Component-Based Comparison of Privacy-First Exposure Notification Protocols

Preprint, compiled May 22, 2020

Ellie Daw

TCN Coalition, Crimson Vista
ellie.daw@crimsonvista.com

Abstract

Various privacy-preserving protocols for exposure notification have been developed across the globe in order to aid in scaling contact tracing efforts during the COVID-19 crisis, a strategy proven to be critical in effectively slowing the spread of infectious disease[1]. Although having a multitude of people working toward a similar goal creates momentum and aids in quality refinement, it also causes confusion for entities hoping to adopt one protocol for application development. This paper compares the protocols component-by-component, accumulating in a comprehensive comparison table, so that entities are able to take action based on their priorities.

1 Introduction

It is well known that contact tracing and exposure notification is an important part of the response to infectious diseases. Historically, a manual approach to contact tracing has been taken, where a member of hospital staff or a volunteer will work with the infected patient in order to trace where he or she had been and who he or she had been in contact with. The staff member would then notify each of these contacts about potential exposure. The ability to quarantine the people who have come in to contact with an infected individual means that the spread of the disease can be slowed.

As COVID-19 rapidly spreads, it is clear that manual contact tracing and exposure notification can be difficult to scale up with the same speed. A digital solution could aid the process by automating some of the steps. In general, smartphones will be able to make note of digital identifiers observed while the user is going about daily life, then later check if any of those identifiers came from someone who has been diagnosed with COVID-19.

Importantly, a simulation from researchers at Oxford demonstrates that a carefully planned application could stop the epidemic "if approximately 60% of the population use the app, and even with lower numbers of app users, we still estimate a reduction in the number of coronavirus cases and deaths."[2] To achieve this level of adoption, there are two main pillars of work: technical interoperability of various applications, and consumer willingness to participate. Much effort is being put into technical interoperability by teams around the world, but specifics of those efforts are out of scope for this survey. In order to support consumer willingness, experts realized the need to address privacy concerns that, if left unanswered, may stop people from wanting to participate. In light of this need, digital exposure notification protocols that preserve the privacy of the participants began to be developed.

1.1 *Protocols Surveyed*

This paper is meant to be a survey of those privacy-preserving, digital solutions to exposure notification, comparing the protocols by component. The protocols which fit this description include:

- Temporary Contact Number (TCN) Protocol from the TCN Coalition, all information in this paper is from the TCN GitHub[3].
- Decentralized Privacy-Preserving Proximity Tracing (DP-3T) from many international contributors in both industry and academia; the first author is from École polytechnique fédérale de Lausanne. All information in this paper is from the DP-3T Whitepaper, design 2 with tighter privacy guarantees[4].
- MIT Private Automated Contact Tracing (PACT) from Massachusetts Institute of Technology, all information in this paper is from The PACT protocol specification[5].
- UW Privacy-Sensitive Protocols And Mechanisms for Mobile Contact Tracing (PACT) from the University of Washington and Microsoft, all information in this paper is from the PACT eprint article[6].
- Google/Apple Exposure Notification (GAEN) from Apple and Google, all information in this paper is from the Cryptography Specification[7].

The following components are compared:

- Anonymous ID Generation (Section 4)
- Reporting Upon Positive Infection (Section 5)
- Anonymous ID Observation and Exposure Matching (Section 6)
- Responsibility of the Backend Server (Section 7)

Further, if a protocol's documentation included specific additional information on security and privacy concerns, those are included in sections 8 and 9. Additional project differentiators are also discussed in section 10. Finally, a summary of the component comparisons is included in the form of a table in section 11.

1.2 *Limitations of Digital Exposure Notification*

A digital contact-tracing app is not the only piece to the puzzle as we look toward lessening the negative effects of COVID-19 in the near future. There are limitations to digital exposure notification technologies including false positives, false negatives, adoption rate and user concerns[8], defining a contact event, and more. These concerns are critical to an all-encompassing solution, but are out of the scope of this survey.

2 Definitions

Contact Tracing is the process of public health staff working with a patient "to help them recall everyone with whom they have had close contact during the timeframe while they may have been infectious."[1] **Exposure Notification** refers to the next step taken by the staff member: notifying the identified contacts of their potential exposure to the infection. These processes have historically been manual; hence the new term *digital exposure notification*, which refers to technology which can perform the contact tracing and exposure notification functions of human tracers without the same scaling challenges.

