we will discuss in more detail in Sect. V, it is known that existing rolled out Digital Contact Tracing (DCT) systems exhibit a number of important security and privacy risks [4], [5], [6], [7].

The controversial debate on tracing apps goes significantly beyond purely technical concepts and includes also questions related to effectiveness, socio-political impact, surveillance capitalism, and technological sovereignty (see, e.g., [8], [9]). In this paper, we contribute to this debate by providing a systematization of the development, deployment, effectiveness, security, and privacy of contact tracing apps aiming to cover these relevant aspects. Further, in order to tackle the shortcomings of existing approaches, we introduce a novel user-controlled privacy-preserving contact tracing system called TRACECORONA. It leverages a robust privacy architecture based on Diffie-Hellman key exchange to provide a level of security and anonymity unparalleled by any of the other systems proposed so far. It also improves the effectiveness and accuracy of the overall system and its resilience to misuse through the ability to *verify* all critical encounters.

In particular, we provide following contributions:

- We introduce a categorization of the requirements on Digital Contact Tracing (DCT) systems in four dimensions, namely: effectiveness, privacy, and security as well as ethical considerations (Sect. III).
- We review existing contact tracing approaches by analyzing prominent schemes with respect to these aforementioned aspects and construct a generic framework for categorizing them (Sect. IV). We point out that the advantages and disadvantages of a DCT system should not be prejudged by certain design choices. For example, contrary to commonly voiced opinions, one cannot claim that decentralized approaches satisfy "privacy by design". We argue that more emphasis should be given to data quality since decentralized solutions are limited in their capabilities to mitigate errors due to technological limitations and in preventing data pollution attacks by third parties.
- We extensively analyze to what degree prominent and widely deployed DCT approaches fulfill these requirements (Sect. V) providing a comprehensive and insightful view on each type of these approaches. In contrast to the vast endorsement received from many governments and scientists, we show that GAEN performs poorly in satisfying almost all requirements. When taking into account various costs associated with privacy leaks and failures due to data pollution, the justification for an approach like GAEN becomes highly questionable.
- Based on our analyses and findings, we discuss other schemes and technologies that could fulfill those requirements (Sect. VI). In particular, we elaborate on approaches based on Diffie-Hellman (DH) key exchange that have not yet been deployed in practice. Our analysis shows that DH-based systems provide better security and privacy guarantees than GAEN while maintaining comparable effectiveness (Sect. VI). We propose a novel distributed contact tracing system, TRACECORONA, providing strong security and privacy guarantees. In contrast to almost all existing approaches that are based on exchanging pseudonymous proximity identifiers, our approach leverages advanced cryptographic algorithms to establish and verify encounter tokens that are unique to each encounter between two users. Further, we propose various use cases and deployments of TRACECORONA including a hybrid approach (cf. Sect. VI-D). We implemented, deployed, and published TRACECORONA for beta user test. More than 2000 users have used the TRACECORONA app without any functional problems.

In summary, we provide a comprehensive set of requirements to evaluate DCT systems. We show that current approaches do not fulfill such requirements at large, e.g., have number of security, privacy and effectiveness issues. Hence, we propose TRACECORONA, a novel approach that address the deficiencies of existing DCT system.

II. DIGITAL CONTACT TRACING

In this section, we present system models, architectures and technologies of Digital Contact Tracing (DCT) Systems.

TABLE I: Notations.

User (U)	A Person that uses a DCT App
User App (App)	A DCT app installed on users' devices
Tracing Service Provider (SP)	Providing a system (e.g., servers and apps) for identifying at-risk contacts
Health Authority (HA)	Authenticating the user infection status
Infected user	A user that has tested positive for COVID-19
Affected user	A user that has encountered an infected user
Indirect contacts	A user that has encountered an affected user

[4]In the course of this debate about 300 security and privacy researchers from 26 countries signed an open letter criticizing the specific privacy risks of some centralized contact tracing approaches, advocating privacy-preserving solutions whenever better privacy can be obtained without penalizing effectiveness (https://drive.google.com/file/d/1OQg2dxPu-x-RZzETlpV3lFa259Nrpk1J/view). This signed letter has been often abused claiming that centralized systems are bad and decentralized systems do what is needed to detect at-risk contacts, and moreover they do it protecting privacy.

[5]MD COVID Alert, https://tinyurl.com/yeymtrm2

A. System Model

Figure 1 shows the typical system model of contact tracing schemes. There are three types of entities: Users U (e.g., U_i and U_j) of the tracing system (app), a contact tracing Service Provider (SP), as well as a health authority (HA). In the following, we discuss these roles in more detail.

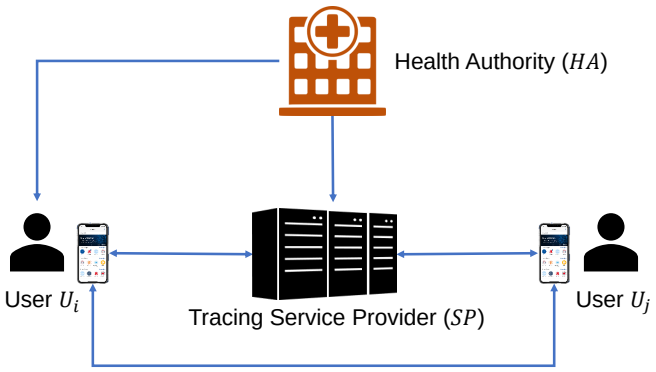


Fig. 1: System model of Digital Contract Tracing (DCT).

1) Users: A user U_i uses a dedicated *contact tracing app* installed on its device (typically a smartphone) to collect information required to determine contacts with other users of the system. Different technologies can be used for this purpose, e.g., directly through exchange of specific information over a proximity communication protocol like Bluetooth LE, or, indirectly with the help of a trace of location information obtained from a positioning system like GPS, by determining simultaneous co-presence of the users at the same location at the same time. We will discuss various technologies in Sect. II-C. Users' contact tracing apps collect and store this information about contacts of users locally on users' mobile devices. In case a user U_i is tested positive with a disease (like COVID-19), the user is expected to use the contact tracing app to warn other users of the system by uploading the collected information about his/her contacts to the contact tracing service provider SP.

2) Tracing Service Provider: The Tracing Service Provider SP is responsible for collecting and distributing information necessary for identifying contacts with infected users and/or notifying other users of such contacts. In *centralized* systems, the SP determines contacts between infected users and other users and issues notifications to them, whereas in *decentralized* systems, the determination of possible contacts is performed by the users' contact tracing apps.

3) Health Authority: The Health Authority HA is responsible for identifying infected users (e.g., through administered medical tests) and authenticating their infection status towards SP. This is necessary to prevent malicious users $\mathcal{A}^u$ from pretending to be infected and thereby triggering false alarms with users they have had contacts with. To do this, HA will issue a user-specific unique authenticator, e.g., a transaction authentication number (TAN) (a form of single use one-time password (OTP)) to an infected user U_i, who can subsequently present this authenticator when uploading their information to SP. By verifying the authenticator with HA, the SP can verify the infection status of the user U_i.

B. Centralized vs. Decentralized Architectures

In general, contact tracing approaches can be divided into two main design architectures, *centralized* and *decentralized*, based on whether the identification of encounters between users is performed by the tracing service provider SP or by the tracing apps of users U. Figure 2 shows an overview of both architectures. Both approaches are based on individual users' tracing apps recording temporary identifiers (*TempID*s) of other devices they encounter. In the case a user U_i is infected, he uses his tracing app to upload identifiers to SP. In centralized systems, the recorded identifiers of *other* apps will be uploaded, whereas in decentralized systems, the *TempID*s used by the tracing app *itself* in the recent past will be uploaded.

The main difference between these schemes is the fact that in the centralized system the service provider SP generates all *TempID*s centrally and is therefore able to link the infected user with the (pseudonymous) identities of other users, whereas in the decentralized approach, the *TempID*s are generated individually by each tracing app. The determination of contacts can therefore only be performed by the actual tracing apps involved in an encounter. The tracing app conducts this by downloading the *TempID*s of infected users, e.g., U_i from SP and comparing these to the *TempID*s the tracing app has encountered in the past. This approach therefore effectively limits the exposure of sensitive information about encounters to SP.

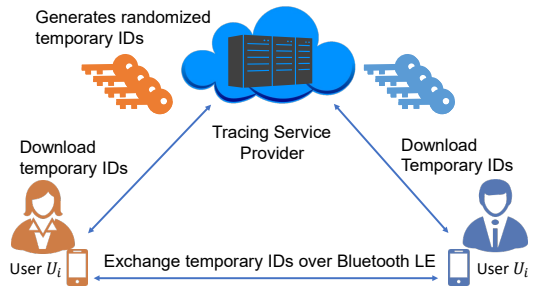
In contrast to common belief, however, this difference does not directly guarantee "privacy by design" for decentralized systems and susceptibility to "mass surveillance" in centralized systems. The actual evaluation of these models highly depends on the underlying architectural decisions and on the various threat models considered. We will analyze the effectiveness, privacy and security of these architectural approaches in more detail in Sect. V.

Before considering these approaches in more detail, we will elaborate on the security and privacy requirements in Sect. III that are important driving factors behind the decisions to adopt particular contact tracing approaches as these are also highly influenced by political and legal decisions taken by governments in individual countries.

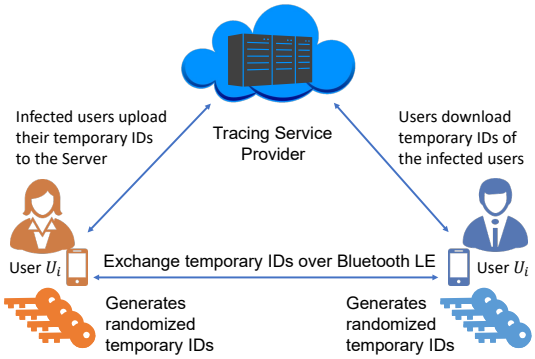
C. Technologies to Determine Encounters

In general, there are two types of technologies to determine encounters: (1) location-based technologies such as GPS and QR-codes used for venue check-ins and (2) peer-to-peer proximity detection-based technologies like Bluetooth, Ultra-wideband (UWB), and ultrasound. Currently, Bluetooth is the most dominant technology deployed in contact tracing.

GPS-based Location. GPS tracking can be used to determine the location of individuals and hence encounters between persons present at the same location at the same time (e.g., [10]). However, in practice, using GPS for this purpose is challenging, as it is relatively inaccurate, especially in indoor



(a) Centralized.



(b) Decentralized.

Fig. 2: Centralized vs. decentralized architectures

areas. Indoor encounters are particularly important to be captured accurately due to the higher risk of contagion in closed spaces. More importantly, GPS traces of users can reveal a lot of sensitive information about users and their habits[6]. In a centralized GPS-based approach, GPS traces of all users are typically sent to the *SP*. This information, however, can be misused for surveillance and profiling. In a decentralized approach, infected users have to upload their GPS traces, i.e., make them publicly available so that other users can check their potential encounters with the infected person by comparing their own location trace with the released locations of the infected person. The use of these solutions requires enabling the location service on the mobile device. This implies, however, that also other apps that have been given the permission to use the user's location, including, e.g., Google's location service on Android devices, can collect detailed location traces potentially along with other data such as IP and MAC addresses [11].

Privacy protection in such schemes can be addressed to some degree by allowing infected users to redact locations that they deem sensitive from their individual traces before sharing them with others. This, however, can have serious negative effects. Indeed, many potential encounters may be lost if users redact sensitive locations like workplace or specific entertainment venues like bars or clubs from their traces, thus reducing the utility of the system for contact tracing.

QR Code-based Location. Instead of using geolocation information to identify the presence of a user at a location, QR codes placed at specific venues can be used to allow users to "check-in" at the venue and thus record information about the presence at the venue at a specific point in time. This can be either done by recording identifiable contact information of users and venues including names, phone numbers, and addresses (of the guests and restaurants) in a QR Code so that it can be exchanged easily by scanning the QR code using a mobile app [12], similar to check-in approaches in hotels. In some other solutions, users scan a QR code associated with a venue. The QR code contains a venue-specific URL linking to a cloud-based service *SP* associated with the venue, which the mobile or web app uses to enter the contact information of the user. To the best of our knowledge, deployed QR-code based apps follow a centralized architecture, where all data (encounter information) are stored by *SP* [12]. Some approaches, such as [12], enable users (including individual users and the venues) to encrypt the data before sending them to *SP* (the encryption keys are kept locally). In particular, individual users encrypt the contact information of venues (e.g., addresses, phone numbers, and the time of visit) and vice-versa, i.e., the venues also encrypt the contact information of users). If a user U_i is tested positive for COVID-19, the user will send the encryption keys to health authority *HA* who can decrypt the encounter information stored by *SP* and know which venues the user has visited and at what time. Afterwards, *HA* will ask those venues to provide keys to decrypt contact information of all users (guests) also stored by *SP* who visited the venues around the given time. Finally, *HA* notifies the corresponding users. However, similar to GPS-based approaches, QR Code-based approaches also reveal the movement traces and encounter information (who they have met) of infected users [13]. Another drawback of QR-code-based systems is that for it to be effective, many venues such as restaurants need to adopt the system.

Bluetooth Low Energy (BLE). BLE can be used for sensing the proximity between individual users' devices, e.g., [3], [14]. Indeed, many recent approaches for contact tracing on smartphones use Bluetooth proximity detection. The participating smartphones beacon out information like temporary identifiers (*TempIDs*) that can be sensed by other devices. In addition, also related metadata like the signal strength of the beacon may be recorded. Using the signal strength information, some approaches seek to provide estimates about the distance of the encounter. However, it has been shown that signal strength can provide only a very rough estimate about the actual distance of devices, as it is influenced by other factors like device orientation and surrounding structures [15]. Nevertheless, since BLE is widely available on most recent smartphone versions, it seems the most viable alternative for implementing proximity detection on smartphones that are widely used by the population in many countries.

Compared to GPS and QR-code based approaches, BLE would seem to reveal the least amount of information about the users because *HA* and *SP* do not collect physical locations as

[6]"They Stormed the Capitol. Their Apps Tracked Them.", https://tinyurl.com/5ddun4dm

well as actual encounter times. Thus, only anonymized random strings are shared among the apps using BLE. However, BLE-based approaches still have several security, privacy, effectiveness, and ethical problems. For example, they are susceptible to fake exposure injection attacks, e.g., relay attacks, or user profiling, e.g., movement tracking and user identification. We will elaborate all of these problems in detail in Sect. V.

Ultra-Wideband (UWB). In contrast to Bluetooth and other technologies that cannot measure distances among users precisely, UWB radio technology can be used to measure the distance at an accuracy level of a few centimeters. Therefore, some approaches propose using UWB [16]. However, using UWB faces some challenges such as the fact that very few smartphones support UWB and that it is not energy efficient [16]. This makes UWB-based approaches less practical. Istomin et al. [16] propose a hybrid approach that combines Bluetooth and UWB, so that Bluetooth is used to constantly broadcast BLE signals and to identify other devices in vicinity. Once an encounter is identified (i.e., a device detects another device via BLE), UWB is used to measure the exact distance between two devices in order to estimate the risk of exposure.

Ultrasound. Since ultrasonic communication range is very short (only effective for distances shorter than one meter) and likely blocked by walls or other objects, it can also be used for determining encounters [17]. However, Meklenburg et al. [17] also raise many open questions about the practicality of ultrasound-based approaches such as scalability (as it appears nearly impossible to handle multiple connections because it is hard to transfer and separate several ultrasonic signals simultaneously), low accuracy (due to ambient noise), or high power consumption. Further, this approach requires microphones and speakers of smartphones to be enabled. This would make devices vulnerable to several known privacy and security attacks such as tracking user's activities, leaking user's conversations, or injecting fake voice commands [18].

