PRIVACY PRESERVING AND INTEROPERABLE CONTACT TRACING AND EXPOSURE ANALYSIS USING BLOCKCHAIN

Version 1.0

Table of Contents

ABSTRACT

PPICT-EAB, an abbreviation of Privacy-Preserving and Interoperable Contact Tracing And Exposure Analysis Using Blockchain is a protocol that allows government agencies to control the spread of a pandemic through automated contact tracing and automated exposure analysis without violating the data privacy of citizens. This article describes the key concepts and algorithms used in PPICT-EAB.

INTRODUCTION

The world has suffered a huge blow due to the pandemic. Economies all around the world have crumbled and humanity is falling victim to unemployment and hunger. When will this crisis end? It's still an uncertainty. Till the 16th of April, 210 Countries and Territories around the world have reported a total of 2,100,970 confirmed cases of the COVID-19 and a death toll of 136,048 deaths. Many countries have implemented precautionary measures such as extensive lockdowns, closing borders, enforcing social distancing, contact tracing, etc to curb the spread of the pandemic. To effectively curb the spread of the disease, it is vital to do contact tracing of infected people and perform a large scale exposure analysis.

What is contact tracing? Why is it important?

Contact tracing is an invaluable tool for reducing the spread of infectious diseases. Its primary aim is to reduce a disease's effective reproductive number (R) by identifying people who have had exposure to the virus through a person who has been infected and establishing contact with them to provide quick detection, guidance, and adequate treatment. By putting a stop to virus transmission chains, contact tracing can aid in "flattening the curve" and reducing the peak burden of a disease on the healthcare system.

What is exposure analysis? Why is it important?

Studies say that coronavirus may continue to stay on surfaces for a couple of hours or up to many days. This may change according to different conditions (e.g. type of surface, temperature or humidity of the environment).

1. Airborne droplets - 3 Hours
2. Hard and shiny surfaces (Glass/Countertops/Plastic/Stainless steel) - Upto 72 hours
3. Porous surfaces (Cardboard/Paper/Fabrics) - Up to 24 Hrs

There is a large chance for coronavirus to persist in surfaces of facilities present in a hot zone. Because of this, a person who goes to a hot zone has a larger probability of being infected with COVID-19. For example, if a person who has tested positive for COVID-19 goes to a restaurant and touches the handrails, door handles, benches or any such object, the virus can be transmitted to these surfaces and could persist on such a surface based on the above-mentioned points. Meanwhile, someone else could visit the same restaurant and come in contact with any of these surfaces to be exposed to the virus. It is extremely important for governments to act in a proactive manner by making calculations according to these exposures

and subsequently identify micro hotspots and alert users to stay away from such hotspots. The owners of these infrastructure owners can also be notified to disinfect these infrastructures. This is a major component of exposure analysis which a lot of software systems out there don't take into account.

Automated Contact Tracing

Many organizations and startups have proposed solutions and technologies to fight COVID19. Most of the solutions involve the idea of automated contact tracing to identify the direction of the spread of the disease. The proposed solutions want to track the location details of citizens until the COVID-19 curve is flattened. A citizen's location details are tracked via the GPS present in their smartphones. Some solutions are designed to track the Bluetooth proximity data to identify people who came to close proximity with an infected person.

These solutions have caused privacy concerns in many parts of the world, especially in the European union. Commenting in a statement, Thierry Breton — the EU commissioner for Internal Market — said: "Contact tracing apps to limit the spread of coronavirus can be useful, especially as part of Member States' exit strategies. However, strong privacy safeguards are a prerequisite for the uptake of these apps, and therefore their usefulness. While we should be innovative and make the best use of technology in fighting the pandemic, we will not compromise on our values and privacy requirements." The Commission's top-line "essential requirements" for national contacts tracing apps are;

- Voluntary

- Approved by the national health authority
- Privacy-preserving
- Dismantled as soon as no longer needed

In this situation, several organizations have come up with protocols and compliances requirements for contact tracing applications. In the next section, we will take a look at some of the key concepts and ideas of prominent compliances and protocols.

An overview of existing compliance models

PEPP-PT

The main idea behind the creation of PEPP-PT was to follow the strong European privacy and data protection laws and principles. The founding principle was to make the technology available to as many countries, managers of infectious disease responses, and developers as quickly and as conveniently as possible. The technical mechanisms and standards provided by PEPP-PT fully protect privacy and leverage the possibilities and features of digital technology to maximize the speed and real-time capability of any national pandemic response.

PEPP-PT's primary "privacy-preserving" claim falls on the use of system architectures that do not require location data to be collected. Instead, devices that come near each other would share pseudonymized IDs — which could later be used to send notifications to an individual if the system calculates an infection risk has occurred. An infected individual's contacts would be

uploaded at the point of diagnosis — allowing notifications to be sent to other devices with which had come into contact.