Deterministic algorithms always produce the same output for a given input. This contrasts probabilistic algorithms which introduce an element of randomness.

Decentralized refers to computations being performed at the endpoints, or smartphones in the case of exposure notification, as opposed to being performed at a centralized server.

Cuckoo filters are probabilistic data structures which allow for set membership checking with an acceptable false positive rate (which can be chosen by the implementer). They are an improvement on the widely used *bloom filter* because they allow for deletion and have better performance[9].

3 Protocol from 10,000 Feet

The protocols described in this paper are privacy-preserving and decentralized. They all follow the same general steps:

1. User1 generates local key material, creates anonymous identifiers, and broadcasts these anonymous identifiers to nearby users via Bluetooth.
2. Users store anonymous identifiers that they observe from others.
3. When User1 tests positive for COVID-19, she uploads relevant information to a backend server.
4. Users periodically download data from the backend server, using the relevant information to re-generate anonymous identifiers, subsequently checking if those re-generated values from an infected user match any of the stored values that he or she observed. If yes, the app alerts the user that he or she could have been exposed.

4 Anonymous ID Generation

This section describes the way each protocol generates the anonymous identifiers it broadcasts. Table 1 summarizes these processes.

4.1 *TCN*

The user's device will generate an Ed25519 keypair, calling the private key the *report authorization key* `rak` and the public key the *report verification key* `rvk`. The subsequently generated values are called *temporary contact key (TCK)*s. The device generates initial *TCK*s by hashing `rak` using a domain-separated hash function with 256-bits of output (`H_tck`). The reference implementation uses SHA256 with `b‘‘H_TCK’’` as the domain separator. The hash digest and the `rvk` are concatenated and hashed, outputting the first usable *TCK*, called `tck_1`.

$$\texttt{tck\_0} \leftarrow \texttt{H\_tck(rak)}$$
$$\texttt{tck\_1} \leftarrow \texttt{H\_tck(rvk} \,\|\, \texttt{tck\_0)}$$

TCK's are ratcheted values; the operation is as follows:

$$\texttt{tck\_i} \leftarrow \texttt{H\_tck(rvk || tck\_\{i-1\})}$$

TCN's are derived from a `tck_i` by:

$$\texttt{tcn\_i} \leftarrow \texttt{H\_tcn(le\_u16(i)} \,\|\, \texttt{tck\_i)}$$

where `H_tcn` is a domain separated hash function with 128 bits of output. The reference implementation uses SHA256 with `bḦ_TCN¨` as the domain separator. To prevent linkability attacks, the TCK ratchet is synchronized with the MAC rotation at the Bluetooth layer.

4.2 *DP-3T*

In DP-3T, the ephemeral ID values are called Ephemeral IDs `EphID`, and are valid for an *epoch* `i` which is encoded relative to a fixed value for all entities in the system.

The device generates a random 32 byte seed per epoch and sets

$$\texttt{EphID} = \mathit{TRUNCATE128}(H(\texttt{seed}_\texttt{i}))$$

where `H` is a hash function and `TRUNCATE128` truncates the output to 128 bits. Seeds for all epochs in the last 14 (a variable that should be determined by health authorities) days are stored, which allows regeneration of `EphIDs` later by a device checking for exposure.

4.3 *MIT PACT*

Every hour, a device generates a random 256 bit seed `s` which is used to generate a Bluetooth *chirp* `r_t`, calculated

$$\texttt{r\_t} = \mathit{PRF}(\texttt{s, t})$$

where `t` is the current time and `PRF` is a pseudo-random function. The seeds and the times they were generated are stored on the user's device for a medically relevant period of time.

4.4 *UW PACT*

The UW PACT protocol defines the following parameters:

- time unit $\mathtt{dt}$ where $\Delta\ x\ \mathtt{dt}$ = infection window, i.e. two weeks
- bit length n of identifiers = 128
- a function $G : \{0, 1\}^n \rightarrow \{0, 1\}^{2n}$, a secure pseudorandom number generator

With $n = 128$, $G(x)$ can be $SHA256(x)$. To calculate an ephemeral ID $\mathtt{id}_i$, a device begins by generating a random n-bit seed $\mathtt{S}_0$, then calculates:

$$(\mathtt{S_i},\ \mathtt{id_i}) \leftarrow G(\mathtt{S_{i-1}})$$

The user only stores the set of seeds S^* for a relevant time period, for example 14 days.