III. Quadrilemma Requirements

As mentioned above, digital contact tracing (DCT) schemes need to collect information about infected individuals. Although many countries have deployed contact tracing apps, the effectiveness of DCT is so far still unclear. Moreover, DCT poses a number of privacy and security challenges on the underlying scheme design, since it collects and processes sensitive information which is related to users' health and users' contacts to some extent. In this section, we systematically consider the requirements for DCT based on four pillars: effectiveness, privacy, security, and ethical aspects. These requirements are broken down and listed in Tab. II. Next, we will discuss each of them in detail.

A. Effectiveness

1) Accuracy (R-Ef1) - Distance and Duration: For accurately estimating the risk of contagion it is necessary to estimate the duration of each contact (in minutes) along with a good estimate of the distance between the users involved in the encounter. The duration of contacts ideally could be detected by continuously scanning for the presence of BLE devices in proximity to verify the continued presence of other devices. This aggressive approach will, however, lead to significant energy consumption draining the smartphone battery quickly. In practice, one needs therefore to pause the scanning for several seconds before the next scan to preserve energy. Computing a good estimate of the distance between devices is even more challenging since there are multiple factors (e.g., positioning of the antenna in the smartphone, obstacles in between smartphones, and their orientations) that introduce significant errors to distance estimates. Indeed, experiments performed by Leith and Farrel [15] showed that GAEN is quite imprecise in estimating the distance of devices of potential at-risk exposures.

2) Superspreaders (R-Ef2): The mere capability of detecting at-risk exposures was initially considered sufficient by many endorsers of decentralized systems like, e.g., the team around the influential DP-3T [19] contact tracing approach, which also had a considerable influence on the GAEN design adopted by Google and Apple. However, along the way, more epidemiological insights about the behavior of SARS-CoV-2 have been discovered. Among them is the fact that a very relevant aspect for understanding the spread of the virus is the important role of so-called *superspreaders*. Indeed, Reichert et al. [20] showed that while there is a large percentage of infected individuals that do not transmit the virus at all, there is a small fraction of infected individuals that instead are very contagious and cause numerous further infections. A DCT system aiming at effectively defeating SARS-CoV-2 should therefore also take into account the importance of superspreaders and provide mechanisms allowing to detect them and their potential contacts.

Contagious asymptomatic infected individuals (CAIIs). Particularly problematic are so-called asymptomatic infected individuals, i.e., persons that are infected and contagious, but asymptomatic and thus may unwillingly spread the disease. Such individuals have a very low chance of being tested positive since they do not show any symptoms of being sick and therefore will not likely seek to be tested. Even if they want to be tested, in many countries, they will not be prioritized in testing. Hence, they can have an active role in spreading the virus. However, as such individuals are unlikely to be tested and receive a positive diagnosis from *HA* (which is a prerequisite for uploading information about contacts to the service provider *SP*), it is unlikely that such persons will ever be able to use the DCT system to warn other users about possible at-risk contacts with them.

3) Accountability (R-Ef3): Implementing, deploying, and operating a DCT system can be very costly and requires a majority of the population to participate in its operation. Therefore, the system should provide adequate and valid information about its effectiveness in a privacy-preserving way. For example, the system should be able to provide basic statistics about the number of active users, infected users, users notified about potential at-risk exposures, as well as false positive rates, etc. At a minimum, the system should be able to demonstrate clear benefits in comparison to a purely random selection of users to be quarantined in specific at-risk groups (e.g., where the infection rate is higher) [15]. Although

some GAEN-based apps do provide reports on some measures related to the system's effectiveness, such measures can be biased, unreliable or misleading [21], [8] as we will discuss in detail in Sect. V-B1.

B. Privacy

The main privacy concerns relate to the abuse of a DCT in order to *identify* users, *track* users, or *extract the social graph* of users. Information that is emitted to the user's surroundings by contact tracing apps and shared with other involved parties should not introduce such privacy risks as elaborated next.

1) Identifying users (R-P1): DCT systems aim at identifying encounters, not users. Therefore, the systems should not leak any information that can be used to establish the true identity of any individual user.

2) Tracking users (R-P2): DCT apps work by continuously beaconing pseudonymous identifiers into their surroundings. These identifiers should not be linkable, i.e., it should not be possible to trace the movements of any user over time, as this may potentially enable to deduce facts about the user's behaviour and lead to an identification of the user.

3) Extracting the social graph (R-P3): In general, contacts (especially long encounters), are often related to social relationships (i.e., users that decide to be close to each other). When handling contact information, a DCT system should make sure that one cannot abuse information collected by it to generate a relevant part of the social graph of any user, since this may enable to draw conclusions about social relationships between users and thus potentially identify them.

Note: Obviously, there exists in some cases inherent information leakage due to specific circumstances, e.g., in situations in which the adversary is in the proximity only to one specific person. If the adversary later receives an at-risk notification, it will be trivial for the adversary to conclude that this one person is indeed the infected person. Therefore, when considering the above three privacy requirements, we will always focus on *large-scale attacks* and will in particular focus on identifying attacks affecting potentially many users.

C. Security

The effectiveness of a DCT system is severely impacted if a system is not resilient to large-scale data pollution attacks. Such attacks can generate, for instance, false at-risk notifications (false positives) therefore jeopardizing the correctness of the contact tracing system. Indeed, massive false at-risk notifications could result in spreading panic among the general population. Moreover, this could also cause unnecessary strain on the health system through unnecessary testing and negative impact on the society due to unnecessary self-quarantining.

1) Fake exposure claims (R-S1): The system should prevent a malicious or dishonest user $\mathcal{A}^u$ that aims to circumvent the DCT system to claim that he or she has encountered an infected user. There can be different motivations for this attack: (i) $\mathcal{A}^u$ aims to harm the reliability of the system by manipulating encounter checking results, (ii) $\mathcal{A}^u$ uses the fake exposure status as an excuse to stay at home instead of going to work or participating in an event, and (iii) $\mathcal{A}^u$ intentionally shares wrong encounter information to epidemiologists, thus sabotaging their analysis of the epidemiological situation.

2) Fake exposure injection - Relay/replay attacks (R-S2): This attack aims to inject fake contacts on a large scale resulting in many false exposure notifications. Here, a fake contact indicates the state that the DCT system incorrectly determines that two users were in "close contact" at a specific time although they were not. It affects the main goal of DCT system as to identify contacts that potentially cause high exposure risks. Relay attacks are a typical example of fake exposure injection attacks. In a relay attack, the adversary captures the temporary IDs of a user U_i and broadcasts them in other locations (e.g., other cities). As a result, the system incorrectly identifies the users in the other locations who captured those temporary IDs to have encountered U_i.

D. Ethics

1) Transparency and voluntary participation (R-Et1): The whole process (design, development, deployment, and operation) of a contact tracing system must be transparent to users and the systems must be removed immediately when the pandemic is over to avoid misuse. Further, users should be free to decide whether they want to participate in the system or not, and be free to withdraw their participation anytime they wish. Otherwise, users will not trust, and thus will not be willing to use DCT apps. This will affect the crucial need of a high adoption rate of DCT.

2) Independence (R-Et2): The contact tracing process (design, development, deployment and operation) in a particular region should be independent of any parties with potential vested interests. Procedural controls of the contact tracing system should underlie a transparent public scrutiny and be solely under the control of democratically-elected governments. In particular, giant technology corporations (e.g., Mobile OS vendors) should not be allowed to use their technological or market dominance to control or drive DCT systems since they might be biased in it for the sake of their own subjective benefits, e.g., using DCT data for business purposes could undermine the de-facto ability of legitimate governments to oversee the use of data collected for contact tracing purposes.

IV. Systematization of Contact Tracing Schemes

In this section, we systematize state-of-the-art contact tracing schemes and their adoptions.

A. Architectures

1) Centralized Architecture: Centralized contact tracing approaches have been already deployed in many countries, e.g., Singapore (TraceToGether[7]) and France (TousAntiCovid[8]). Some of these schemes can be considered privacy-invasive as they collect sensitive personal data like GPS/locations, and phone numbers, e.g., AarogyaSetu[9], the Indian App. In this

[7]https://www.tracetogether.gov.sg/

[8]https://www.gouvernement.fr/info-coronavirus/tousanticovid

[9]https://www.mygov.in/aarogya-setu-app/

TABLE II: List of requirements for digital contact tracing.

	Requirement	Description
	Effectiveness	
R-Ef1	Accuracy	Specifying distance and duration of encounters
R-Ef2	Superspreader	Identifying superspreaders and their contacts
R-Ef3	Accountability	Providing statistics to evaluate the actual effectiveness
	Privacy	
R-P1	Identifying users	Users should always remain anonymous
R-P2	Tracing users	Users should not be tracked
R-P3	Extracting social graph	Making sure that no social graph can be extracted
	Security	
R-S1	Fake exposure claim	Preventing malicious users to lie about their exposure status
R-S2	Fake exposure injection	Preventing relay/replay attacks
	Ethics	
R-Et1	Transparency and voluntary use	The system must be transparent and based on voluntary use
R-Et2	Independence	Ones should not be allowed to use their technological or market dominance to control DCT systems in their favour

section, we primarily focus on schemes that use BLE technology (as apposed to GPS) to determine users' encounters, because it is the the most prevalent and allows for building less privacy-invasive tracing systems.

Generic Framework. We will systematize prominent contact tracing solutions based on centralized architecture, e.g., BlueTrace [14], PEPP-PT [22] and TousAntiCovid that are conceptually very similar. We first present a generic framework for centralized schemes and then discuss how it is impllemented in practice. Figure 3 shows a typical generic centralized protocol. The system consists of a Tracing Service Provider *SP* and users, e.g., User U_i and User U_j. In **Step 1**, users register their contact tracing app to the system either in an anonymous [22] or identifiable way (e.g., using their mobile phone number) [14]. The Tracing Service Provider *SP* generates and sends a unique pseudonymous *UserID* associated to each user. Each user generates batches of temporary identifiers *TempID*s associated with each *UserID* by encrypting the *UserID* (using an HMAC-based Key Derivation Function - HKDF) along with validity timestamps t_k as follows: $TempID_i^{t_k} = HKDF(UserID_i, t_k)$ and $TempID_j^{t_k} = HKDF(UserID_j, t_k)$. In **Step 2**, the user's tracing app beacons *TempID*s into its proximity and simultaneously listens for other user apps' *TempID*s and stores them locally. **Step 3** shows the matching and notification steps in case U_i is tested positive for SARS-CoV-2: U_i uploads the *TempID*s of the users that U_i has encountered along with corresponding timestamps (e.g., during the last 14 days) to the *SP*. *SP* then identifies the users who have encountered U_i by comparing the received temporary identifiers to ones derived on the server side. If there is a match, *SP* notifies the identified users.

a) PEPP-PT and TousAntiCovid: TousAntiCovid is the official app in France and it is based on PEPP-PT. These approaches follow the generic framework shown in Fig. 3. In contrast to BlueTrace discussed below, users do not need

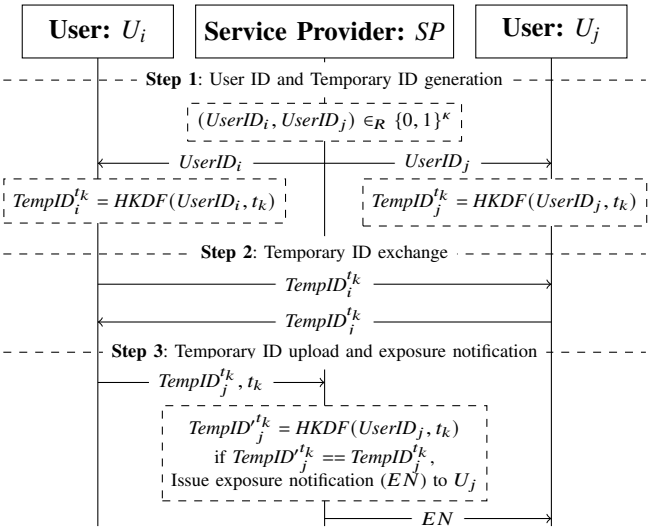


Fig. 3: Generic framework of centralized approaches.

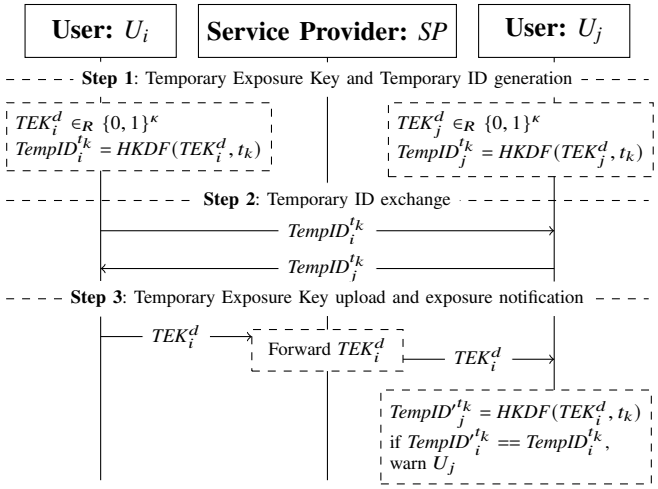


Fig. 4: Generic framework of decentralized approaches.

to provide personal information like mobile phone numbers to register with *SP*. Therefore, *SP* only notifies users (as identified by their *UserID*) via the contact tracing app.

b) BlueTrace [14]: This is a contact tracing framework adopted by, e.g., Singapore (TraceTogether) and Australia [CovidSafe). BlueTrace follows the generic framework with some modifications. Firstly, it requires users to register using their mobile phone numbers so that *SP* can then contact the corresponding users directly by telephone and notify them about the contact with an infected person. Secondly, users' tracing apps do not generate *TempID*s by themselves but receive pre-generated *TempID*s from *SP*. To derive the *TempID*s, BlueTrace uses the *HKDF* function with four inputs: a user ID *UserID*, a timestamp t_k, a random initialization vector *IV*, and an authenticity tag *AuthTag* for integrity verification using a secret master key *K* known only to *SP*, as shown in (1):

$$TempID = HKDF(UserID \| t_k \| IV \| AuthTag) \quad (1)$$

In addition to BlueTrace, PEPP-PT, and TousAntiCovid, a number of other centralized tracing approaches have been

proposed. We discuss these approaches in detail in Appendix A.

Hybrid approaches or centralized approaches. Castelluccia et al. [23] propose a *hybrid solution* called Desire that combines centralized and decentralized techniques. The main differences to the generic framework are in **Step 1**, where instead of temporary identifiers, Diffie-Hellman (DH) keys are generated by the apps, and in **Step 3**, where infected users upload encounter tokens derived from the DH keys in *an anonymous way*, e.g., using an anonymization network like Tor or another Mix network. However, according to our classification introduced in Sect. II-B, Desire should still be considered a centralized approach, since the Tracing Service Provider *SP* still (1) receives all encounter tokens, i.e., *TempID*s from all users, (2) performs encounter token matching, and (3) is responsible for sending exposure notifications to corresponding users based on the users' App IDs. We will discuss the effectiveness of Desire further in Sect. V.

2) Decentralized Architecture: A number of decentralized contact tracing approaches have been proposed, e.g., DP3T [19], MIT-PACT [24], and the *Exposure Notification API* [3] of Apple and Google (GAEN), which is closely related to the so-called Design 1 of the DP3T project (DP3T-1). Conceptually, these approaches are highly similar and follow the basic structure shown in Fig. 4.

