# Adding Location Context to Apple/Google Exposure Notification Bluetooth API: MIT SafePaths Encryption Proposals for GPS + Bluetooth

Camera Culture Group, MIT Media Lab
MIT Alliance on Distributed and Private Machine Learning
SafePaths Alliance

Version 0.1, Apr 26th, 2020

## Abstract:

Contact tracing requires a strong understanding of location and context of the infection encounters. Although Bluetooth technology does not provide location or context of the encounters, we present key privacy preserving ideas to capture this context that can be used in or alongside the forthcoming Google/Apple Bluetooth Exposure Notification API (GAEN). There are four different ways of propagating context between the two users. We propose two simple ideas to allow private location logging without revealing the location history in the app. In addition, we propose two encryption based methods. The first encryption method is a variant of Apple FindMy protocol that already allows nearby Apple devices to capture GPS location of the lost Apple device. The second encryption method, which we recommend is the best option, is a minor modification of the existing GAEN protocol so that GPS information is available to a healthy phone only if exposed. We highlight the benefits and potential privacy issues with each proposed method for context propagation. It will still be the role of the Public Health smartphone app to decide how to use the location-time context to build a full fledged contract tracing and public health solution.

## Motivation:

Currently exposure notification in GAEN only reports the day of the exposure but not the location or time. We believe the context of location and time is critical for (i) user to self assess if they were exposed (e.g. if they were wearing a mask or maintaining social distance at that time) and inform others who were with this person in case they were not carrying the smartphone (ii) improve user's trust in the system to reject false positives (e.g. if they picked up BT signal from behind a wall) (iii) help public health officials to perform the contact tracing operations that require such a context (e.g. to request everyone at a wedding to self-isolate at home if there was an infected person for a long time). Lack of context and the lack of agency can lead to irrational behavior and even civil unrest as we have explained in our document on Contact Tracing: Holistic Solution beyond Bluetooth. The user's behaviour and their level of precautionary measure taken varies in different environments. The risk from an exposure to a carrier is different in different situations. This makes the need for spatio-temporal context extremely relevant in contact tracing.

Our goal is to allow location-time context to be delivered to the user during exposure notification. Here are some possibilities and challenges.

- Log location in the same app: The App can get rejected because it stores time or location along with the Bluetooth packets under the current protocol specification and compliance by the Google/Apple Exposure Notification(GAEN) API.
- Use a secondary app: One can store the GPS trails and time stamp using another app which again creates the issue for communication between the two apps and also prevents mass adoption, as only a small percentage would install and run the two apps in parallel.
- Reuse ENIN information: The GAEN approved app can estimate timestamp from ENIN which is windowed at every ten minutes and uses a tolerance window of +/-2 hours during the diagnosis key matching. So, one way of achieving this is to use the existing protocol and use ENIN timestamp from the Bluetooth payload as a primary key between the two databases and look up GPS for intersecting code using another app.

There is another important aspect of contact tracing which has been overlooked in the majority of the existing Bluetooth based approaches - the lack of assistance to the Policy Makers, Epidemiologists, City Officials, Response Teams etc. The existence of location context in a contact tracing app allows the Covid+ citizens, with consent, to provide data to the aforementioned decision makers which can shape the policy significantly to a level where the outcome could change drastically if right interventions are performed.

## SafePaths Protocol to combine GPS + GAEN:

We propose a protocol that stays very close to the GAEN protocol based on the idea that GPS information is available to a healthy phone only if exposed. If Alice is healthy and comes in proximity to Bob who was later diagnosed Covid+, Alice will be able to see the location (and time) of that encounter but the rest of her location history remains invisible or encrypted. This is our suggested modification to GAEN. We also propose other three ideas that do not require changes to GAEN protocol but require Apple/Google to allow such apps to access GPS privately on device and run other servers.

The 4 privacy preserving solutions to provide location-time context can be summarized as follows :
The idea is to use BLE for proximity and GPS for context (but not proximity).