4.5 GAEN

This protocol leverages a daily *Temporary Exposure Key* which is used to generate a *Rolling Proximity Identifier Key*, which is then used to generate the ephemeral IDs called *Rolling Proximity Identifiers*. To set up for participation in exposure notification, a device will need a key associated with an interval number. *ENIntervalNumber* provides a number for each 10 minute time window, calculated based on Unix epoch time:

$$ENIntervalNumber(\mathtt{Timestamp}) \leftarrow \frac{\mathtt{Timestamp}}{60x10}$$

ENIntervalNumber is a 32 bit unsigned little endian value (`uint32_t`). The *TEKRollingPeriod* is the number of intervals for which a *Temporary Exposure Key* is valid, defined here as 144 (the number of 10 minute intervals in one 24-hour day). A device must first calculate which interval to begin at, calculated as:

$$\mathtt{i} \leftarrow \left\lfloor \frac{ENIntervalNumber(\mathtt{t_g})}{\mathtt{TEKRollingPeriod}} \right\rfloor \times \mathtt{TEKRollingPeriod}$$

where $\mathtt{t_g}$ is the time at key generation. With this context, the device now generates a *Temporary Exposure Key* for interval $\mathtt{i}$ using:

$$tek_i \leftarrow CRNG(16)$$

where *CRNG* is a cryptographic random number generator. Each key is stored along with its interval number, and the key is regenerated at the end of each *TEKRollingPeriod*. *Rolling Proximity Keys* are generated using the *Temporary Exposure Key* by:

$$\mathtt{RPIK_i} \leftarrow HKDF(tek_i,\ NULL,\ \mathtt{UTF8(``EN\text{-}RPIK'')},\ 16)$$

which is used to derive *Rolling Proximity Identifiers*:

$$\mathtt{RPI}_{i,j} \leftarrow AES\_128(\mathtt{RPIK}_i,\ \mathtt{PaddedData}_j)$$

where

- $\mathtt{j}$ is the Unix epoch time at the time of RPI generation
- $\mathtt{ENIN}_j \leftarrow ENIntervalNumber(j)$

and `PaddedData` is the following 16 bytes:

- $\mathtt{PaddedData}_j[0...5]$ = UTF8(‘‘EN-RPI’’)
- $\mathtt{PaddedData}_j[6...11] = 0x000000000000$
- $\mathtt{PaddedData}_j[12...15] = \mathtt{ENIN}_j$

Protocol	Anonymous ID Generation	Anonymous ID Beacon Length (bytes)
TCN	Asymmetric keypair, deterministic ratchet	16
DP-3T	Random seed, deterministic hash + truncate	16
MIT PACT	Random seed, deterministic PRF	28
UW PACT	Random seed, deterministic PRF	32
GAEN	Random temporary key, deterministic HKDF	32

Table 1: Anonymous ID Generation and Anonymous ID Beacon Length in bytes (note that these values are typically set with a recommended value and could theoretically be different by, for example, choosing a different hash function)

The `RPI` is replaced each time the Bluetooth randomized MAC address changes.

Associated Encrypted Metadata is concatenated to the RPI and the two are broadcast together. The Associated Encrypted Metadata can only be decrypted upon a positive test report by the user who broadcast the data.

$$\mathtt{AssociatedEncryptedMetadata}_{i,j} \leftarrow AES_{128}CTR(AEMK_i,\ RPI_{i,j},\ Metadata)$$

The Associated Encrypted Metadata Key is generated using the Temporary Exposure Key as follows:

$$AEMK_i \leftarrow HKDF(tek_i,\ NULL,\ UTF8(\text{"}\mathtt{EN\text{-}AEMK}\text{"}),\ 16)$$

5 Reporting upon Positive Infection

Each protocol varies slightly in how it reports a positive infection result. Table 2 summarizes this section.