Similar to centralized approaches, the decentralized approaches are based on temporary identifiers *TempID* (called *Rolling Proximity Identifiers* in GAEN, *Ephemeral Identifiers* in DP3T or *Chirps* in MIT-PACT) that tracing apps beacon into their surroundings over BLE. In **Step 1**, the *TempID*s are generated locally by using an *HKDF* function that encrypts a random *Temporary Exposure Key (TEK)* along with a timestamp t_k, e.g., $TempID_i^{t_k} = HKDF(TEK_i^d, t_k)$ where d is the epoch, e.g., the day that the key is valid in GAEN. Existing approaches usually define individual *TempID*s to be valid for 10 to 15 minutes. In **Step 2**, similar to centralized approaches, user Apps broadcast their *TempID*s and record *TempID*s from other apps in their vicinity. The fundamental difference to the centralized architecture is in **Step 3**, as the checking for possible encounters and notifying the user in case an at-risk contact is identified are performed locally by user Apps. As shown in Fig. 4, an infected user U_i uploads her *TEK*s, e.g., TEK_i^d to *SP* which forwards them to all other users. A non-infected user U_j downloads the *TEK*s and extracts *TempID′*s, e.g., $TempID'^{t_k}_j = HKDF(TEK_i^d, t_k)$. U_j then checks whether there is a match with the *TempID*s that he has collected from other users who he has encountered. If there is a match (i.e., $TempID'^{t_k}_i == TempID_i^{t_k}$), it means an at-risk exposure is identified and the user is notified.

a) GAEN and DP3T design 1 (DP3T-1): These schemes follow the generic framework. In these schemes, the *TEK* is also called Daily Tracing Key (DTK) referring to the fact that *TEK* will be changed daily. *TempID*s are changed every 10 minutes so that 24x6=144 *TempID*s will be generated per day.

b) DP3T design 2 (DP3T-2) [19], and MIT-PACT [24]: In these designs, the *TEK* changes at the same frequency as *TempID*s, so that the *TempID* of an infected user U_j are unlikable, even if the associated *TEK*s are published by the *SP*.

In addition to DP3T [19], MIT-PACT [24], and GAEN [3], there are several other decentralized tracing approaches, e.g., Pronto-C2 [25], ClerverParrot [26], and Epione [27]. We discuss these approaches in detail in Appendix A.

Advanced Privacy Techniques. There are several (mainly cryptography-based) privacy techniques that have been considered to enhance privacy of DCT, e.g., Blind signatures [28], [25], Blockchain [25], Private Set Intersection (PSI)/filter [27], and secret sharing [19]. Unfortunately, none of these contact tracing approaches have been used in practice yet. We discuss these approaches in detail in Appendix B.

V. Contact Tracing Schemes and Quadrilemma

In this section, we systematically analyze the most prominent centralized (BlueTrace and its derivatives) (Sect. IV-A1) and decentralized DCT approaches (DP3T, GAEN and PACT) (Sect. IV-A2) with regard to the aspects laid out in the quadrilemma introduced in Sect. III. We summarize our analysis in Tab. III.

Note on technical limitations of BLE Accuracy. As discussed in Sect. III-A1, measuring the distance between smartphones using BLE is not very reliable due to its the inherent technical limitations. Hence, we note that all approaches based on BLE-proximity sensing share the same challenge of not being able to reliably estimate the distance between devices involved in a contact. Therefore, none of such approaches will be able to entirely fulfill the Accuracy requirement R-Ef1. We will therefore exclude this aspect from our subsequent analysis.

A. BlueTrace and Others

1) Effectiveness: In this subsection, we evaluate the effectiveness of BlueTrace and its variants based on the requirements introduced in Sect. III-A.

Superspreader. In BlueTrace, the *SP* receives full information about all encounters of all infected users, and can thus use this information to identify potential superspreaders of CAIIs. If *SP* detects that several infected users have encountered the same user, then the the likelihood of that user being a superspreader is relatively high. Hence, *SP* can recommend such users to get tested and, perhaps more importantly, can immediately notify other users who have encountered the CAII user. Hence, the requirement R-Ef2 is fulfilled.

Accountability. Centralized approaches and hybrid approaches like Desire [23] also provide useful information for epidemiologists and policy makers as follows:

- Encounter information, w.r.t. lockdown/social distancing measures: The number of encounters and metadata (e.g., duration of each encounter) of individual users can provide information about the effectiveness of different lockdown or social distancing measures, i.e., how the number of encounters increases/decreases, or, how encounter durations change.
- Exposure information: How frequently and for how long users encounter infected users.

- Hotspot prediction: The *SP* can predict hotspots based on encounter and exposure information if it keeps track of where a user is (only rough information, e.g., the postcode or the city). The *SP* can do this by checking the IP addresses of client apps from ISPs; the server does not need to store the IP addresses. It is practical since, e.g., most existing web servers/content delivery networks (CDNs) have the IP addresses of their visitors. Alternatively, the system can ask users to provide their postcodes/cities when they install the App, which is unlikely to be a major privacy concern for most users.

Therefore, centralized approaches fulfill Requirement R-Ef3.

2) Privacy: In the following, we evaluate how BlueTrace fulfills privacy requirements introduced in Sect. III-B.

Identifying users and Extracting social graph. Existing centralized contact tracing approaches (Sect. IV-A1) are vulnerable to this attack because *SP* has knowledge of temporary identifiers *TempID*s of users involved in contacts. Hence, R-P1 is not achieved. *SP* can use this information to extract the encounter graph of infected users (i.e., which users have been in contact with infected users) since the matching is performed by *SP* [29], [8]. This could allow *SP* to build the social graph of the infected users [30]. As such, R-P3 is not achieved either. BlueTrace is vulnerable to linkability attacks [31] in which the *TempID*s of a device can be linked based on their corresponding BLE MAC addresses (cf. Sect. V-B2).

Tracing users. In a centralized contact tracing system such as BlueTrace, a malicious tracing service provider $\mathcal{A}^s$ knows all *TempID*s of each user while an eavesdropping adversary $\mathcal{A}^e$ knows where and when a temporary identifier *TempID* was observed. Therefore, the collusion of these two adversaries can lead to significant violations of user privacy as the movements of all users and their contacts can be tracked [29], [25]. It also means the adversary can derive the social graph (e.g., knows where and when users have met) of all users so that the R-P2 requirement is not fulfilled.

3) Security: In the following, we will summarize how existing attacks affect the security of BlueTrace its variants.

Fake exposure Injection. BlueTrace and others are vulnerable to *large-scale* relay attacks because the adversary $\mathcal{A}^w$ can easily capture and relay *TempID*s that the apps frequently beacon via Bluetooth. It does therefore not satisfy requirement R-S2 [29], [32].

4) Ethics: BlueTrace is open-source and provides a solution for stand-alone apps (i.e, it does not depend on any built-in contact tracing APIs running in mobile operating systems such as Android and iOS). This makes the app transparent to users, and the health authorities have full control of the app's functionalities and deployment, satisfying the R-Et1 and R-Et2 requirements.

B. GAEN

In this section, we analyze decentralized approaches based on client-derived *TempID*s presented in Sect. IV-A2.

1) Effectiveness: Although GAEN-based apps have been deployed in many countries since 2020, their contributions with regard to confining the spread of the pandemic have been heavily disputed. Indeed, several existing studies raise serious questions concerning the effectiveness of GAEN-based apps [8], [33], [34] as well as the trustworthiness of the self-reported effectiveness of some GAEN-based apps. For example, a report of Wymant et al. [2] states that 6,1% of users who got notified via the NHS Covid-19 App were eventually tested positive. However, since the study took place in the UK during strict lock-down measures, most contacts at that time likely took take place within households or essential working places, so that a large fraction of the affected persons were likely tested due to the regular manual contact tracing process, regardless of possible app notifications. There is therefore no clear evidence whether GAEN-based apps have provided any significant additional benefits with regard to the regular manual contact tracing processes.

Superspreaders. In contrast to BlueTrace, in GAEN-based systems, *SP* does not have encounter information. Therefore, it is hard to see how GAEN could determine Superspreaders and CAII users. Potentially one could use the user's App to roughly determine whether the user is a Superspreader or CAII user based on the numbers of *TEK*s with infected users. However, this solution can be abused since the number of matching TEKs is not reliable, i.e., self-injection of fake at-risk contacts are trivial in GAEN, e.g., by applying the fake exposure claim attack (cf. Sect. V-B3), or, a relay attack such as the time-travel machine attack [6]. Hence, GAEN does not achieve R-Ef3.

2) Privacy: **Profiling Attacks.** Several works show that GAEN is vulnerable to Profiling Attacks [5], [21], [31], [35]. An eavesdropping adversary $\mathcal{A}^e$ can possibly track the movements of infected users in the previous days and in some cases it can also figure out who they have been in contact with, still taking into account recently infected users only. Based on these data, the adversary can build partial profiles and social graphs of the infected users [21], [19]. To achieve this, $\mathcal{A}^e$ deploys a Bluetooth sensor network to collect (sniff) *TempID*s emitted by the GAEN. $\mathcal{A}^e$ then links the collected *TempID*s to each infected user, resulting in a movement graph of the users. From this data, $\mathcal{A}^e$ can infer sensitive information about infected users and their activities, e.g., where infected users live (locations where they stay during evenings and at night), work and leisure activities. Hence, requirements R-P2 and R-P3 are not satisfied.

Furthermore, the movements of infected users can be traced by exploiting a weakness in the scheme requiring infected users to upload their *TEK*s from which all rolling proximity identifiers (*TempID*s) of a whole day can be derived. $\mathcal{A}^e$ can thus link the *TempID*s that are derived from the same *TEK*, so that $\mathcal{A}^e$ can get to know the movement profiles of the infected users within its Bluetooth sensing network. Moreover, since movement patterns of a user may have marked similarities over several different days, $\mathcal{A}^e$ may also be able to link movement profiles of the same user over more than one day (R-P2 is not achieved).

To mitigate this attack vector, Troncoso et al. [19] propose a more advanced "design 2" version of their DP3T protocol and Rivest et al. [24] propose a solution named PACT that allows infected users to only upload short-lived ephemeral IDs (that are changed every 15 minutes or even faster). This means $\mathcal{A}^e$

can only track users for less than 15 minutes, which is not enough to build informative movement profiles of the user.

Yet, Knight et al. [31] show that GAEN and other BLE-based approaches on Android are vulnerable to tracing due to linkability of *TempID*s.

In this attack, an adversary can link *TempID*s based on their corresponding random BLE MAC addresses when *TempID*s are advertised (beaconed) using resolvable Random Private Addresses (RPAs). This is because all random BLE MAC addresses of a device are derived from the Identity Resolving Key (IRK) of the Bluetooth adapter that is unchanged unless the device is factory-reset. This attack affects all Android apps since Android only supports resolvable RPA protocol while iOS also supports non-resolvable RPA protocol.

Tracing users. In decentralized contact tracing approaches like GAEN, adversary $\mathcal{A}^s$ (a dishonest *SP*) can get to know the Temporary Exposure Keys (*TEK*s) of infected users (for a maximum of 14 days). Hence, movements of infected users can be tracked up to 14 days, i.e., R-P2 is not satisfied.

Privacy problems caused by logging. Reardon et al., [36] show that GAEN keeps the log of Bluetooth communications, so that any built-in apps have access to it. This can be misused to identify users, tracking users movements as well as social graphs of the users. Hence, all three privacy requirements are not fulfilled.

3) Security: Existing works show that GAEN is vulnerable to several attacks. Next, we will elaborate how such attacks affect the security of GAEN.

Fake exposure claims. Existing distributed tracing schemes (e.g., [3], [19], [24]) that are based on simply exchanging temporary identifiers *TempID*s (called rolling proximity identifiers or RPIs in GAEN), are vulnerable to fake exposure claim attacks because a malicious user $\mathcal{A}^u$ can falsely claim that a *TempID* of an infected user downloaded from the tracing server is actually one that it has collected via the Bluetooth channel, thus incorrectly indicating a contact with the infected user, in violation of requirement R-S1.

Relay/Replay Attacks. Similar to BlueTrace, GAEN, DP3T-1, DP3T-2 and PACT are also vulnerable to *large-scale* Relay/Replay Attacks, in which the adversary $\mathcal{A}^w$ captures and relays rolling proximity identifiers *TempID*s beaconed by the apps. Even worse, the GAEN API [3] allows for a two-hour time window for $\mathcal{A}^w$ to conduct relay attacks. Recently, various flavors of relay attacks against GAEN were introduced [5], [21], [37], [7], [29], [38], [39] For instance, Baumgärtner et al. [5] realized this attack on the German Corona-Warm-App [40] which is based on GAEN. They showed that $\mathcal{A}^w$ can even use off-the-shelf smartphones to capture and relay *TempID*s between three different cities in Germany. Morover, Dehaye and Reardon [39] introduce a new effective *large-scale* relay attack against the *SwissCovid* app also based on the GAEN API. They demonstrate that a malicious Software Development Kit (SDK) can transform benign devices into malicious relay stations for relaying *TempID*s without the knowledge of the device owners. Hence, requirement R-S2 is clearly not satisfied by GAEN.

Time travel attacks. Iovino et al. [6] demonstrate that an adversary $\mathcal{A}^w$ does not need to capture *TempID*s from potentially infected users, but can replay a *TempID* derived from an "outdated" *TEK* key advertised on the contact tracing server directly. In order to make the outdated *TempID* valid, they introduce a time machine attack in which the adversary $\mathcal{A}^w$ in proximity of a victim remotely manipulates the clock of the victim's phone by setting it back with the help of a simple commodity device (that costs only US $10). When the victim's clock is later eventually restored to the correct time, the tracing app will associate the replayed *TempID* to a valid *TEK* and raise an exposure notification alert.

The KISS attack. In the same work [6], Iovino et al. also pointed out a bug in GAEN-based contract tracing apps that falsely accept *TempID*s captured during the same day when the *TEK* was published. Therefore, the adversary only needs to download a *TEK* of the current day, derive *TempID*s from the *TEK* and replay them.

Tracing Forgery: The Terrorist Attack. In this class of attacks (see [7]), the adversary colludes with infected users (who want to monetize their infection status) to obtain their Temporary Exposure Key (*TEK*). The adversary then replays *TempID*s derived from the *TEK* at the adversary-chosen (targeted) places. Finally, the colluding infected user will upload the *TEK* resulting in false exposure alarms in the devices that have captured the replayed *TempID*s.

4) Ethics: GAEN based apps have received criticism about ethical aspects such as lack of transparency and coercion [9], [8]. These works indicate that the requirements R-Et1 and R-Et2 are not achieved.

VI. Proposed Approach - TraceCORONA

In this section, we first provide a generic framework for Diffie-Hellman (DH)-based schemes. We then present our novel scheme, TraceCORONA, a fully fledged example of a DH-based approach and highlight its benefits compared to the prominent approaches analyzed in Sect. VII.

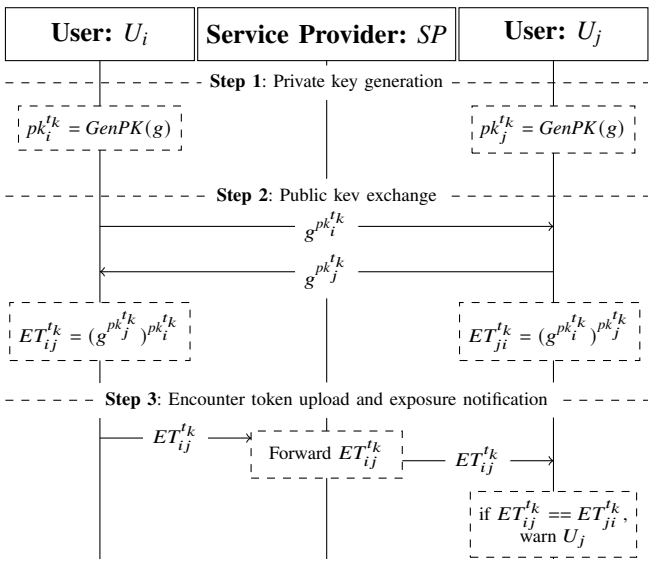


Fig. 5: Generic framework of DH-based Approaches.