1. GPS logged on device, data does not leaves the phone, no visualization
2. GPS + time blurred and logged on phone, data does not leaves the phone, no visualization
3. FindMy variant:  Encrypt RPI
4. GAEN variant: Encrypt your own GPS with DailyKey and Broadcast over BLE

Please use this document for GAEN terminology which we also use here.
Unless stated otherwise, Healthy person is X (Alice) and a person who gets diagnosed of COVID is Y (Bob).
RPI: Rolling Proximity Identifiers are privacy-preserving identifiers that are broadcast in Bluetooth payloads.
ENIN: Exposure Number Interval Number (Index of a 10 min window, 144 such windows per day)
DailyKey: Diagnosis Keys of which a subset becomes Temporary Exposure Keys (previously known as Daily Tracing Keys)

1. *GPS stays on the app for every device, cannot be visualized and does not leave the phone*

a. Details
    i. Approach 1 based on direct RPI indexing: Every User logs their location indexed with RPI (rotating proximity identifiers). When exposure notification arrives, it matches the RPI with logged location. Here we keep the location only if there is RPI (which means there is another BT user). In this case the database of GPS is indexed with RPI.
    ii. Approach 2 based on calculated ENIN: The 144 RPI generated from the Daily Key provides the match with one of the RPIs (10 minute window) so App has access to the timestamp. From that timestamp, the app can recover the GPS location for that time. In this case the database of GPS is indexed with time.
    iii. No direct visualization of location log is made available to the user to prevent unauthorized persons (nosy employers, abusive spouses or border agents) from seeing it.
b. Benefits
    i. Location data stays local and casual unauthorized reader cannot see (e.g. nosy employer, abusive spouse)
c. Issues
    i. The App could get rejected because it may not follow GAEN guidelines if it stores location log.
    ii. Sophisticated hackers can reverse engineer the exact location from logs.

2. *BT for proximity, GPS for context, GPS is blurred sufficiently for Privacy, cannot be visualized and does not leave the phone of healthy person.*

    **This is our recommended protocol if GAEN and GPS APIs can co-exist.**

    a. Details
        i. Same as in the case of points 1, but the location-time is quantized in a space-time to blur it before storing locally on the phone. For example, based on the local population density, the blur can be about 200m in a city or 1km in the suburbs. With the help of the user's own memory, the user can figure out where the exposure encounter could've happened.Even with this blurred location,
    b. Benefits
        i. A nosy employer cannot see specific location data
        ii. A sophisticated hacker who can reverse engineer the exact location because it is always quantized
    c. Issues
        i. Context could be reduced significantly and hence this method appears to bring the classic dogma of utility vs privacy tradeoff.
        ii. Context could also be wrong in some cases.
        iii. Possibility of On-the-fly attacks where the attacker knows precise location.

3. *GPS is encrypted with the RPI (rolling identifier) from the infected person*

    Details: “Upload what you heard” as in FindMy protocol

    Every user encrypts their own GPS with a key as the public key of the user nearby. A healthy user phone downloads DailyKey and computes the list of RPIs. The healthy phone checks for the intersection with the list of heard RPI using the existing GAEN API. The corresponding encrypted location data is decrypted using their own private key. Note that both UUID_x and PublicKey_x keep on rotating after every 15 minutes time interval. Rotating PublicKey_x while keeping the same PrivateKey_x in an efficient manner is non-trivial hence we propose to use the already built-in FindMyDevice protocol by Apple.

a. Algorithm: (X is a healthy person and Y is infected):
    i. X emits [UUID_x, PublicKey_x].
    ii. Y receives [UUID_x, PublicKey_x]
    iii. Y encrypts their own GPS as D = *Encrypt*(PublicKey_x, GPS) at the time of receiving and stores it locally.
    iv. Y gets identified as infected and uploads all [UUID_x, D_x].
    v. X pulls down the data from the diagnosis server and performs comparison, finds match with UUID_x and use its own PrivateKey_x to obtain:
    GPS_x = *Decrypt*(PrivateKey_x, D_x)
b. Challenges
    i. Scalability as this would need a lot of public keys to be generated.
    ii. An attacker can reverse engineer the key X

4. *GPS in existing GAEN protocol:*

Details: “Upload what you broadcasted” as in GAEN protocol (as well as MIT PACT)
In the following three proposals we extend the existing GAEN scheme to allow context sharing. We propose two different ways where one is based on Asymmetric key infrastructure while the other is based on Symmetric key infrastructure.

Our goal is to use the new Associated Encrypted Metadata (AEM) supported in GAEN. Current AEM is for sharing BLE signal strength. We assume we can add the GPS values to this metadata broadcast over BLE.