5.1 TCN

A TCN report notifying contacts encountered between $0 < j1 < j2$ is structured as:

$$\mathtt{report} \leftarrow \mathtt{rvk} \,||\, \mathtt{tck\_\{}j1{-}1\mathtt{\}} \,||\, \mathtt{le\_u16}(j1) \,||\, \mathtt{le\_u16}(j2) \,||\, \mathtt{memo}$$

where `memo` is a variable-length byte string between 2 and 257 bytes long. It provides a compact space for free-form messages, ensuring extensibility. It has the structure:

$$\mathtt{type:}\ \ \mathtt{u8} \,||\, \mathtt{len:}\ \ \mathtt{u8} \,||\, \mathtt{data:}\ [u8;\ len]$$

where data is 0-255 bytes long encoded with type `type`. Types defined so far include:

- `0x0` CoEpi symptom report v1
- `0x1` CovidWatch test result v1
- `0x2` ito report v1
- `0xff` reserved (can be used to add more than 256 types later)

The rest values between `0x2` and `0xff` are reserved for allocation to applications upon request.

5.2 DP-3T

Once a user has tested positive and wishes to report their positive result, they select the set of epochs I for which to upload seeds and sends a set of $(i, \mathtt{seed}_i)$ for all epochs i in set I. The purpose of explicit epoch identification is to allow users to exclude certain epochs, such as all epochs for a certain evening.

5.3 MIT PACT

Permission numbers are generated by a testing authority and dis-jointly distributed to healthcare providers. Each permission number is single use and will be given to an infected individual by a health authority, allowing that individual to upload *chirp logs* to the server. Once the user tests positive and the individual has obtained a valid *permission number*, the user's device will upload the tuple $(\mathtt{s}, \mathtt{t}_1, \mathtt{t}_2)$ where s is the seed used to generate a *chirp*, and $\mathtt{t}_1$ and $\mathtt{t}_2$ are the start and end times of the seed validity window.

5.4 UW PACT

To report a positive test, a device will upload the set of stored seeds, the start time, and the end time (S^*, t^*, t_i) to the server, then deletes all seeds stored locally. The server performs sanity checks on the time values and adds the seeds to the public list L.

5.5 GAEN

When a person tests positive, their device uploads the *Temporary Exposure Keys* and the *ENIntervalNumber* i for the time period which they may have been exposing other users. This set of keys is referred to as *Diagnosis Keys*.

Protocol	Reporting	Report Length (bytes)
TCN	Uploads report with public key, the start time, the end time, the tck which can be used to regenerate tcn's for the time block, and an optional memo field	134-389 signed, 70-325 unsigned
DP-3T	Uploads a set of epoch number and seed $(i, \mathtt{seed}_i)$ for all relevant epochs	32+epoch number length
MIT PACT	Obtains *permission number* and uploads *chirp logs* consisting of seed, start time, and end time tuples $(\mathtt{s}, \mathtt{t}_1, \mathtt{t}_2$	32+time lengths
UW PACT	Uploads a set of stored seeds, start time, and end time (S^*, t^*, t_i)	32+time lengths
GAEN	Uploads *Diagnosis Keys* consisting of *Temporary Exposure Keys* and the *ENIntervalNumber* i	32+interval number length

Table 2: Reporting

6 Anonymous ID Observation and Exposure Matching

Each device must be listening for the appropriate Bluetooth broadcasts. Table 3 summarizes the information in this section.

The processes used by the different protocols for exposure matching is largely similar for most protocols. Table 4 summarizes this section.

6.1 TCN

When a device observes a TCN, it stores the anonymous value locally in order to check against re-generated values from reports fetched later.

A user wishing to check for exposure notifications should obtain positive reports from the server, recompute the TCNs, and compare the recomputed TCNs with the TCNs they observed. The user can also verify the integrity of the report by verifying the signature using the included rvk. TCNs can be regenerated by:

$$\begin{aligned}
\mathtt{tck\_j1} &\leftarrow H\_tck(\mathtt{rvk} \,\|\, \mathtt{tck\_\{j1-1\}})\\
\mathtt{tcn\_j1} &\leftarrow H\_tcn(\mathtt{le\_u16}(j1) \,\|\, \mathtt{tck\_j1})\\
\mathtt{tck\_\{j1+1\}} &\leftarrow H\_tck(\mathtt{rvk} \,\|\, \mathtt{tck\_j1})\\
\mathtt{tcn\_\{j1+1\}} &\leftarrow H\_tcn(\mathtt{le\_u16}(j1+1) \,\|\, \mathtt{tck\_\{j1+1\}})\\
&\ldots
\end{aligned}$$

The steps above are *Ratchet*, then *Generate*; *Ratchet*, then *Generate*; etc.

6.2 DP-3T

When a device observes EphIDs, it stores $H(\mathtt{EphID} \,\|\, \mathtt{i})$, the proximity, the duration, and a coarse time indicator (for example, the date).