A. Generic framework of DH-based approaches.

The core idea of decentralized approaches based on asymmetric key cryptography like Diffie-Hellman is that two users establish a *unique and secret* Encounter Token (*ET*) using a key exchange protocol when they are in proximity by exchanging short-lived random public keys via BLE. In this paper, we use Diffie-Hellman as a key exchange protocol. Figure 5 shows an overview of the use of DH-based encounter tokens in a contact tracing scheme. In **Step 1**, users U_i and U_j generate their own private keys $pk_i^{t_k}$ and $pk_j^{t_k}$ respectively for each time interval t_k that is changing every T (e.g., 15) minutes. These private keys are used to derive corresponding public keys $pubk_i^{t_k} = g^{pk_i^{t_k}}$ and $pubk_j^{t_k} = g^{pk_j^{t_k}}$. In **Step 2**, the public keys are exchanged via BLE when two devices are in vicinity. For encounters surpassing a specified minimal duration, e.g., 5 minutes, an *ET* will be calculated, e.g., U_i calculates $ET_{ij}^{t_k}$ from U_i's private key $pk_i^{t_k}$ and U_j's public key $pubk_j^{t_k}$ as follows: $ET_{ij}^{t_k} = (g^{pk_j^{t_k}})^{pk_i^{t_k}}$. Since U_i and U_j never share their private keys, only they can know their secret encounter token $ET_{ij}^{t_k}$. It is worth noting that the DH key generation and encounter token calculation processes do not need to happen on-line. For saving battery, it can be deferred to the next time when the smartphone is being charged. In **Step 3**, when a user (e.g., U_i) is tested positive for COVID-19, U_i sends its encounter token $ET_{ij}^{t_k}$ to the *SP* which will forward $ET_{ij}^{t_k}$ to other users. Once U_j receives $ET_{ij}^{t_k}$, it will compare $ET_{ij}^{t_k}$ to the *ET*s it has calculated. If $ET_{ij}^{t_k}$ is equal to $ET_{ji}^{t_k}$, U_j is notified that it has encountered an infected user.

Although we use the well-known DH-based approach for illustrative purposes, any other two-party key-exchange protocols where parties send only one short message to each other are applicable. Thus, existing proposals like CleverParrot [26], PRONTO-C2 [25], and Epione [27] use Elliptic-curve DH (ECDH). Further, these approaches provide several modifications and optimizations to improve the effectiveness, security and privacy of the system (cf. Sect. IX-A).

B. Limitations of DH-based approaches

Our proposed approach TRACECORONA seeks to address three technical limitations of DH-based approaches as follows:

- **Size restriction of BLE beacon message.** Since public keys are in general too big for BLE beacon messages, existing solutions apply workarounds, e.g., PRONTO-C2 needs to handle a bulletin board, or CleverParrot has to reduce the key size and requires operating systems to enable special BLE advertising messages.
- **Sharing encounter tokens *ET*s.** Uploading *ET*s directly may raise privacy risk. Hence, we aim to keep *ET*s always secret.
- **No time window restriction.** Existing approaches do not limit limit time window that would open opportunity for two-way relay attacks.

In the following, we will present TRACECORONA and discuss how we address those limitations in detail.

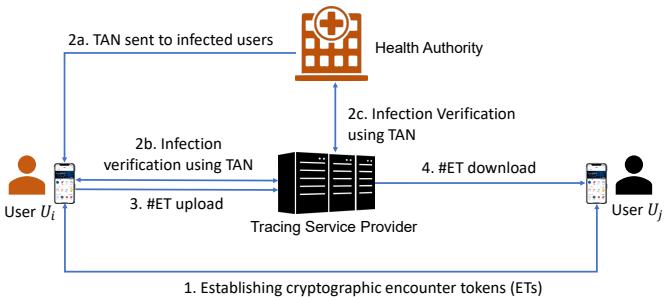


Fig. 6: TRACECORONA system overview.

C. TRACECORONA Design

1) System Overview: Our design follows the system model (cf. Fig. 1) and the generic framework for DH-based schemes shown in Fig. 5. An overview of the basic usage scenario of TRACECORONA is shown in Fig. 6. For a discussion on complementary application scenarios like wearable devices and private contact tracing please refer to Appendix C. The functionality of TRACECORONA can be divided into four phases: (1) Encounter token establishment, (2) infection verification, (3) token information upload, and (4) token information download and contact verification. Next, we will describe each of these phases in detail.

2) Encounter Token Establishment: TRACECORONA App uses BLE as a proximity communication protocol to advertise a random ephemeral identifier to other devices in the environment and to scan for the identifiers of other apps. Once an ephemeral identifier of another app has been observed for a minimum duration (e.g., 5 minutes), a connection over BLE to the other app is opened and an Encounter Token (ET) is established using the Elliptic Curve Diffie-Hellman (ECDH) key exchange protocol. Figure 7 shows the token establishment protocol in detail for two users U_i and U_j. Following typical ECDH notation, let Q denote the public key, d the private key and G the generator. Let T denote the period of a rolling key time frame and l be the index of the time frame $f^l = [l*T, (l+1)*T]$. Let K_i and K_j be the sets of ETs of users U_i and U_j, respectively. Let k_{ij}^l be an ET established between two user Apps U_i and U_j at time point t_{ij}^l, i.e., a timestamp falling in time frame f^l. The process of establishing an ET is then as follows:

1) **Step 1:** For every time frame f^l, users U_i and U_j generate a ECDH keypair including private keys d_i^l and d_j^l, and public keys $Q_i^l = d_i^l * G$ and $Q_j^l = d_j^l * G$, respectively, where G is the generator defining the used cyclic subgroup of the elliptic curve.
2) **Step 2:** U_i and U_j exchange their public keys Q_i^l and Q_j^l via Bluetooth LE.
3) **Step 3:** Each user calculates the encounter token based on its private key and the received public key. In particular, U_i calculates $k_{ij}^l = d_i^l * Q_j^l$ while U_j calculates $k_{ji}^l = d_j^l * Q_i^l$. Obviously, $k_{ij}^l = k_{ij}^l = d_i^l * d_j^l * G$. Each user then adds the encounter token into its encounter token set: $K_i \leftarrow K_i \cup \{k_{ij}^l\}$ for U_i and $K_j \leftarrow K_j \cup \{k_{ji}^l\}$ for U_j. After k_{ij}^l is established, U_i and U_j continue exchanging their

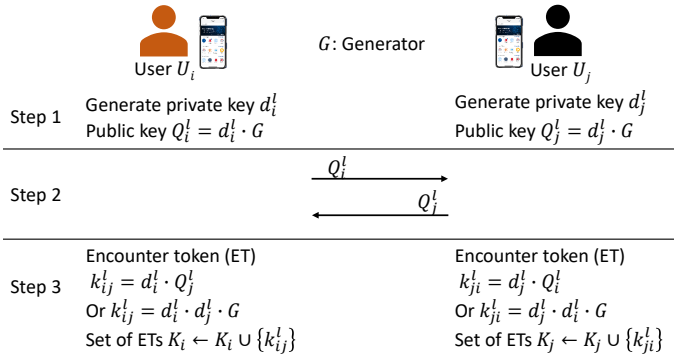


Fig. 7: Elliptic-curve Diffie–Hellman (ECDH)-based encounter token establishment.

ephemeral identifiers periodically to monitor the duration D_{ij}^l of the encounter and the strength of the Bluetooth signals S_{ij}^l (which roughly correlate with how far or near two users are from each other). In summary, the data recording the start of the encounter t_{ij}^l, the duration of the encounter $D_{k_{ij}^l}$ and the strength of the Bluetooth signal S_{ij}^l, are stored as metadata associated with token k_{ij}^l.

It is worth noting that in order to preserve battery lifetime, **Step 1** and **Step 3** can be done offline, i.e., when the smartphones are being charged (e.g., during the night).

3) Infection Verification and Encounter Token Upload: Since the main goal of the system is to notify users who have encountered infected users (tested positive for COVID-19), the system needs to make sure that only infected users can use the system to release their encounter tokens K. In our system, the Health Authority *HA* issues for each infected user a unique authentication code, a so-called Transaction Authentication Number (*TAN*). If an infected user wants to share their encounter tokens, it can use this *TAN* to prove its infection status by uploading the *TAN* along with their encounter token information.

Fig. 8 illustrates the infection verification and encounter token uploading phases. **In Step 1** and **Step 2** *HA* sends a TAN_i to infected user U_i. This can be done by using any appropriate out-of-band channel: in person, via SMS, via regular mail or via e-mail. TAN_i can also be sent along with the test results. The infected user can input their *TAN* directly by typing the number in or use their smartphone's camera to scan a QR code containing the *TAN*. **Step 3** shows how U_i can upload its encounter token information. Timestamp t_{ij}^l is encrypted using AES encryption using the encounter token k_{ij}^l as the key (or a key derivation function can be used to derive a key from k_{ij}^l). Let $m_{ij}^l = E_{k_{ij}^l}(t_{ij}^l)$ denote the encryption of t_{ij}^l. U_i sends TAN_i and a list L_{K_i} consisting of the m_{ij}^l along with corresponding hashes $h_{ij}^l = H(k_{ij}^l)$ of the encounter tokens k_{ij}^l to server *SP*. We have thus $L_{K_i} = \{\ldots, (m_{ij}^l, h_{ij}^l), \ldots\}$. *SP* forwards TAN_i to *HA* to verify whether TAN_i is valid or not. If TAN_i is valid, it will extract and store each element (m_{ij}^l, h_{ij}^l) of L_{K_i} separately.

It is worth noting that TRACECORONA provides both usability and privacy benefits by enabling infected users to remove specific unnecessary or sensitive encounter tokens that, e.g., (1) had only a short duration, thus being not essential for contracting the disease, or, (2) happened at a time or place that users do not want to disclose even anonymously, e.g., at a sensitive event or meeting.

4) Encounter Token Download: All TRACECORONA Apps download regularly, e.g., every night, encounter token information from server *SP* to identify potential exposure risks. Figure 9 shows the encounter download protocol. Let $L_k = \{\ldots, (H(k_{ij}^l), E_{k_{ij}^l}(t_{ij}^l)), \ldots\}$ be the list of the hashes and metadata of all encounter tokens of all infected users since the last update. To avoid linking entries related to a particular infected user together based on their position in the list, all entries in L_k are shuffled before sending them to users. Once a user U_j receives L_k, it compares the received token hashes to its own token hashes to discover matching encounters. If a matching encounter hash, e.g., $H(k_{ij}^l)$ is identified, U_j decrypts the matching encounter token metadata using the associated encounter token k_{ij}^l as the key: $t_{ij}^l = D_{k_{ji}^l}(E_{k_{ij}^l}(n_i||t_{ij}^l))$. U_j then checks the validity of encounter token w.r.t. to encounter time t_{ij}^l to make sure that k_{ij}^l and k_{ji}^l were established during the same time frame. This will limit the time-window available for a relay attack as we will discuss in Sect. VII-A3. Assuming that the clocks of the two devices are deviating by at most ϵ seconds, if $|t_{ij}^l - t_{ji}^l| \leq \epsilon$, k_{ij}^l and k_{ji}^l are considered to have been derived at the same time, i.e., the matching of k_{ij}^l and k_{ji}^l is valid. The system then uses metadata information, e.g., the time of the encounter t_{ij}^l, the duration of the encounter $D_{k_{ij}^l}$ and the signal strength S_{ij}^l to assess the risk of this exposure.

D. Hybrid Approach

In the following, we will present a hybrid approach that provides a trade-off between the effectiveness and the privacy requirements of centralized and decentralized architectures, i.e., maximizes effectiveness of the app while preserving privacy of the users. As discussed in Sect. III-A3, the accountability requirement (R-Ef3) refers to the possibility to evaluate the effectiveness of a DCT scheme. Therefore, we focus on this requirement by specifying what kind of data are needed to satisfy it and how they can be submitted to the health authority *HA* and the tracing service provider *SP*. Our design leverages the advantages of the DH-based scheme.

1) Useful data: To fulfill the requirement R-Ef3 (Accountability), the App needs to send authentic, but anonymized data in a privacy-preserving way to *SP*. Table IV shows potentially useful types of data that can help to evaluate and optimize the DCT system. Such types of data can also be helpful to epidemiologists and decision makers to understand the virus spreading patterns and, e.g, deploy effective policies to limit the pandemic.

[10]Exposure notification from a DCT is less important if users already knew their exposure status before being notified by a DCT app, e.g., the affected users who live in the same household to an infected users are expected to be informed immediately when the test result is available.

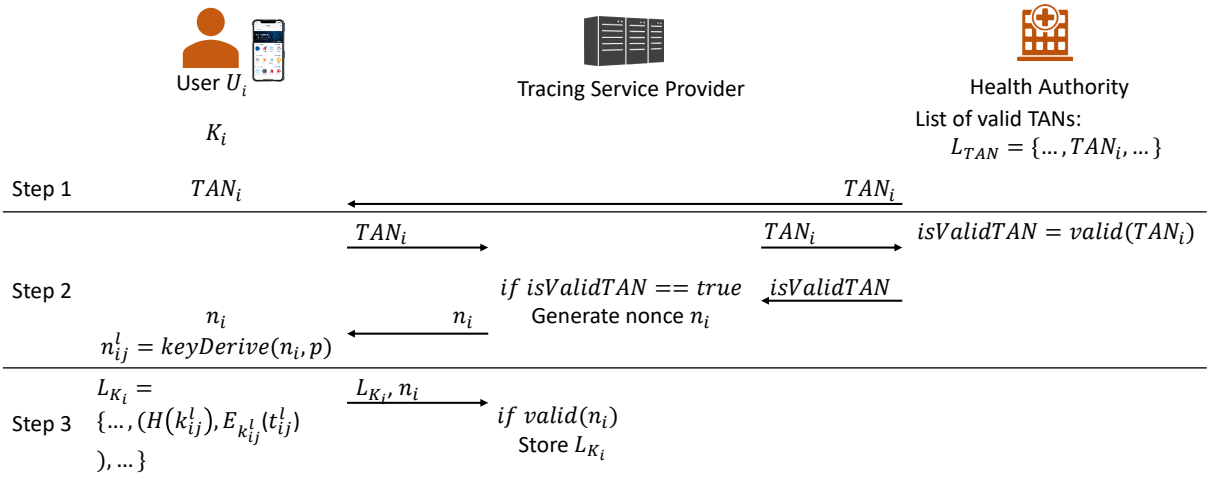
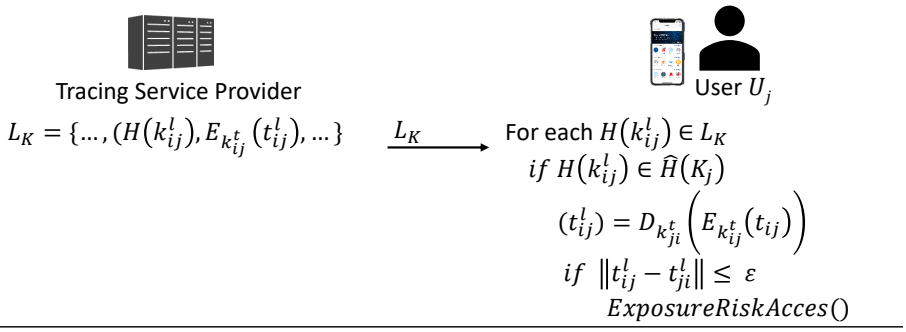


Fig. 8: Infection verification and encounter token upload.



L_K : The set of shuffled hashes and metadata of all encounter tokens of the infected users from the last update.

Fig. 9: Encounter Token Download and Exposure notification.