Using this protocol, **Alice can recover the GPS location of her contact (and hence her own location) only for locations for which she received the Exposure Notification**. She does not have access to the rest of her location history or the infected person’s location history.

a. Using Asymmetric key encryption - Algo Steps:
    Every device has a Rotating Public Key - RPublicKey and a single PrivateKey.
    This RPublicKey is different from the RPI that is used in GAEN. We aim to encrypt the GPS with this RPublicKey, and the health user can recover the encrypted GPS using the corresponding PrivateKey downloaded from the server.
    i. RPI = Rolling Proximity Identifiers broadcast over BLE as in GAEN

RPublicKey = Rotating Public Key - RPublicKey and a single PrivateKey
Infected phone Y emits [RPI_y, RPublicKey, *Encrypt*(RPublicKey, GPS_y)]
Healthy phone X, over BLE, received this
[RPI_y, RPublicKey, *Encrypt*(RPublicKey, GPS_y)] and stores locally

ii. If infected, Y uploads (DailyKey_y, PrivateKey_y)
iii. With GAEN, Healthy phone X, downloads (DailyKey_y, PrivKey_y).
iv. X from DailyKey_y, generates {RPI_y}
v. X finds the corresponding entry
[RPI_y, RPublicKey, and *Encrypt*(RPublicKey, GPS_y)]
vi. X obtains GPS_y = *Decrypt*(PrivKey_y, *Encrypt*(RPublicKey, GPS_y))

b. Using Symmetric key encryption - Algo Steps:
**This is our recommended algorithm with minimal change to GAEN.**
This does not change anything about the uploading of DailyKey or (TEK).
It changes the BLE payload to include GPS encrypted with DailyKey.
The *Encrypt* method could be AES (key size can be adjusted to use standard 128 or 256 bit size).

i. Infected phone Y emits
[RPI_y, *Encrypt*(DailyKey_y, GPS_y)]
ii. Healthy phone X, over BT, receive this
[RPI_y, G_y = *Encrypt*(DailyKey_y, GPS_y)] and stores locally.
iii. Once diagnosed as infected, phone Y uploads (DailyKey_y)
iv. With GAEN, Healthy phone X, downloads (DailyKey_y).
v. From DailyKey_y, phone X reconstructs RPI_y.
vi. Find the corresponding entry (RPI_y, G_y)
vii. Decrypts GPS_y = *Decrypt*(DailyKey, G_y)

c. Using Symmetric key encryption and **Consent after infection**.
We allow a Consent mechanism after the broadcast.
The user may have consented to broadcast BLE Ids and also shared their encrypted GPS. But the infected user may change their opinion to share only the BLE and not allow the healthy user to decrypt the GPS location. User keeps a 'Consent_secret' code, and this is appended to the DailyKey. The RPI is derived from DailyKey and hence BLE decoding is not impacted. But GPS is encrypted with the key that appends DailyKey_y with Consent_secret. If the inflected user refuses to upload the Consent_secret, the broadcast encrypted GPS coordinates cannot be recovered.
Rest of the protocol is the same as 4(b) above.

i. Infected phone Y emits
[RPI_y, *Encrypt*(DailyKey_y||Consent_secret, GPS_y)] (|| refers to the concatenation operator)

ii. Healthy phone X, over BT, receive this (RPI_y, G_y = *Encrypt*(DailyKey_y||Consent_secret, GPS_y)) and stores locally.
iii. Once diagnosed as infected, they upload (DailyKey, Consent_secret), where 'Consent_secret' is a unique secret key for allowing consent to be provided for decrypting location context or just provide exposure notification (Note: For brevity we are keeping consent_secret here as a non-rotating unique secret key but this can be changed in the upcoming version of the draft)
iv. With GAEN, Healthy phone X, downloads (DailyKey_y, Consent_secret).
v. From DailyKey_y, reconstructs RPI_y.
vi. Find the corresponding entry (RPI_y, G_y)
vii. Decrypts GPS_y = *Decrypt*(DailyKey||Consent_secret, G_y)

d. **Encrypted and Blurred GPS and post-infection Consent**
**This is our recommended algorithm if GPS is supported only in BT payload.**
This does not change anything about the uploading of DailyKey (TEK).
It changes the BLE payload to include blurred GPS encrypted with DailyKey.
It allows infected user to change the CONSENT as frequently as DailyKey is changed. Currently, DailyKey changes per day but in the future, they may change more frequently as in PACT, allowing more fine grained Consent by the infected user.