The backend server sends the Cuckoo filter F after periodic updates to all user devices. The devices then check to see if any of the hash values saved after EphID observation are present in F. If any *are* present, the user may be at risk and the risk score can be calculated by the application.

6.3 MIT PACT

When a device observes a PACT *chirp*, it stores the *chirp*, the time the *chirp* was observed (if the same value was observed multiple times in one minute, it is consolidated into one), the Bluetooth signal strength, and optionally the location where the *chirp* was observed. This set of data is stored locally for 3 months.

A device will download relevant portions of the exposure database, re-generate *chirp* values from the seeds, and check for matching values in the set of seeds it observed. The values downloaded will be of the form (s, t_1, t_2), and the device will calculate

$$\mathtt{r} = PRF(\mathtt{s}, \mathtt{t})$$

for each time value between t_1 and t_2, checking if those values matched any of the observed values.

Protocol	Anonymous ID Observation and Local Storage
TCN	Stores TCN value
DP-3T	Stores hash of `EphID` and epoch number `i`, proximity, duration, and coarse time indicator
MIT PACT	Stores *chirp*, time, Bluetooth signal strength, and optionally location
UW PACT	Stores `id` and time `t`
GAEN	Stores RPI value

Table 3: Anonymous ID Observation and Local Storage

Protocol	Exposure Matching
TCN	Downloads reports from server, verify report signatures, re-generate TCNs, check against list of observed TCNs
DP-3T	Server sends Cuckoo filter `F` to devices, devices check whether saved values (hash of `EphID` and epoch number `i`) are present in the filter, if yes then alert a potential exposure
MIT PACT	Downloads relevant portions of exposure database, re-generate *chirp* values, check against observed *chirp*s.
UW PACT	Download relevant portions of public list, re-generate `id`s from seeds and approximate time of each `id` broadcast, check (`id`, `t`) pair against observed pairs
GAEN	Downlaods *Diagnosis Keys* from server, re-generate RPIs, check against list of observed RPIs

Table 4: Exposure Matching

6.4 UW PACT

When a device observes an anonymous `id`, it stores the `id` value along with the time of observation `t` (`id`, `t`).

A device will download relevant portions of the public list L, and re-generates anonymous IDs starting from $\mathtt{S}^*$ along with the approximate time each `id` would have been broadcast based off of the start and end times t^* and t_i and the generation process. If any of the re-generated `id`s match what they have observed and the approximate time is sufficiently close to the time they recorded the observation, their device will notify them of a potential exposure.

6.5 GAEN

When a device observes an `RPI`, it stores the value in order to check against re-generated values from diagnosis keys fetched later.

Each device periodically downloads *Diagnosis Keys* from the *Diagnosis Server*, using the keys and corresponding interval numbers to re-generate the `RPI`s for that interval. It can then check to see if it observed any of the `RPI`s it just generated, provided the interval numbers correspond as well (a +/- two hour tolerance is allowed).

Protocol	Backend Responsibilities
TCN	Report storage
DP-3T	Cuckoo filter `F` containing uploaded seed pairs, pushing `F` to client devices
MIT PACT	Chirp log storage, permission number validation
UW PACT	Public list L storage
GAEN	Diagnosis key storage

Table 5: Backend Responsibilities

7 Responsibility of the Backend Server

An important consideration for organizations wishing to implement one of these protocols is the responsibility of the backend server. Often, the protocols will not define many of the components that go in to backend development. However, sometimes the protocol will require certain functionality out of the server. Table 5 summarizes this section.

7.1 TCN

The server simply stores the reports, serving them to clients who request them for their own local checking. Servers may validate the report signatures, but should leave the signatures intact for the client to verify.

7.2 DP-3T

The backend server for DP-3T implements a Cuckoo filter, periodically generating a new filter `F` and inserting uploaded seed pairs $(\mathtt{i}, \mathtt{seed}_i)$ by inserting

$$H(\mathit{TRUNCATE128}(H(\mathtt{seed}_i)) \,||\, \mathtt{i})$$

which is $H(\mathtt{EphID}||\mathtt{i})$. The server also sends the Cuckoo filter `F` to client devices rather than the clients fetching `F` from the server.

7.3 MIT PACT

The backend server for PACT implementations is expected to first verify the validity of a permission number, then to store and make available the *chirp logs* which are uploaded to it.