2) Further levels of contacts - early notifications: Only considering first-level contacts may not be enough. Affected users who were in contact with an infected user and thus may have gotten infected, may already spread the virus to other users (here referred to as second-level contacts) before ever being notified. Since there is a significant delay from the time that the direct (first-level) contact took place until the the direct contacts get an exposure notification and potentially a positive test result for themselves which is required to upload encounter tokens to warn second-level contacts, valuable time is lost, leading to potential new infection chains. Also, if direct contacts get infected but remain asymptomatic and do thus not get tested, second-level contacts may never receive exposure notifications in spite of the apparent risk of contagion.

Figure 10 shows an example of a typical scenario in which a user U_i gets infected on Day 0 and becomes contagious (spreading the virus) after around three days. However, it can take around four or five days until symptoms emerge. In the first year of the pandemic, it was typical in many countries, that it took around 2 days for a symptomatic user to be tested and another day (or even longer) to get the positive test result. Assuming that infected user U_i uploads their encounter information and the affected users receive exposure notification immediately, it still takes 5 days from the time when U_i starts spreading the virus until the affected users are notified. This means the affected users who got infected already can spread the virus for two to three days (from Day 6 to Day 8) to further second-level contacts. Therefore, to prevent potentially infected second-level contacts from spreading the virus, they should also be notified as early as possible.

In a DH-based system this can be easily achieved by allowing direct contacts U_j receiving exposure notifications to upload their encounter tokens even *before* receiving a positive test result in order to warn second-level contacts. To authenticate the uploaded tokens, the direct contact U_j only needs to provide the secret value of the encounter token ET_{ij} of its contact with the infected person U_i to *SP*, which can then verify it against the published hashed version of the token, and thus authenticate that a contact between the originator of the uploaded tokens (U_j) and an infected person (U_i) has in fact taken place. This prevents malicious users that are not direct contacts to claim fake exposures. However, encounter tokens concerning second-level contacts should be tagged by *SP* as such to distinguish them from ones concerning direct contacts, as the risk of contagion for second-level contacts likely is lower. Therefore, also any notifications posted by the contact tracing app to second-level contacts should clearly communicate that the warning concerns second-level contacts

TABLE III: The advantages of DH-based approaches in comparison to state-of-the-art approaches. (*) on the user side. (**) Possibly only infected users. (***) prevent one-way and limit real-time two-way attacks. +/- means achieve/not achieve corresponding requirements.

	Centralized	Decentralized		
	BlueTrace/ PEPP-PT/ TousAntiCovid	GAEN/ DP3T-1	DP3T-2/ MIT-PACT/ UW-PACT	TraceCORONA/ Pronto-C2/ CleverParrot
User identifier	Phone number /App ID	Random keys	Random keys	Random keys
Life-time of initial keys	Long-lived	Daily	Short-Lived	Short-lived
Superspreader	+	-*	-*	+*
CAII	+	-*	-*	+*
Identifying users	-	-**	+	+
Tracking users	-	-**	+	+
Extracting social graph	-	-**	+	+
Fake exposure claim	-	-	-	+
Relay attack	-	-	-	+***
Transparency	+	-	+	+
Independency	+	-	+	+

TABLE IV: Useful information for epidemiological analysis and evaluation and optimization of a DCT system.

Number of active users
Number of infected users
Number of encounters of infected users
Number of affected users
Number of encounters of affected users
True positive rate
Importance of notification [10]
Distribution of Risk score
The correlation between risk score and true positive rate

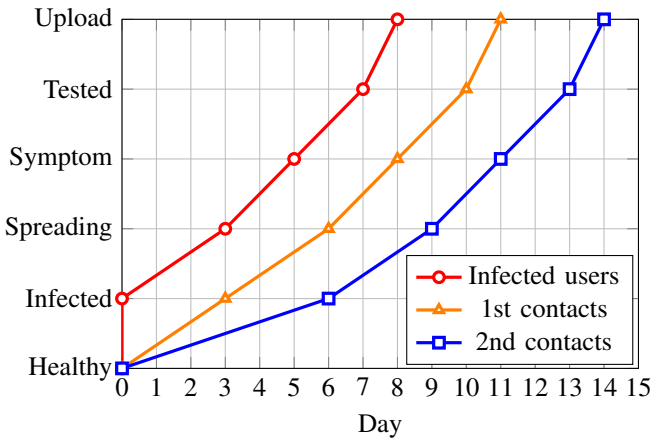


Fig. 10: An example of the development of statuses of infected users, their first-level contacts (1st contacts) and second-level contacts (2nd contacts).

and not direct exposure.

3) Sharing Epidemiological Information with Health Authorities: As discussed in the previous section, a direct contact U_j can prove it exposure status with an infected user (U_i) based on the possession of the secret value of the encounter token ET_{ij}. TRACECORONA utilizes this to authenticate the correctness of exposure information that users may voluntarily want to share with health care research institutions, thereby preventing malicious users from corrupting the data by providing faked exposure information to the researchers. This helps in improving the accuracy and correctness of the epidemiological modelling used as as basis for political decision making in the crisis situation.

4) Sharing Epidemiological Information via Healthcare Professionals: Since healthcare professionals like doctors collect information about their patients that come for a COVID-19 test or for consultation for their symptoms, doctors can act as a source of reliable information for epidemiological analysis in a properly anonymized form. For example, the healthcare professional could provide for each patient following anonymous information to help in assessing the epidemiological situation as well as the effectiveness of the contact tracing system: whether the user was notified by the contact tracing app and what the possible risk score was, whether the user knew about a potential exposure status even before being notified by the app, possible symptoms, and the test result. These kind of data provided to the epidemiological analysis do as such not reveal any information about the true identity of individual patients, but they do provide crucial information necessary to evaluate the effectiveness of the contact tracing app.

VII. Security and Privacy Analysis of TraceCORONA

A. Advantages of DH-based approaches

In this section, we will analyze DH-based approaches in general and TRACECORONA in particular in comparison to GAEN and BlueTrace with regard to requirements laid out in Sect. III.

1) Effectiveness: **Superspreader.** Although the Tracing Service Provider *SP* only receives anonymous encounter tokens that are not sufficient to detect Superspreaders and CAII users, the contact tracing App itself can be used to warn its user in case the App identifies a large number of contacts with other infected users, since this can be an indication that actually the user itself is a Superspreader or CAII who has been the source of contagion for those infections. As a result, the user could seek immediate testing, but also immediately upload their encounter tokens to warn others as discussed in

Sect. VI-D2. Further, the App can prove the user's status as a suspected Superspreader or CAII to *SP* by uploading the secret encounter tokens it has in common with infected users. By verifying these against the published hashes of encounter tokens of infected users the *SP* can verify that the user is indeed a person with many contacts with infected people and therefore a possible Superspreader. The *SP* can then tag the encounter tokens of the user accordingly, so that exposure notifications related these tokens can additionally be marked as being related to a 'possible superspreader' contact. Hence, requirement R-Ef2 related to the ability to identify Superspreaders can be successfully addressed.

2) Privacy: In DH-based systems, the ECDH public keys change every 15 minutes. This means that an eavesdropper adversary $\mathcal{A}^e$ cannot link public keys of a user, i.e., $\mathcal{A}^e$ can only track the movement of a user for less than 15 minutes, which is not enough to build informative movement profiles of the user.

Surveillance. Like other decentralized BLE systems, this attack fails against DH-based systems since the matching of contacts is done exclusively by the smartphone apps. A malicious service provider $\mathcal{A}^s$ does not benefit from learning the ETs of infected users since the uploaded encounter tokens do not reveal any information about the counterparts of those encounters.

Mass Surveillance. In TRACECORONA, even if a malicious service provider $\mathcal{A}^s$ colludes with an eavesdropper $\mathcal{A}^e$, the adversaries only get to know the hashes of encounter tokens of infected users and possible locations where $\mathcal{A}^e$ has collected them. However, as discussed in Sect. VI-B, since $\mathcal{A}^e$ can obtain ETs only through direct interaction with the monitored users and ETs are created only if encounters last for a specific time (e.g., 5 minutes), $\mathcal{A}^e$ is much more limited in its ability to obtain ETs associated with other users. In particular, $\mathcal{A}^e$ will be unable to establish *any* ETs with users that are just shortly passing by an eavesdropping station, so that the adversary's ability to track the movements of infected users is very limited. It is to be noted that this is a significant difference to both centralized and decentralized approaches presented in Sects. IV-A1 and IV-A2, since in these approaches the ability of the eavesdropping adversary $\mathcal{A}^e$ is in this sense unlimited and it can effectively sense the presence *all* users passing by its eavesdropping stations, even based on *one single observation* of the user.

On the other hand, in the basic set-up, a malicious service provider $\mathcal{A}^s$ acting as *SP* could link encounter tokens ET of a specific infected user, as they would be submitted in one transaction when they are uploaded to the service provider *SP*. However, this threat can be effectively mitigated by applying appropriate anonymization techniques like blind signatures with an anonymous postbox service [28] to the upload process of encounter tokens. This assures that even a malicious service provider $\mathcal{A}^s$ cannot link individual encounter tokens of infected users, thereby limiting the trackability of individual system users to relatively short time frames of, e.g., 15 minutes. By applying aforementioned defense techniques, TRACECORONA can therefore effectively address the requirements regarding providing protections against identifying (R-P1) and tracking (R-P2) users and extracting their social graphs (R-P3).

3) Security: In the following, we will explain how DH-based systems can mitigate current attacks, hence, fulfill the security requirements.

Fake exposure claim. DH-based systems can easily mitigate fake exposure claims (requirement R-S1). As mentioned in Sect. VI, infected users only share the hashes of encounter tokens meaning that the values of the encounter tokens themselves are always kept secret, so that only users actually participating in the encounter obtain the corresponding encounter token. Therefore, by proving possession of the (secret) encounter token, a user can prove that a contact with the counterpart has in fact taken place. The only way a dishonest user $\mathcal{A}^u$ can make fake exposure claims is to obtain access to the phones of users having matching encounter tokens and extracting them. However, this attack requires compromising individual devices one-by-one and can therefore not be easily scaled.

Relay/Replay Attack. Principally, all proximity-based approaches are vulnerable to this attack. This is a crucial to note that the main goal of this attack is to inject false exposure notifications on a large scale. However, DH-based systems provide two effective mitigation techniques that reduce the window of opportunity for attackers: on one hand, two-way communication is required for establishing contact tokens, prohibiting massive duplication of contact information by just copying beacon information, and, on the other hand, using limited time windows for validating contacts.

Two-way communication. In contrast to existing approaches [3], [19], [22], [24], [14] that are vulnerable to *one-way* relay attacks (cf. Sect. V), DH-based schemes utilize a *handshake* protocol requiring *two-way* communication to establish an encounter token. This means $\mathcal{A}^w$ cannot simply capture the beaconed information in one place and replay it in many other places like it would be possible in other schemes. $\mathcal{A}^w$ has to capture and relay messages at both places at the same time. This not only limits the time window of the attack but more importantly, it imposes a restriction on the scale of the attack since a mobile device cannot communicate with too many other devices at the same time, as Bluetooth LE provides only a limited number of channels and bandwidth. Based on our estimation, an average smartphone can only handle 8 Bluetooth LE connections simultaneously in a reliable manner. Therefore, in theory $\mathcal{A}^w$ can relay the handshake of one device to at most 8 other remote devices, while this number is not limited in other approaches.

Limited time window. In DH-based schemes, two users U_i and U_j in proximity of each other establish a unique secret encounter token ET_{ij}. An nfected user U_i can use ET_{ij} to encrypt any meta-data that only U_j can decrypt. Leveraging this property, in a DH-based scheme, e.g., TRACECORONA, the exact timestamp of an encounter can be encrypted and added to encounter token metadata so that user Apps checking encounter tokens can also check the exact encounter time. Therefore, only matching encounters that took place within a time window of at most ϵ seconds are considered as valid encounters, thereby limiting the window of opportunity for

relay attacks. Other decentralized schemes like [3], [19], [24] cannot impose such limitations on the timestamps of ephemeral IDs, because the involved tracing apps can not mutually verify the actual time point of when contacts take place due to the fact that only one-way communication is used.

Due to this, the GAEN API [3] allows a two-hour time window for synchronizing *RPI*, i.e., $\mathcal{A}^w$ can have up to two hours to conduct relay attacks (cf. Sect. V-B3). In DH-based schemes, this ϵ could be limited to seconds when assuming that smartphones used for contact tracing apps can sync their clocks via an Internet connection or during the exchange of the public keys. Note that all contact tracing apps need a frequent Internet connection for uploading and downloading encounter information.

Thus, the combination of these two advantages, requirement of two-way communication and small time window help DH-based schemes such as TRACECORONA to significantly reduce the impact of relay attacks on the system.

4) Ethics: Like BlueTrace, DH-based systems like TRACECORONA can be implemented with complete access to the source code, guaranteeing transparency. It is a standalone app that does not depend on any built-in contact tracing APIs running deep inside the mobile operating systems such as Android or iOS, thus satisfying requirements with regard to transparency and (R-Et1) and independence (R-Et2). This is in stark contrast to proprietary and closed GAEN systems strictly enforced by Google and Apple. Especially in Apple's iOS systems independent contact tracing applications that continuously need to use BLE in the background are blocked by the operating system so that effective BLE sensing as required by contact tracing apps is in practice not possible. Instead, Apple forces all contact tracing approaches to rely on their closed and proprietary GAEN API whose functionality can not be independently examined nor verified. It is therefore highly debatable, whether this approach is ethical, as Apple in fact forces users into using their GAEN solution, having to involuntarily accept all the deficiencies discussed in this paper, or, refrain from using contact tracing solutions at all.

B. Summary of Benefits of DH-based Approaches and Comparison to Other Approaches

We summarize key differences and security and privacy advantages of DH-based systems in comparison to existing approaches in Tab. III. Further, for a better overview, we illustrate the comparisons in Fig. 11. It shows that GAEN collects lower scores among all the alternatives. The DH-based systems provide better security and privacy protection than all other discussed solutions. For example, DH-based approaches are resilient to fake exposure claim attacks and wormhole adversary (i.e., narrowing the attack window time and requiring more communication effort as the adversary would have to operate real-time two-way communication relays). Moreover, comparing to the most widely spread contact tracing framework by Apple and Google, which is vulnerable to profiling attacks as the adversary can track the movements of infected users, DH-based systems guarantee a better protection. Interesting but not surprisingly, BlueTrace is the best w.r.t to fulfilling effectiveness requirements since it can potentially detect Superspreader and CAII and provide useful data to epidemiologists while this could be challenging to other approaches. In terms of ethics, GAEN again is on the lower end because it received many criticisms due to their coercion and the lack of transparency. More importantly, our hybrid approach inherits the advantages of DH-based approaches in terms of security and ethical aspects, while being on par with centralized approaches with regard to effectiveness.

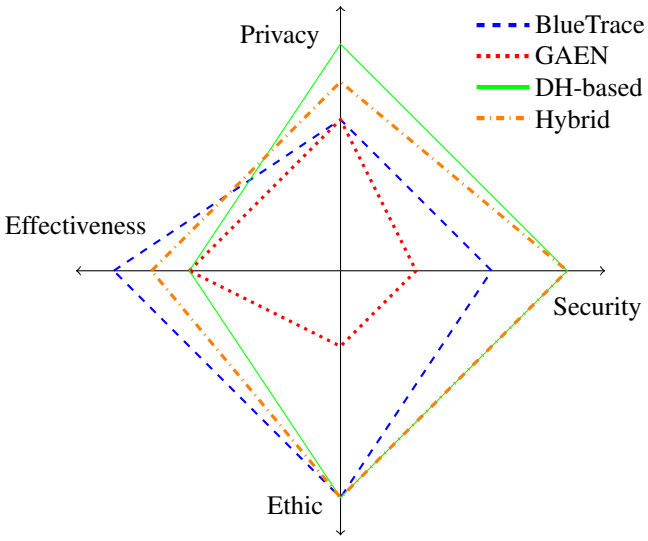


Fig. 11: Quadrilemma comparison of prominent contact tracing scheme.