i. Infected phone Y emits
[RPI_y, *Encrypt*(DailyKey_y||Consent_secret, FGPS_y)]
(|| refers to the concatenation operator) and
FGPS_y is quantized GPS. Quantization based blurring of location can be fixed or vary from 100's of meters to 1km based on population density.
ii. Healthy phone X, over BT, receive this (RPI_y, G_y = *Encrypt*(DailyKey_y||Consent_secret, FGPS_y)) and stores locally.
iii. Once diagnosed as infected, they upload (DailyKey, Consent_secret), where 'Consent_secret' is a unique secret key for allowing consent to be provided for decrypting location context or just provide exposure notification
iv. With GAEN, Healthy phone X, downloads (DailyKey_y, Consent_secret).
v. From DailyKey_y, reconstructs RPI_y.
vi. Find the corresponding entry (RPI_y, G_y)
vii. Decrypts FGPS_y = *Decrypt*(DailyKey||Consent, G_y) which is a blurred location that is enough to provide context for the user but does not provide the exact location.

e. Benefits:
i. Only DailyKey is uploaded as in GAEN, so no change in upload protocol
ii. Minor change in BLE payload

        iii. GPS (encrypted) is available only on the proximate phone so there is little or no risk to a non-proximate person.
        iv. GPS history is invisible.
    f. Challenges:
        i. BT payload increases but GAEN payload has plenty of space

## Discussion:

We consider three different possible attack levels (The three levels are not mutually exclusive here) -

- **On the fly attack**: This scheme of attack is performed by an attacker acting either as a Healthy person or a future infected person. They can potentially do two kind of attacks:
    - Snoop on information.
    - Spread wrong information through the Bluetooth transmission.
- **Post processing attack**
    - In this setting of attack, the attacker tries to make sense out of encrypted and unencrypted information available after collection on their phone or force someone else to show these information present on their phone.
- **Distributed multi party attack**
    - In this scheme, multiple individuals align together to share their data with each other in a distributed way to attack the secrecy and privacy of individuals or groups of individuals.

There are different threat actors against whom protection is required -

- Nosy person looking at stored GPS trails because visualization is easy
    - Capabilities:
        - Can force the user to open the App and show any data visible on the screen.
- Hacker looking at stored GPS trails if available in raw format somewhere in the app
    - Capabilities:
        - Can reverse engineer their own App to inject code on top of APIs(This is only possible by jailbreaking the iOS and rooting the Android OS)
        - Can perform packet captures and snooping
        - Can not force the user to open App and share data with the attacker.
- State actors reverse engineering information using poorly encrypted trails (using side channel)
    - Capabilities:
        - Combines the capabilities of the above actors and in addition can leverage multiple sources and supercomputing capabilities for cryptanalysis.

From the GPS location, for an added context, the user may need to call a reverse geocoding API to find the street address or name of the business there. If performed naively, this API call will leak user location. The easiest way to resolve would be to perform regional map caching but this approach is beyond the scope of this document.

## Conclusion:

In this proposal we have outlined several ways of allowing context enabled contact tracing. We believe the context of location and time will allow citizens to take informed decisions and reduce panic.

In the spirit of respecting privacy, allowing consent, and delivering context within the Exposure Notification service, we advocate for the adoption of Proposal 4b.

## References

FindMy Protocols
- the iOS Security Guide including the details of how the public keys are derived
- the 2019 “Behind the scenes of iOS and Mac Security” BlackHat talk

MIT SafePaths documents
Contact Tracing: Holistic Solution beyond Bluetooth
https://www.youtube.com/watch?v=eoLE9OXvxiU
Slides: https://www.dropbox.com/s/olruh063zq51cxk/RaskarCOVIDSafePaths24Apr.pptx?dl=0

PACT Protocols from MIT
http://pact.mit.edu/

GAEN Protocols (Google Apple Exposure Notification)
https://www.apple.com/covid19/contacttracing

Contact Tracing: Holistic Solution beyond Bluetooth

## Contributors so far:

Ramesh Raskar
Abhishek Singh
Sam Zimmerman