7.4 UW PACT

The server simply stores the public list L, serving relevant portions to clients who request them for their own local checking.

7.5 GAEN (Apple/Google)

The backend is referred to as the *Diagnosis Server*, and aggregates *Diagnosis Keys* from all users who have tested positive. These keys are made available to all users for exposure matching.

8 Security Considerations

In this section, various relevant security considerations are described, and Table 6 summarizes which protocols address each consideration.

Linkability Anonymous IDs should not be linkable to one another. This is mitigated by the protocols included in this survey through a combination of cryptographic pseudorandomness and intentional ID rotation schedules. Even for the protocols which have a single seed generate multiple anonymous IDs, linkage is still limited to period for which that seed is valid (chosen to be small).

Replay Attacks Anonymous identifiers are observed and re-broadcast by an adversarial user. The risk of successful replay attacks is the triggering of a false exposure notification. This is mitigated by most protocols in this survey by incorporating a timestamp or a validity period in either the reporting or the observation phases, which can then be checked during exposure matching.

Impersonation Attacks A malicious user tests positive and reports another user's reporting data, or a malicious user attempts to broadcast identifiers belonging to another user. If **strong source integrity** is achieved, these attacks should not be possible.

Address Carryover Attack The TCN GitHub site has a helpful description and illustration of an address carryover attack, which happens when multiple identifiers can be linked to the same source. In exposure notification protocols that broadcast via Bluetooth, this would happen when the Bluetooth MAC address rotation and the anonymous identifier rotation are not aligned, causing some overlap. This can be mitigated by making sure that anonymous identifiers rotation aligns with Bluetooth MAC address rotation. The rotation frequencies do not have to match, but overlap should be avoided. All of the protocols surveyed here explicitly describe this, except for DP-3T; it can still be mitigated in DP-3T if the number of `EphIDs` `n` is chosen intentionally.

	Linkability Prevention	**Replay Attack Prevention**	**Impersonation Attack Prevention**	**Strong Source Integrity**	**Address Carryover Attack Prevention**
TCN	✓	✓	✓	✓	✓
DP-3T	✓	✓	✓	-	-
MIT PACT	✓	✓	✓	-	✓
UW PACT	✓	✓	✓	-	✓
GAEN	✓	✓	✓	-	✓

Table 6: A ✓indicates that the privacy security is addressed and mitigation strategies are suggested in the protocol documentation. A - indicates that the security concern is either not met or not discussed in the documentation.

9 Privacy Considerations

In this section, various relevant privacy considerations are described, and Table 8 summarizes which protocols address each consideration. Due to the protocols in this paper all being privacy-preserving, many of these privacy properties are inherent to the design of the protocol. Therefore, the privacy concern is addressed without having to provide additional mitigation strategies.

Broadcast IDs Anonymity The identifiers which are broadcast should be indistinguishable from random values. This enables privacy in the case of an honest-but-curious server, and enables privacy-preservation between users of the system. All of the protocols included in this survey address this concern inherently through the use of the cryptographic mechanisms discussed above during anonymous ID generation.

Re-Identification The ability to identify which user an anonymous ID belongs to after that user has contributed a positive test result to the server. In the broader context of contact tracing and exposure notification, complete protection against this can be very difficult, even when the process is not digital. For example, if a person has only come in to close proximity with one other person in the previous two weeks and receives an exposure notification, she can infer the identity of the person who tested positive.[6] Re-identification concerns are addressed to the extent possible within these protocols by not collecting any identifying information and through the use of cryptography.

Privacy of Non-Infected Users The privacy of a non-infected user who has not submitted a positive infection report to the server should be protected. The protocols included in this survey address this concern by design; data does not leave the device of a user unless both 1) the user tests positive, and 2) the user consents to reporting his or her positive test result. One exception to this is the additional opt-in feature available in DP-3T for sharing data with epidemiologists. Even then, such data does not include location information.

Data Privacy on the Backend Server Because the backend server will store all reports of a positive test, we want to be sure that this scenario does not cause a privacy leak. The protocols included in this survey all address this by not collecting identifying information in the first place. As long as the backend does not log IP addresses for correlation to reporting data, privacy of the individual is preserved. DP-3T's additional use of a Cuckoo filter hides the value of the anonymous identifiers as well.