VIII. IMPLEMENTATION AND BETA TEST

We prototyped TRACECORONA for the Android smartphone platform and tested it in a public beta test. The implementation uses Kotlin, an emerging programming language for Android. We have not implemented TRACECORONA on iOS because it does not allow apps to use Bluetooth communication in the background. We use the native Android Bluetooth Low Energy APIs to implement the Encounter Token Establishment protocol. Further, our cryptographic functions, e.g., ECDH are based on the Bouncy Castle library. For the server acting as *SP*, the code is written in Java and run on Ubuntu Server operating system. In principle, our app can run on any Android smartphone that supports Bluetooth LE, i.e., Android 5.0 and later.

Alpha testing. We internally tested the app with 25 devices covering various models and manufacturers[11]. The results show that our app works without any problems and consumes 5-8 % battery for a whole day (24 hours) of contact tracing without further optimizations. It is worth noting that, for the apps based on Bluetooth proximity detection, the main power consumption emerges from advertising and scanning for other devices. Therefore, our approach would consume more or less the same amount of battery as state-of-the-art approaches like BlueTrace [14] or GAEN [3] if the time windows and

[11]https://tracecorona.net/list-of-device-models-on-which-tracecorona-has-been-testedhttps-tracecorona-net-download-tracecorona/

frequencies of advertising and scanning processes are set the similar.

Beta test. We published the TRACECORONA app on our website and interestingly the app has drawn a lot of attention[12]. Indeed, more than 2000 users have downloaded and tested the app. We have received many positive feedbacks on the app features and performance, except received criticism that the app does not work on very old devices that do not support Bluetooth LE. However, this is a technical limitation that is out of our control.

Implementation on wearable devices. To demonstrate the possibility of deploying TRACECORONA even on on wearable devices like wristbands (For a full description please refer to Sect. III in the supplementary materials), we have implemented our design on Adafruit HUZZAH32 (ESP32), a MCU developer board that costs about US $20. Our program size (including libraries) is 1.12 MB. Our evaluation shows that the device needs only 666 milliseconds to synchronize all the keys with the phone for a whole day. It can exchange Ephemeral Identification EI and Ephemeral ECDH Public Key Q with multiple other ESP32 devices and smartphones simultaneously without any delay compared to the communication between smartphones.

IX. RELATED WORK

A. DH-based approaches

PRONTO-C2 [25] The main problem of DH-based approach is that the size of the public key might exceed the space limit of BLE advertising messages (i.e., identifier beacons). The minimum requirement for a standardized ECDH key is 256 bits (or even 384 bits tp provide security against a powerful adversary) while in a typical BLE advertising message there is space for 128 bits only. PRONTO-C2's authors propose to store the public keys on a bulletin board that can be maintained by the *SP* or can be decentralized, and even implemented with a blockchain. Hence, instead of broadcasting the public keys via BLE, the devices only beacon the references (i.e., addresses) of the keys in the bulletin. When a user is infected, a cryptographic hash of encounter tokens is uploaded to the bulletin board. As discussed in Sect. VI-B, TRACECORONA solves this problem by utilizing BLE connections to transfer public keys without any data restrictions.

CleverParrot [26] To deal with the issue of fitting a DH public key in a BLE advertising message, CleverParrot proposes using a minimum key size of 224 bits (28 bytes) based on the elliptic curve P-224. They choose this key size since it is the same as the one use in Apple's Find My protocol[13]. However, it is worth noting that Apple's Find My protocol is a special function in iOS. In fact, both Android and iOS support only 128-bit BLE advertising messages. Therefore, CleverParrot cannot be implemented in practice unless Google and Apple change their BLE platform or they have to adopt and treat CleverParrot as a special function like Apple's Find My.

DH with Private Set Intersection Cardinality (PSI-CA) Epione [27] leverages Function Secret Sharing (FSS) techniques [41] to prevent other users from learning information about the encounter tokens uploaded by infected users. In particular, this approach enables clients (user Apps) in collaboration with the servers *SP* to learn matching encounter tokens, i.e., U_j can know how many encounters with infected users it has without downloading these encounters as shown in **Step 3** Fig. 5.

B. Survey on existing DCT schemes, apps and challenges

There are a number of works that survey existing DCT schemes, apps and challenges. Those works can be categorized into two groups: (i) discussing technical specifications, operations and issues of the rolled out apps [42], [43] and (ii) studying certain aspects of some DCT schemes [44], [29].

Sun et al. [42] focus on investigating the security and privacy issues of DCT apps on Android. Wen et al. [43] vet privacy issues of 41 country apps that have rolled our worldwide, in which they focus on analysis of documentation but also binary code to figure out what data an app collects and discuss the potential privacy risks.

Unlike those works that focus on the apps, Vaudenay et al. [29] focus on investigating the security and privacy issues of several schemes along with their architectures. The most relevent to our work is the study provided by Ahmed et al. [44]. They discuss 8 different potential attacks on 12 country apps divided in three groups: centralized, decentralized and hybrid architectures. However, those works do not provide an abstraction that groups evaluation requirements of similar schemes as we do in our work.

While existing works point out a number of privacy problems of GAEN (cf. Sect. V-B2), Ahmed et al. claim that GAEN protects privacy of users and criticize that existing attacks are unrealistic [45]. However, they do not provide arguments and evidence for their claim, i.e., it is not clear how GAEN can defend against such attacks. In fact, their main experiments only confirm the principal design requirements of GAEN like Randomness of Bluetooth addresses or RPI intervals that are also included in existing attack models [5], [19], [25], [21]. Unfortunately, the paper also gives some misleading information. For example, it states that: "in normal operation, the TEK downloaded are not readily available to the user and the exposure assessment is done away from the user." However, the uploaded TEK keys of infected users are in fact by design public information that is accessible to any moderately sophisticated adversary[14]. We summarize existing works on analyzing DCT in Tab. VIII (Appendix E).

X. CONCLUSION

In our work we have considered existing digital contact tracing DCT architectures, schemes, technologies and apps.

[12]https://tracecorona.net/download-tracecorona/

[13]"Find My overview", Apple, https://support.apple.com/guide/security/locating-missing-devices-sece994d0126/1/web/1.

[14]An archive collecting *TEK*s of the German Contact Tracing App: https://ctt.pfstr.de/

We provide a systematic analysis and comparison of existing approaches based on a natural quadrilemma (i.e., obtaining simultaneously effectiveness, security, privacy while taking ethical aspects into account) in digital contact tracing. Our study shows that GAEN which is adopted by many EU countries and states in the USA, unfortunately is less effective and worse in protecting sensitive data compared to other approaches like BlueTrace and DH-based schemes like ProntoC2 and CleverParrot. We propose TRACECORONA that addresses security and privacy challenges of existing contact tracing approaches while providing comparable effectiveness. In contrast to state-of-the-art approaches that are based on exchanging ephemeral IDs, TRACECORONA allows users to anonymously establish encounter-specific tokens using short-range wireless communication like Bluetooth. The encounter tokens can be later on used to warn users of potential exposure risks with infected persons. We systematically and extensively analyze the security and privacy of TRACECORONA in comparison to existing approaches in Sect. VI to show that TRACECORONA is resilient to various attacks and thus provides better security and privacy guarantees than other approaches. We have implemented TRACECORONA and tested it with 25 different devices of various brands and models. Further, we have published a beta test version of TRACECORONA that has been downloaded and used by more than 2000 users without any major functional problems demonstrating that TRACECORONA is practical.

REFERENCES

[1] L. Ferretti, C. Wymant, M. Kendall, L. Zhao, A. Nurtay, L. Abeler-Doerner, M. Parker, D. Bonsall, and C. Fraser, "Quantifying sars-cov-2 transmission suggests epidemic control with digital contact tracing," *Science*, 2020.

[2] C. Wymant, L. Ferretti, D. Tsallis, M. Charalambides, L. Abeler-Dörner, D. B. R. Hinch, M. Kendall, L. Milsom, M. Ayres, C. Holmes, M. Briers, and C. Fraser, "The epidemiological impact of the nhs covid-19 app." *Nature*, vol. 594, no. 4, 2021. [Online]. Available: https://doi.org/10.1038/s41586-021-03606-z

[3] Apple and Google, "Exposure Notification: Cryptography Specification, v1.2," April 2020, https://www.apple.com/covid19/contacttracing.

[4] Y. Gvili, "Security Analysis of the COVID-19 Contact Tracing Specifications by Apple Inc. and Google Inc." Cryptology ePrint Archive, Report 2020/428, April 2020, https://eprint.iacr.org/2020/428.

[5] L. Baumgärtner, A. Dmitrienko, B. Freisleben, A. Gruler, J. Höchst, J. Kühlberg, M. Mezini, M. Miettinen, A. Muhamedagic, T. D. Nguyen *et al.*, "Mind the gap: Security & privacy risks of contact tracing apps," in *19th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, 2020.

[6] V. Iovino, S. Vaudenay, and M. Vuagnoux, "On the effectiveness of time travel to inject covid-19 alerts," *The Cryptographer's Track at the RSA Conference, CT-RSA2021*, 2021, https://eprint.iacr.org/2020/1393.

[7] G. Avitabile, D. Friolo, and I. Visconti, "Tenk-u: Terrorist attacks for fake exposure notifications in contact tracing systems," *19th International Conference on Applied Cryptography and Network Security, ACNS2021*, 2021, https://eprint.iacr.org/2020/1150.

[8] L. White and P. van Basshuysen, "Without a trace: Why did corona apps fail?" *Journal of Medical Ethics*, 2021. [Online]. Available: https://jme.bmj.com/content/early/2021/01/08/medethics-2020-107061

[9] M. Lanzing, "Contact tracing apps: an ethical roadmap," 2020, https://doi.org/10.1007/s10676-020-09548-w.

[10] B. Pinkas and E. Roneny, "Hashomer: A proposal for a privacy-preserving bluetooth based contact tracing scheme for hamagen," 2020, https://github.com/eyalr0/HashomerCryptoRef.

[11] D. Leith and S. Farrell, "Contact tracing app privacy: What data is shared by europe's gaen contact tracing apps," https://www.scss.tcd.ie/Doug.Leith/pubs/contact_tracing_app_traffic.pdf.

[12] "Luca app," 2021, https://www.luca-app.de/.

[13] T. Stadler, W. Lueks, K. Kohls, and C. Troncoso, "Preliminary analysis of potential harms in the luca tracing system," 2021.

[14] J. Bay, J. Kek, A. Tan, C. S. Hau, L. Yongquan, J. Tan, and T. A. Quy, "Bluetrace: A privacy-preserving protocol for community-driven contact tracing across borders," Apr. 2020. [Online]. Available: https://bluetrace.io/static/bluetrace_whitepaper-938063656596c104632def383eb33b3c.pdf

[15] D. J. Leith and S. Farrell, "Measurement-based evaluation of google/apple exposure notification api for proximity detection in a light-rail tram," *PLOS ONE*, vol. 15, no. 9, pp. 1–16, 09 2020. [Online]. Available: https://doi.org/10.1371/journal.pone.0239943

[16] T. Istomin, E. Leoni, D. Molteni, A. L. Murphy, G. P. Picco, and M. Griva, "Janus: Efficient and accurate dual-radio social contact detection," 2021.

[17] J. Meklenburg, M. Specter, M. Wentz, H. Balakrishnan, A. Chandrakasan, J. Cohn, G. Hatke, L. Ivers, R. Rivest, G. J. Sussman, and D. Weitzner, "Sonicpact: An ultrasonic ranging method for the private automated contact tracing (pact) protocol," https://pact.mit.edu/wp-content/uploads/2020/11/SonicPACT_Final_v2-with-logos-revA.pdf.

[18] A. Trivedi and D. Vasisht, "Digital contact tracing: Technologies, shortcomings, and the path forward," *SIGCOMM Comput. Commun. Rev.*, vol. 50, no. 4, p. 75–81, Oct. 2020. [Online]. Available: https://doi.org/10.1145/3431832.3431841

[19] C. Troncoso, M. Payer, J. Hubaux, M. Salathé, J. R. Larus, E. Bugnion, W. Lueks, T. Stadler, A. Pyrgelis, D. Antonioli, L. Barman, S. Chatel, K. G. Paterson, S. Capkun, D. A. Basin, J. Beutel, D. Jackson, M. Roeschlin, P. Leu, B. Preneel, N. P. Smart, A. Abidin, S. F. Gürses, M. Veale, C. Cremers, M. Backes, N. O. Tippenhauer, R. Binns, C. Cattuto, A. Barrat, D. Fiore, M. Barbosa, R. Oliveira, and J. Pereira, "Decentralized privacy-preserving proximity tracing," *CoRR*, vol. abs/2005.12273, 2020. [Online]. Available: https://arxiv.org/abs/2005.12273

[20] L. Reichert, S. Brack, and B. Scheuermann, "Lighthouses: A warning system for super-spreader events," Cryptology ePrint Archive, Report 2020/1473, 2020, https://eprint.iacr.org/2020/1473.

[21] S. Vaudenay, "Analysis of DP-3T," Cryptology ePrint Archive, Report 2020/399, April 2020. [Online]. Available: https://eprint.iacr.org/2020/399

[22] PEPP-PT, "pepp-pt," 2020. [Online]. Available: https://www.pepp-pt.org/content

[23] A. Boutet, C. Castelluccia, M. Cunche, V. Roca, A. Baud, P.-G. Raverdy, and C. Lauradoux., "Desire: Leveraging the best of centralized and decentralized contact tracing systems," ACM Digital Threats: Research and Practice, Special Issue on Security and Privacy for Covid-19, 2021., 2021.

[24] R. L. Rivest *et al.*, "The pact protocol specification," 2020, https://pact.mit.edu/wp-content/uploads/2020/11/The-PACT-protocol-specification-2020.pdf.

[25] G. Avitabile, V. Botta, V. Iovino, and I. Visconti, "Towards defeating mass surveillance and sars-cov-2: The pronto-c2 fully decentralized automatic contact tracing system," CoronaDef Workshop at NDSS 2021, 2021, https://www.ndss-symposium.org/ndss-paper/auto-draft-164/.

[26] R. Canetti, Y. T. Kalai, A. Lysyanskaya, R. L. Rivest, A. Shamir, E. Shen, A. Trachtenberg, M. Varia, and D. J. Weitzner, "Privacy-preserving automated exposure notification," Cryptology ePrint Archive, Report 2020/863, 2020, https://eprint.iacr.org/2020/863.

[27] N. Trieu, K. Shehata, P. Saxena, R. Shokri, and D. Song, "Epione: Lightweight contact tracing with strong privacy," 2020.

[28] L. Reichert, S. Brack, and B. Scheuermann, "Ovid: Message-based automatic contact tracing," CoronaDef at NDSS 2021, 2021, https://www.ndss-symposium.org/ndss-paper/auto-draft-165/.

[29] S. Vaudenay, "Centralized or decentralized? the contact tracing dilemma," Cryptology ePrint Archive, Report 2020/531, 05 2020, https://eprint.iacr.org/2020/531.

[30] "Security and privacy analysis of the document 'pepp-pt: Data protection and information security architecture," DP-3T project, Apr. 2020. [Online]. Available: https://github.com/DP-3T/documents/blob/master/Security%20analysis/PEPP-PT_%20Data%20Protection%20Architechture%20-%20Security%20and%20privacy%20analysis.pdf

[31] Z. Brighton-Knight, J. Mussared, and A. Tiu, "Linkability of rolling proximity identifiers in google's implementation of the exposure notification system," Technical report, https://github.com/alwentiu/contact-tracing-research/blob/main/GAEN.pdf.

[32] G. Avitabile, V. Botta, V. Iovino, and I. Visconti, "Towards defeating mass surveillance and sars-cov-2: The pronto-c2 fully decentralized

automatic contact tracing system," Cryptology ePrint Archive, Report 2020/493, 2020, https://eprint.iacr.org/2020/493.
[33] L. White and P. van Basshuysen, "Privacy versus public health? a reassessment of centralised and decentralised digital contact tracing," in *Science and Engineering Ethics*, 2021, https://doi.org/10.1007/s11948-021-00301-0.
[34] S. Vaudenay, "The dark side of swisscovid," 2020, https://lasec.epfl.ch/people/vaudenay/swisscovid.html.
[35] J.-H. Hoepman, "A critique of the google apple exposure notification (gaen) framework," 2021.
[36] J. Reardon, "Why google should stop logging contact-tracing data," blog.appcensus.io, 2021, https://blog.appcensus.io/2021/04/27/why-google-should-stop-logging-contact-tracing-data/.
[37] "Replay attack "in the past"," 2020, https://github.com/immuni-app/immuni-app-android/issues/278.
[38] R. Gennaro, A. Krellenstein, and J. Krellenstein, "Exposure notification system may allow for large-scale voter suppression," 2020, https://static1.squarespace.com/static/5e937afbfd7a75746167b39c/t/5f47a87e58d3de0db3da91b2/1598531714869/Exposure_Notification.pdf.
[39] P.-O. Dehaye and J. Reardon, "Swisscovid: a critical analysis of risk assessment by swiss authorities," 2020, https://arxiv.org/abs/2006.10719.
[40] D. Telekom and SAP, "Corona-Warn-App - The Official COVID-19 Exposure Notification App for Germany," https://github.com/corona-warn-app.
[41] E. Boyle, N. Gilboa, and Y. Ishai, "Function secret sharing," in *Advances in Cryptology - EUROCRYPT 2015*, E. Oswald and M. Fischlin, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 337–367.
[42] R. Sun, W. Wang, M. Xue, G. Tyson, S. Camtepe, and D. C. Ranasinghe, "An empirical assessment of global covid-19 contact tracing applications," 2021.
[43] H. Wen, Q. Zhao, Z. Lin, D. Xuan, and N. Shroff, "A study of the privacy of covid-19 contact tracing apps," in *Security and Privacy in Communication Networks*, N. Park, K. Sun, S. Foresti, K. Butler, and N. Saxena, Eds. Cham: Springer International Publishing, 2020, pp. 297–317.
[44] N. Ahmed, R. A. Michelin, W. Xue, S. Ruj, R. Malaney, S. S. Kanhere, A. Seneviratne, W. Hu, H. Janicke, and S. K. Jha, "A survey of covid-19 contact tracing apps," *IEEE Access*, vol. 8, pp. 134 577–134 601, 2020.
[45] S. Ahmed, Y. Xiao, C. Fung, M. Yung *et al.*, "Privacy guarantees of ble contact tracing: A case study on covidwise," *arXiv preprint arXiv:2111.08842*, 2021.
[46] "Tousanticovid," Government of France, 2020, https://solidarites-sante.gouv.fr/soins-et-maladies/maladies/maladies-infectieuses/coronavirus/tousanticovid.
[47] "Covidradar," covidradar.mx, 2020, https://covidradar.mx/.
[48] "Virusradar," virusradar.hu, 2020, https://virusradar.hu/.
[49] "Bluezone," bluezone.gov.vn, 2020, https://bluezone.gov.vn/.
[50] "Rakning c-19," www.covid.is, 2020, https://www.covid.is/app/en.
[51] "Safe paths," safepaths.mit.edu, 2020, https://safepaths.mit.edu/.
[52] "Tawakkalna," ta.sdaia.gov.sa, 2020, https://ta.sdaia.gov.sa/.
[53] "Virusafe," coronavirus.bg, 2020, https://app.coronavirus.bg/.
[54] "Shlonik," Kuwait Central Agency for Information Technology-Health & Fitness, 2020, https://play.google.com/store/apps/details?id=com.healthcarekw.app&hl=en_US&gl=US.
[55] "Aarogya Setu Mobile App," Government of India, https://www.mygov.in/aarogya-setu-app/.
[56] "Beaware bahrain," iga.gov.bh, 2020, https://bahrain.bh/.
[57] "Ehteraz," acta.gov.qa, 2020, https://www.acta.gov.qa/en/ehteraz/.
[58] "Morchana," Digital Government Development Agency, Thailand, 2020, https://www.dga.or.th/.
[59] "Pedulilindungi," Ministry of communication and informatics, Indonesia, 2020, https://www.pedulilindungi.id/.
[60] "In coronavirus fight, china gives citizens a color code, with red flags," nytimes.com, 2020, https://www.nytimes.com/2020/03/01/business/china-coronavirus-surveillance.html.
[61] J.-H. Hoepman, "Hansel and gretel and the virus: Privacy conscious contact tracing," 2021.
[62] F. Buccafurri, V. De Angelis, and C. Labrini, "A privacy-preserving solution for proximity tracing avoiding identifier exchanging," in *2020 International Conference on Cyberworlds (CW)*, 2020, pp. 235–242.
[63] J. Chan, D. Foster, S. Gollakota, E. Horvitz, J. Jaeger, S. Kakade, T. Kohno, J. Langford, J. Larson, P. Sharma, S. Singanamalla, J. Sunshine, and S. Tessaro, "Pact: Privacy sensitive protocols and mechanisms for mobile contact tracing," 2020.
[64] "South korea to step-up online coronavirus tracking," smartcitiesworld.net, 2020, https://www.smartcitiesworld.net/news/news/south-korea-to-step-up-online-coronavirus-tracking-5109.
[65] M. o. H. Israel govement, "Hamagen," 2020, https://govextra.gov.il/ministry-of-health/hamagen-app/download-en/.
[66] "Tracecorona: Anonymous decentralized contact tracing for pandemic response," TraceCORONA, May 2020, tracecorona.net.
[67] S. Dittmer, Y. Ishai, S. Lu, R. Ostrovsky, M. Elsabagh, N. Kiourtis, B. Schulte, and A. Stavrou, "Function secret sharing for psi-ca: With applications to private contact tracing," Cryptology ePrint Archive, Report 2020/1599, 2020, https://eprint.iacr.org/2020/1599.
[68] H. R. Hasan, K. Salah, R. Jayaraman, I. Yaqoob, M. Omar, and S. Ellahham, "Covid-19 contact tracing using blockchain," *IEEE Access*, vol. 9, pp. 62 956–62 971, 2021.
[69] A. Berke, M. A. Bakker, P. Vepakomma, R. Raskar, K. Larson, and A. S. Pentland, "Assessing disease exposure risk with location histories and protecting privacy: A cryptographic approach in response to A global pandemic," *CoRR*, vol. abs/2003.14412, 2020. [Online]. Available: https://arxiv.org/abs/2003.14412
[70] "Maximizing privacy and effectiveness in covid-19 apps," OpenMined, 2020, https://blog.openmined.org/covid-app-privacy-advice/.
[71] X. Cheng, H. Yang, A. S. Krishnan, P. Schaumont, and Y. Yang, "KHOVID: interoperable privacy preserving digital contact tracing," *CoRR*, vol. abs/2012.09375, 2020. [Online]. Available: https://arxiv.org/abs/2012.09375
[72] Salesforce, "Track employee health-related interactions safely and securely," 2020, https://www.salesforce.com/products/contact-tracing/overview/.
[73] IBM, "Watson works: A safer way to return to the workplace," 2020, https://www.ibm.com/watson/watson-works?lnk=hpmcov_bs&lnk2=learn.
[74] A. Boutet, C. Castelluccia, M. Cunche, A. Dmitrienko, V. Iovino, M. Miettinen, T. D. Nguyen, V. Roca, A.-R. Sadeghi, S. Vaudenay, I. Visconti, , and M. Vuagnoux, "Contact tracing by giant data collectors: Opening pandora's box of threats to privacy, sovereignty and national security," 2020, https://tracecorona.net/wp-content/uploads/2020/12/Digital_Contact_Tracing.pdf.
[75] A. Crocker, K. Opsahl, and B. Cyphers, "The Challenge of Proximity Apps For COVID-19 Contact Tracing," 2020. [Online]. Available: https://www.eff.org/de/deeplinks/2020/04/challenge-proximity-apps-covid-19-contact-tracing
[76] N. Danz, O. Derwisch, A. Lehmann, W. Puenter, M. Stolle, and J. Ziemann, "Security and privacy of decentralized cryptographic contact tracing," Cryptology ePrint Archive, Report 2020/1309, 2020, https://eprint.iacr.org/2020/1309.
[77] S. Kojaku, L. Hébert-Dufresne, E. Mones, S. Lehmann, and Y.-Y. Ahn, "The effectiveness of backward contact tracing in networks," 2020.
[78] D. J. Leith and S. Farrell, "Coronavirus contact tracing: Evaluating the potential of using bluetooth received signal strength for proximity detection," 2020.
[79] S. Del Valle, J. Hyman, H. Hethcote, and S. Eubank, "Mixing patterns between age groups in social networks," *Social Networks*, vol. 29, no. 4, pp. 539–554, 2007. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0378873307000330

BIOGRAPHY

Thien Duc Nguyen is a research assistant and a PhD candidate at the System Security Lab of Technical University of Darmstadt (TU Darmstadt), Germany. He is currently working in various topics related to machine learning-based security mechanisms for IoT, security for federated machine learning, contextual security and digital contact tracing.

Markus Miettinen is a postdoctoral researcher at the System Security Lab of TU Darmstadt, Germany. He graduated as Dr.-Ing. from TU Darmstadt in 2018. Before joining academia, he acquired more than a decade of professional experience in industrial research at the Nokia Research Center in Helsinki, Finland and Lausanne, Switzerland. His research interests lie in utilizing machine learning methods for realizing autonomous security solutions for IoT and

mobile computing environments.

Alexandra Dmitrienko is an associate professor and the head of the Secure Software Systems group at the University of Wuerzburg in Germany. She holds a PhD degree in Security and Information Technology from TU Darmstadt (2015). She received numerous awards for her research, including Intel Doctoral Student Honor Award (2013) and ERCIM STM WG Award (2016). Her research interests focus on secure software engineering, systems security and privacy, and security and privacy of mobile, cyber-physical, and distributed systems.

Ahmad-Reza Sadeghi is a full professor for Computer Science at TU Darmstadt, where he directs the System Security Lab, the Intel Private AI Center, and Open S3 Lab. He received his PhD from the University of Saarland. He has served on the editorial board of ACM TISSEC and as Editor-in-Chief for IEEE Security and Privacy Magazine.

Ivan Visconti is a full professor of Computer Science in the Computer and Electrical Engineering and Applied Mathematics Department of the University of Salerno. His research interests focus mainly on designing provably secure and privacy-preserving cryptographic protocols and securing blockchains and their applications. He served for two years as Senior Area Editor for the journal IEEE Transactions on Information Forensics and Security. Several of his results have been published in the most competitive conferences in cryptography (i.e., CRYPTO, EUROCRYPT, TCC).

APPENDIX

A. List of Country Apps

List of centralized approaches. Table V shows an overview of prominent centralized contact tracing solutions. In the table, the apps are categorized by employed technology, e.g., notification method and type of data collected. Further, adoption information is also provided, from which one can see that centralized-based approaches are more preferred in Asian countries than on other continents. In terms of technologies, besides Bluetooth, location is also widely used in many countries. Noticeably, many if not most deployed apps, e.g., [14], [47], [55] collect sensitive data, e.g., phone numbers, names, ages of the users.

It is worth noting that some schemes have been proposed to improve the security and privacy of centralized approaches [61], [16]. For example, Hoepman et al. [61] propose two centralized protocols that aim to reduce the risk of tracking users' locations and replay attacks. The first handshake (peer-to-peer) protocol establishes logs of encounters. In particular, U_i broadcasts a public key, and U_j in the vicinity will respond with U_j's ID encrypted by U_i's public key and vice-versa. However, this peer-to-peer protocol fails if either of the two-way messages is lost. Therefore, Hoepman et al. introduce a second approach that utilizes a central server to tackle this issue so that devices only need to broadcast a random public key. The responses with encrypted ID information will then be sent via the server, i.e., do not need to be sent directly via the BLE channel.

List of decentralized approaches. Table VI shows an overview of prominent solutions. It shows that most approaches use Bluetooth, some use location while a very few approaches using other technologies, e.g., [16] using UWB and [17] using ultrasound. Further, the figure shows that GAEN is widely used in Western Europe and North America while South Korea and Israel use their own location-based approaches. Unfortunately, other decentralized approaches that claim to have better security and privacy guarantees in comparison to GAEN, e.g., Pronto-C2[25], ClerverParrot[26], and Epione [27] have not been adopted yet. In terms of data collection, decentralized apps do not collect personal data of users like phone numbers, email addresses or names. However, the system (consisting of *SP* and Apps) collects encounter information of infected users and additionally, the apps also record information about encounters with other users in the vicinity. Such encounter information can be generally divided into two categories based on which cryptographic approach is used. Apps based on the use of symmetric cryptography collect temporary IDs (*TempID*s) of other users and match these against IDs that they derive from the temporary exposure keys (*TEK*s) of all infected users they download from *SP*. In contrast, apps based on asymmetric cryptography establish encounter-specific cryptographic tokens for each encounter. To enable identification of at-risk encounters, infected users upload hashed values of their encounter tokens to *SP*, from where all other users download them and compare them to the hashed versions of their collected encounter tokens. In case there is a match, this is an indication that an at-risk encounter with an infected person has taken place.

B. Advanced Privacy Techniques

There are several (mainly cryptography-based) privacy techniques that have been considered to enhance privacy of DCT. Table VII shows an overview of such approaches. Unfortunately, none of these contact tracing approaches have been used in practice yet. Next, we will elaborate on these approaches.

Blind Signature. Some approaches, e.g., [28], [25] propose using a blind signature scheme to verify the authenticity of encounter information, e.g., the temporary keys that infected users upload to the tracing service provider *SP*. In contrast to common approaches in which *HA* gives a unique transaction number (TAN) to each of the infected users for uploading their data, allowing *HA* to link infected users in a DCT to *HA*'s patient data, blind signature-based approaches prevent this by enabling *HA* to "blindly" sign the temporary keys of infected users without knowing the actual content. Therefore, neither *HA* nor any other party can know based on verification information to which user of the system published temporary keys belong.

Blockchain. Instead of using a centralized server to collect and forward encounter information of infected users to other users, some approaches by, e.g., Avitabile et al. [25] or Hasan et al. [68] leverage a blockchain to decentralize the process of publishing, verifying, and matching encounter information. This makes the system transparent to users in that it does not require a trusted server.

TABLE V: List of centralized contact tracing approaches and their adoption. *) Contacts are anonymous in France and UK systems; **) Not in France and UK systems; ***) Location can be any source used to locate a user, e.g., GPS coordinates captured by the user's phones or the address of a shop that the user has made a credit card transaction. Personal data can be name, address, gender, or age, etc.