Location Privacy The protocol should not reveal or share location data with any party as this could be sensitive and/or identifying. All of the protocols included in this survey share no location data. However, this does not prevent the scenario where a device recording anonymous identifier broadcasts also stores the location where each broadcast was seen (this is pointed out as a feature in MIT PACT), allowing a positive exposure notification to lead to re-identification of the positive user.

	Broadcast ID Anonymity	Re-Identification Prevention	Privacy of Non-Infected Users	Privacy on the Backend Server	Location Privacy
TCN	✓	✓	✓	✓	✓
DP-3T	✓	✓	✓	✓	✓
MIT PACT	✓	✓	✓	✓	✓
UW PACT	✓	✓	✓	✓	✓
GAEN	✓	✓	✓	✓	✓

Table 7: A ✓indicates that the privacy concern is addressed and mitigation strategies are suggested in the protocol documentation. A - indicates that the privacy concern is either not met or not discussed in the documentation.

10 Additional Project Differentiation

10.1 TCN

TCN Coalition TCN was developed by the TCN Coalition, which united a handful of privacy-preserving exposure notification efforts in order to develop one protocol together to support interoperability. The TCN Coalition has expanded charter to support all privacy-preserving exposure notification initiatives (including those that do not use the TCN protocol) in a variety of ways, as well as to advocate for the privacy of consumers.

10.2 DP-3T

The ability to share data with an epidemiologist This feature is completely opt-in, and does not share location data, rather only shares limited anonymous data upon a contact event with a user who tested positive. If a user is enrolled in this feature, it also sends dummy data at appropriate intervals to avoid leakage of contact events. Epidemiologists can use this data for contact graph analysis.

10.3 MIT PACT

Reverse Contact Tracing In the case that a user becomes infected but has not received an exposure notification or does not know where the infection may have come from, the MIT PACT team proposes the inclusion of a flag in the reporting protocol. This would allow the same exposure matching process to occur, but if a match is found the notification would alert that the user may have been the source of infection.

Chirp Repeaters with Delay One of the challenging scenarios to address is the transmission of COVID-19 on surfaces which are used frequently but not frequently sanitized; the example given by the MIT PACT paper is the checkout counter at a store[5]. The team has proposed devices which observe Bluetooth chirps and repeat them for a fixed amount of time for observation by devices who come into proximity soon after.

10.4 UW PACT

Narrowcasting The UW PACT team has been working with health authorities to develop a way to make announcements to groups of people who may have been present at a certain location at a certain time. They propose that devices would query the server for relevant announcements, preserving location privacy of the user.

10.5 GAEN

Having such a big part of the smartphone market share between them, Apple and Google's collaborative announcements have demonstrated support of a privacy-first approach. One of their notable announcements stated that the exposure notification functionality would be dismantled after 18 months; in other words, they wanted to communicate that this initiative does not suddenly open doors for extended surveillance efforts.

11 Conclusion and Summary Comparison Table

Contact tracing and exposure notification is an important piece of the strategy to slow the spread of infectious diseases. Digital exposure notification technologies aim to complement these processes which are typically done manually, and to do so, the applications must be downloaded by an overwhelming majority of the population. Protecting the privacy of all of the participants is a priority for many technologists working on this problem, and various privacy-preserving exposure notification protocols have been developed.

Although there are important potential limitations to the effectiveness of digital exposure notification applications which warrant discussion, those are out of the scope of this paper. Instead, each component of the privacy-preserving protocols are outlined based on their open-source documentation. The following table includes all of the attributes surveyed in this paper, accumulated into one table for easy comparison. The 10,000 foot view is quite similar for this group of protocols, and they all aim to preserve privacy. This side-by-side comparison shows how each component varies.