Approach	Tech	Notification	Data Collected by Server	Data Collected by Client	Adoption
BlueTrace [14], PEPP-PT [22], TousAntiCovid [46], CovidRada [47], E7mi, VirusRadar [48], BlueZone [49]	Bluetooth	Server via App/SMS, or the Health Authority via telephone	Users' Temporary identifiers (TempIDs), Who encountered infected users*, Phone number**, Personal data (e.g., name or age)**	TempIDs of the encountered users, Phone number, Personal data (e.g., name or age)	Australia, France, Singapore, UK (1st version), Czech Republic, Fiji, Gibraltar, Hungary, Malaysia, Mexico, North Macedonia, Philippines, Tunisia, UAE, Vietnam
Rakning C-19 [50], SafePaths(MIT) [51] Tawakkalna [52], ViruSafe [53], Shlonik [54]	Location***	Server via App/SMS, or the Health Authority via telephone	Location of all users, Who encountered whom, Phone number, Personal data (e.g., name, age)	Location, Phone number, Personal data (e.g., name, age)	Bulgaria, Cyprus, Kuwait, Saudi Arabia; North Dakota, South Dakota, Wyoming (US)
Aarogya Setu [55], BeAware [56], Ehteraz [57], MorChana [58], PeduliLindungi [59]	Bluetooth and location	Server via App/SMS, or the Health Authority via telephone	Location of all users, Who encountered whom, Phone number, Personal data (e.g., name, age, gender, occupation)	Location, Phone number, Personal data (e.g., name, age, gender, occupation	Bahrain, Bangladesh, India, Indonesia, Qatar, Thailand, Turkey; Rhode Island (US)
Chinese health code system e.g., Hangzhou [60]	Combination (Location, credit card transactions)	Server via App/SMS, or the Health Authority via telephone	Location of all users, Who encountered whom	Location, Phone number	China
Janus [16]	Bluetooth and UWB	NA	NA	NA	NA
Hoepman et at. [61]	Bluetooth	Server via App	TempIDs	Encrypted TempIDs	NA
Buccafurri et at. [62]	Location and Bluetooth	NA	NA	NA	NA

TABLE VI: List of decentralized contact tracing approaches and adoption.

Approach	Tech	Notification	Data Collected by Server	Data Collected by Client	Adoption
GAEN [3], DP3T-1 [19]	Bluetooth	The App notifies the User	Temporary exposure keys (TEKs) of infected users	TEKs of infected users, TempIDs of the encounter users	Belgium, Canada, Denmark, Estonia, Finland, Germany, Ireland, Italy, Japan, New Zealand, Northern Ireland, Norway, Poland, Saudi Arabia, South Africa, Switzerland, UK, and 23 states in the USA
DP3T-2 [19], MIT-PACT [24], UW-PACT [63]	Bluetooth	The App notifies the User	Temporary identifiers (TempIDs) of infected users	TempIDs of infected users, TempIDs of the encountered users	NA
Pronto-B2 [32]	Bluetooth	the App notifies the User	Hashed pairs of TempIDs in encounters with infected users	TempIDs of infected users, TempIDs of the encountered users	NA
Co100 [64], HaMagen [65]	Location	The App notifies the User	Location of infected users	Location	South Korea, Israel
TraceCORONA [66], Pronto-C2 [25], CleverParrot [26]	Bluetooth	The App notifies the User	Encrypted encounter tokens (ETs) of infected users	Encrypted ETs of infected users, ETs of the encountered users	NA
Epione [27], Dittmer et al. [67]	Bluetooth	The App notifies the User	TempIDs of infected users	TempIDs of the encountered users	NA

Private Set Intersection (PSI) and binary filters. The core function of a DCT app is to find a match between encounter information of infected users and other users. However, this process raises privacy concerns. In centralized approaches, *SP* has access to encounter information of all infected users, whereas in decentralized approaches, any user can receive information about the encounters of all infected users, providing the potential for possibly inferring information about the user's location and social graph. To solve this problem, Trieu et al. [27] and Dittmer et al. [67] leverage a private set intersection (PSI) approach utilizing Function Secret Sharing (FSS) techniques [41]. The idea is that *SP* (which has a set of *TempID*s of all infected users) and a potentially affected user U_j (who has a set of *TempID*s of the users whom U_j has encountered) collaborate to perform encounter matching in a way that only the matching results (e.g., the number of matching *TempID*s) become known to U_j. This means that *SP* does not get to know encounter information of U_j (e.g., *TempID*s) and vice versa. However, a PSI-based approach often incurs high computation and communication overhead which makes it less practical. A more simple approach is to use a cuckoo filter [19], where the *SP* builds a filter from the *TempID*s of the infected users and sends it to user Apps. Apps can check for matches by inputting each of their observed *TempID*s into the filter and receive an output whether the *TempID* is in the filter or not. Thus, Apps can do matching without having access to the complete list of *TempID*s of infected users.

Secret Sharing. In order to prevent *TempID*s to be captured or relayed easily, Troncoso et al. [19] propose to use secret sharing by dividing a *TempID* into n parts, so-called secret shares that are broadcast into the user's vicinity over time. This means that any other user needs to remain in vicinity of the user to capture at least k out of n shares to be able to reconstruct the *TempID*, making it more difficult for an adversary to stage successful attacks by just relaying *TempID*s of people passing by them. However, this approach has several limitations and drawbacks. Firstly, this approach is ineffective in main attack scenarios like shopping, public events or restaurants where adversary has enough time to capture at least k shares. Secondly, this approach would introduce a significant overhead, latency, and error rate because it requires from k to n times more communication overhead (compared to BlueTrace or GAEN) and significant computation to reconstruct the *TempID* from the shares. Further, the authors argue that the communication overhead is insignificant if the number of shares is equal to the number of broadcasts that the App makes within an epoch (the lifetime of a *TempID*). However, this is a flaw as many encounters would not be recorded if the encounters do start at the beginning of the epoch.

Use of Anonymization Networks. A malicious *SP* may track IP addresses of users uploading and downloading encounter information, which potentially could (1) leak personal information of the user, e.g., home addresses and (2) be used to link users with encounter information they upload. Therefore, a number of existing approaches (e.g., [23]) propose to use an anonymization network like Tor or a Mixnet to prevent such potential leakage.

TABLE VII: Advanced privacy techniques for DCT systems

Techniques	References
Blind signature	[28], [25]
Blockchain	[25], [68]
Private set intersection/ a filter	[27], [67], [69], [70], [19]
Secret sharing	[19]
Tor/Mixnet	[23]
Others	[71]

C. Other Application Scenarios of TRACECORONA

Smartphone and Wearable Devices. Smartphones are prohibited or inconvenient to use in many scenarios like in schools, hospitals, corporate offices, sports and other events, beaches, waterparks, funfairs, etc. Therefore, we propose using wearable devices as a complementary approach for contact tracing. Figure 12 shows our smartband-based TRACECORONA. The Bluetooth tracing function is integrated in the smartbands to emit and record the Ephemeral Public Keys Qs and ephemeral IDs which are generated by the TC-Apps on the smartphones. The smartphones are also responsible for calculating ETs ks from private key ds and public key Qs collected by the smartbands. The protocol is summarized as follows:

1) The smartphones generate ECDH private key ds and public Qs, and ephemeral IDs EIs for the next day or the next few days.
2) The smartphones send Qs and EIs to the smartbands every day or every few days or when users interact with the apps.
3) The smartbands exchangse Qs and EIs with other devices (smartbands or smartphone tracing apps) in vicinity.
4) The smartbands send Q's and EI's that they have received from other devices along with associated metadata (timestamp and duration) to the smartphones.
5) The smartphones calculate the ETs from their own private keys ds and other public keys Q's.
6) The smartphones perform all other phases: infection verification, token information upload and token information download.

In order to avoid double encounter recording as both the phone and its pair smartband exchange public keys to other devices in vicinity, the system can detect the co-presence of two devices using a separate secure Bluetooth channel (since both the phone and the smartband are paired) and let only one of the devices perform tracing. In case the phone and its pair smartband are our of range, the smartband will run the tracing functionality but the phone can pop-up a message to ask user whether they prefer do tracing on the phone or not, e.g., if the smartband is running out of battery.

The wearable device and the smartphone frequently synchronize their data, so that the smartphone sends the Qs that it generates to the wearable device and the wearable device sends the Q's it has received from other devices in proximity along with metadata associated with them to the smartphone. This synchronization function can be executed periodically, e.g., daily or when the phone is being charged or when the user interacts with the TRACECORONA app, i.e., when the app is running in the foreground. For one day,

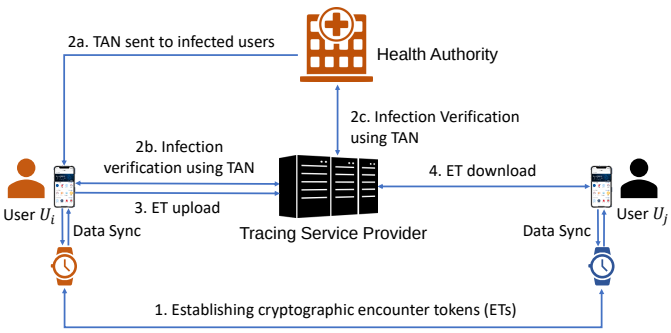


Fig. 12: TraceCORONA system using wristband.

the smartphone sends 96 EIs and 96 EPKs (24 hours x 4 EPKs/hour if T = 15 minutes). In total, the phone sends 96x(65+520) = 56,064 (bits) or 58 kbits. The amount of data that the wearable device sends to the phone depends on how much encounters (contacts) a user has during the day. Recent research shows that on average, a user has about 20 contacts (with a duration over 15 minutes) with other users excluding household members. Therefore, we estimate that the wearable device sends ca. 31 kbits to the phone daily. In theory, Bluetooth LE can transfer from 125 kbit/s to 2 Mbit/s meaning that the daily synchronization process can be done in seconds.

Proximity Communication. In this paper, we focus on Bluetooth Low Energy (Bluetooth LE) as a prominent example. However, TraceCORONA can work with any type of short-range wireless communication protocols like ZigBee or Z-Wave. Further, other communication protocols or data transfer channels like NFC (Near Field Communication), ultrasound or QR code can also be used.

Public and Private Contact Tracing Systems. Many organizations like schools or corporations have needs for performing contact tracing that do not necessarily align well with the needs and requirements of a national or country-level tracing solution [72], [73]. Many businesses and organizations would like to implement their own (private) tracing solutions to have the flexibility of better managing quarantining of employees. To tackle this problem, some manual private contact tracing solutions, e.g., [72] have been introduced. TraceCORONA is flexible in the sense that it can be applied to private contact tracing directly. For example, a TraceCORONA service provider can provide infrastructure (e.g., the server, the apps and dashboard control) to organizations or corporations for a private deployment. Since administrators like personnel departments know who in their organizations or companies are tested positive for COVID-19, they can issue TANs and verify infection state as explained in Sect. VI-C3. Further, our smartband tracing solution can also help to cover many use cases in which using smartphones is forbidden e.g., in many schools or companies as mentioned above.

D. Parameter Settings and Estimations

We now analyze the data use of the TraceCORONA protocols and define following parameters for our analysis.

- EI - Ephemeral Identification (128 bits). EI is used to temporarily identify devices in proximity. item Q - Ephemeral ECDH Public Key (384 bits). Q is used together with associated secret keys to establish encounter tokens.
- $UUID$ - Universally Unique Identifier (128 bits). $UUID$ is used to identify the TraceCORONA app.
- ET - Encounter Token (256 bits). ET is used to uniquely identify the encounter between two users.
- AM - Advertising Message (256 bits). AM is used to advertise (broadcast) $UUID$s and EIs of devices. It changes every T minutes.
- $RSSI$ - Received Signal Strength Indicator. $RSSI$ indicates the strength of a received Bluetooth signal that can be roughly used to estimate the distances between the sender and the receiver.

In the following, we present estimation of data exchange (network bandwidth) and key parameters.

Proximity Detection. Each TraceCORONA App constantly advertises and scans for AM messages. We aim to make sure that a device records other devices in proximity every 2 minutes. The app periodically advertises AMs every minute, during which devices run Bluetooth LE advertising for 40 seconds and are in idle mode for 20 seconds. The app periodically scans AMs every 50 seconds, during which devices run Bluetooth LE scanning for 30 seconds and are in idle mode for 20 seconds. These advertising and scanning patterns are empirically selected to ensure that TraceCORONA keeps track of other devices every 2 minutes.

Public Key Exchange. When a device finds a new device in proximity, the two devices start exchanging public key Q. First, one device starts the Bluetooth client and another device starts the Bluetooth server and both devices exchange their public keys Qs (384-bit length). Both devices store the Qs which later on are used to calculate ETs. Q is changed every T minutes along with EI. During a T-minute lifetime, to save energy, Q is only sent when the device finds a new device.

Encounter distance - 2 meters. In theory, the Bluetooth Low Energy (BLE) signal range is up to 50 meters. In the context of contact tracing, we consider the distance of two meters or smaller.

Encounter time – 15 minutes. The encounter time that is equal to or more than 15 minutes should be considered as a high exposure risk.

Expected number of Encounter Tokens per day - 20. In average, a user has approximately 16 encounters (15 minutes or longer) with other users excluding known contacts e.g., household members or colleagues in the same office [79].

Amount of data uploaded by the app to the server – 4.3 kB. The app of the infected user uploads the hashes (128 bits) of ETs for 14 days (20 ETs per day as mentioned above). Hence, the total amount of data should be, 14*20*128 = 35,840 (bits) or 4.3 (kB).

Amount of data downloaded by the app from the server – 8.6 MB per day. If we assume there are 10,000 new COVID-19 cases per day, the app will download 10,000* 4.3 = 43,000 (kB) or 43 MB per day.

TABLE VIII: Attacks and analyses on existing contact tracing approaches. IDU: Identifying Users, PIU: Profiling Infected Users, ISC: Inferring Social Graph, FEC: Fake exposure Claim, RA: Relay Attack, DCT: Digital Contact Tracing.

	IDU	PIU	ISC	FEC	RA	Effectiveness	Other	Target
Avitabile et al., [25]	x	x	x		X			GAEN DP3T-1 PEPP-PT ROBERT
Avitabile et al., [7]					X			GAEN
Baumgaertne et al., [5]		x	x		x			GAEN
Boutet et al., [74]	x	x	x	x	x			GAEN
Crocker et al., [75]	x	x	x		x	x		GAEN
Danz et al., [76]	x	x	x		x			GAEN DP3T-1
Dehaye et al., [39]					x			GAEN
Gennaro et at., [38]					x			GAEN
Gvili et al., [4]	x	x	x	x	x		DoS*	GAEN
Iovino et al., [6]					x			GAEN
Kojaku et al., [77]						x		GAEN
Lanzing et al., [9]		x	x				Monopolist Coercion	GAEN
Leith et al., [78]						x		Bluetooth
Leith et al., [11]	x	x	x				Data collection	GAEN
Vaudenay et al., [21]		x			x			GAEN DP3T
Vaudenay et al., [29]	x	x	x	x	x			GAEN DP3T PEPP-PT
Wen et al., [43]						x	Data collection	41 apps
White et al., [8]	x	x	x			x		DCT
White et al., [33]	x	x	x			x	Ethics	DCT

Amount of data received by the server (daily) – 43 MB per day. The same to the data that an app downloaded.

Amount of data sent by the server (daily) – 2150 TB per day. If we assume that there are 50 million app users, the server will send 50,000,000* 43 = 2,150,000,000 (MB) or 2150 TB per day.

Maximum numbers of devices in proximity that the App can establish the Encounter Tokens per second - 100. In theory, we can estimate the amount of encounter established based on the bandwidth of BLE communication. To establish an ET, the device need to send and receive an AM and Q that are 192 + 520 = 712 (bits) or 712 *2 = 1,424 (bits) for both sending and receiving. In the worst case, the BLE bandwidth is 125 (kbit/s) that means the app can establish 125,000/ 1,424 = 175 ETs per second. In the ideal case, the BLE bandwidth is 2 Mbit/s that means the app can establish 2,000,000/1,424 = 1,404 ETs per second. However, since the latency to establish a BLE connection is ca. 6 ms, the app can only establish 1000/6 = 160 ETs per second if we consider the data transmission time is much smaller than the connection time. Thus, it is fair to estimate that the app can establish 100 encounters per second.

E. List of Existing Works on Digital Contact Tracing

We summarize attacks on DCT in Tab. VIII.