	TCN	**DP-3T**	**MIT PACT**	**UW PACT**	**GAEN**
Anonymous ID Generation	Asymmetric keypair, deterministic hash ratchet	Random seed, deterministic hash + truncate	Random seed, deterministic PRF	Random seed, deterministic PRF	Random temporary key, deterministic HKDF
Anonymous ID Beacon Length (bytes)	16	16	28	32	32
Reporting	Uploads `report` with public key, the start time, the end time, the `tck` which can be used to regenerate `tcn`'s for the time block, and an optional memo field	Uploads a set of epoch number and seed $(i, \texttt{seed}_i)$ for all relevant epochs	Obtains *permission number* and uploads *chirp logs* consisting of seed, start time, and end time tuples $(\texttt{s}, \texttt{t}_1, \texttt{t}_2$	Uploads a set of stored seeds, start time, and end time (S^*, t^*, t_i)	Uploads *Diagnosis Keys* consisting of *Temporary Exposure Keys* and the *ENIntervalNumber* `i`
Report Length (bytes)	134-389 signed, 70-325 unsigned	32+epoch number length	32+time lengths	32+time lengths	32+interval number length
Anonymous ID Observation and Local Storage	Stores TCN value	Stores hash of `EphID` and epoch number `i`, proximity, duration, and coarse time indicator	Stores *chirp*, time, Bluetooth signal strength, and optionally location	Stores `id` and time `t`	Stores RPI value
Data Reported Beyond Seed and Timing Info	Optional memo field	-	-	-	Associated Encrypted Metadata
Exposure Matching	Downloads reports from server, verify report signatures, re-generate TCNs, check against list of observed TCNs	Server sends Cuckoo filter `F` to devices, devices check whether saved values (hash of `EphID` and epoch number `i`) are present in the filter, if yes then alert a potential exposure	Downloads relevant portions of exposure database, re-generate *chirp* values, check against observed *chirp*s	Download relevant portions of public list, re-generate `id`s from seeds and approximate time of each `id` broadcast, check $(\texttt{id}, \texttt{t})$ pair against observed pairs	Downloads *Diagnosis Keys* from server, re-generate RPIs, check against list of observed RPIs
Backend Responsibilities	Report storage	Cuckoo filter `F` containing uploaded seed pairs, pushing `F` to client devices	Chirp log storage, permission number validation	Public list L storage	Diagnosis key storage
Linkability	✓	✓	✓	✓	✓
Replay Attacks	✓	✓	✓	✓	✓
Impersonation Attacks	✓	✓	✓	✓	✓
Strong Source Integrity	✓	-	-	-	-
Address Carryover Attack	✓	-	✓	✓	✓
Broadcast ID Anonymity	✓	✓	✓	✓	✓
Re-Identification	✓	✓	✓	✓	✓
Privacy of non-infected Users	✓	✓	✓	✓	✓
Privacy on the Backend Server	✓	✓	✓	✓	✓
Location Privacy	✓	✓	✓	✓	✓

Table 8: A ✓indicates that the concern is addressed and mitigation strategies are suggested in the protocol documentation. A - indicates that the concern is either not met or not discussed in documentation.

References

[1] Content source: National Center for Immunization CDC Website and Division of Viral Diseases Respiratory Diseases (NCIRD). Contact tracing : Part of a multipronged approach to fight the covid-19 pandemic.

[2] Digital contact tracing can slow or even stop coronavirus transmission and ease us out of lockdown. https://www.research.ox.ac.uk/Article/2020-04-16-digital-contact-tracing-can-slow-or-even-stop-coronavirus-transmission-and-ease-us-out-of-lockdown, April 2020.

[3] Tcn coalition github. https://github.com/TCNCoalition/TCN, 2020.

[4] Decentralized privacy-preserving proximity tracing. https://github.com/DP-3T/documents/blob/76fdc4e8210c6c9f75ed1e381282d6f8d4655745/DP3T%20White%20Paper.pdf, 2020.

[5] The pact protocol specification. https://pact.mit.edu/wp-content/uploads/2020/04/The-PACT-protocol-specification-ver-0.1.pdf, 2020.

[6] Justin Chan, Dean Foster, Shyam Gollakota, Eric Horvitz, Joseph Jaeger, Sham Kakade, Tadayoshi Kohno, John Langford, Jonathan Larson, Puneet Sharma, Sudheesh Singanamalla, Jacob Sunshine, and Stefano Tessaro. Pact: Privacy sensitive protocols and mechanisms for mobile contact tracing. https://arxiv.org/pdf/2004.03544.pdf, 2020.

[7] Exposure notification, cryptography specification. https://covid19-static.cdn-apple.com/applications/covid19/current/static/contact-tracing/pdf/ExposureNotification-CryptographySpecificationv1.2.pdf, 2020.

[8] Elissa M. Redmiles. User concerns tradeoffs in technology-facilitated contact tracing. https://arxiv.org/abs/2004.13219, 2020.

[9] Bin Fan, David G. Andersen, Michael Kaminsky, and Michael D. Mitzenmacher. Cuckoo filter: Practically better than bloom. *CoNEXT*, pages 75–88, 2014.