PEPP-PT will support both centralized and decentralized approaches. The former meaning IDs are uploaded to a trusted server, such as one controlled by health authority; the latter meaning IDs are held locally on devices, where the infection risk is also calculated — a backend server is only in the loop to relay info to devices. Presently, contacts tracing apps that are not making use of a decentralized infrastructure won't be able to carry out Bluetooth tracking in the background on Android or iOS — as the platforms limit how general apps can access Bluetooth. This implies that users of such apps would have to have the app open and active all the time for proximity tracking to function, with associated (negative) impacts on battery life and device usability.

DP-PPT

A group of European privacy experts has proposed a decentralized system for Bluetooth-based COVID-19 contacts tracing which they argue offers greater protection against abuse and misuse of people's data than apps that pull data into centralized pots. The protocol — which they have named Decentralized Privacy-Preserving Proximity Tracing (DP-PPT) is designed to entail local processing of contacts tracing and risk on the user's device, based on devices generating and sharing ephemeral Bluetooth identifiers.

A backend server is used to push data out to devices — i.e. when an infected person is diagnosed with COVID-19 a health authority would sanction the upload from the person's device

of a compact representation of EphIDs over the infectious period which would be sent to other devices so they could locally compute whether there is a risk and notify the user accordingly. Under this design, there's no requirement for pseudonymized IDs to be centralized, where the pooled data would pose a privacy risk.

The system provides the following security and privacy protections:

- Ensures data minimization. The central server only observes anonymous identifiers of infected people without any proximity information; health authorities learn no information (beyond when a user manually contacts them after being notified), and epidemiologists obtain minimal information regarding close contacts.

- Prevents abuse of data. As the different entities in the system receive the minimum amount of information tailored to their requirements, none of them can abuse the data for other purposes, nor can they be coerced or subpoenaed to make other data available.

- Prevents tracking of non-infected users. No entity, including the backend, can track non-infected users based on broadcasted ephemeral identifiers.

- Graceful dismantling. The system will organically dismantle itself after the end of the epidemic. Infected patients will stop uploading their data to the central server, and people will stop using the app. Data on the server is removed after 14 days.

The app does not aim to provide these functionalities: -

Tracking infected patients: As soon as infected patients report themselves, the app does not try to track them, nor does it provide a mechanism to ensure that they comply with medical orders. Recall that the goal of the app is to avoid asymptomatic users unknowingly spreading a disease. Diagnosed users are assumed to be responsible and take precautions if necessary to go into the public, for instance to a doctor's appointment. Therefore, we do not attempt to detect contacts with infected patients after their diagnosis nor do we attempt to detect or prevent misbehavior. The reason is that the gain in utility (one irresponsible person being under control) does not justify the loss of privacy for other well-behaved infected users. Moreover, this is not a location-tracking app and cannot determine when a user is "in public."

- Finding hotspots or infected users' trajectories: the app does not attempt to identify locations that have a concentration of infected people. This is a design decision. We limit the purpose of the application to the two goals specified above, which enable us to collect and process very little data. In particular, it avoids collecting location data, which is highly sensitive and very difficult to publish in a privacy-preserving way.

Bluetrace

BlueTrace is a protocol for logging Bluetooth encounters between participating devices to facilitate contact tracing, while protecting the users' personal data and privacy. When two participating devices encounter each other, they exchange non-personally identifiable messages that contain temporary identifiers. The identifiers rotate frequently to prevent third

parties from tracking users. The user's encounter history is stored locally on their user's device; none of this data can be directly accessed by the health authority. If a user is infected or is the subject of contact tracing, they will be asked to share their encounter history with the relevant health authority with the use of a PIN. (A verification code may optionally be provided, to authenticate the health authority official's request.) Only the health authority has the ability to decrypt the shared encounter history to obtain and use personally-identifiable information to filter for close contacts and contact potentially infected users. BlueTrace is designed to supplement manual contact tracing by addressing its main limitation: an infected person can only report contacts they are acquainted with and remember having met. BlueTrace could also allow for contact tracing to be more scalable and less resource-intensive. BlueTrace also allows a federated network of credentialed health authorities to each maintain distinct user bases, while allowing for contact tracing between users from different health authority jurisdictions.

Limitations of existing models

Even though PEPP-PT, DP-PPT and Bluetrace models work very well for privacy-preserving contact tracing, these models do not take exposure analysis into account, which is a vital tool for suppressing the spread of the disease. As explained in introduction, It is extremely important for governments to make calculations based on these exposures and dynamically create micro hot zones as well as macro hot zones and alert users to stay away from such hotspots, also these infrastructure owners could be notified to disinfect these infrastructures. This is a key component of exposure analysis which most of the software systems out there fail to consider.

Privacy-Preserving and Interoperable Contact Tracing And Exposure Analysis Using Blockchain

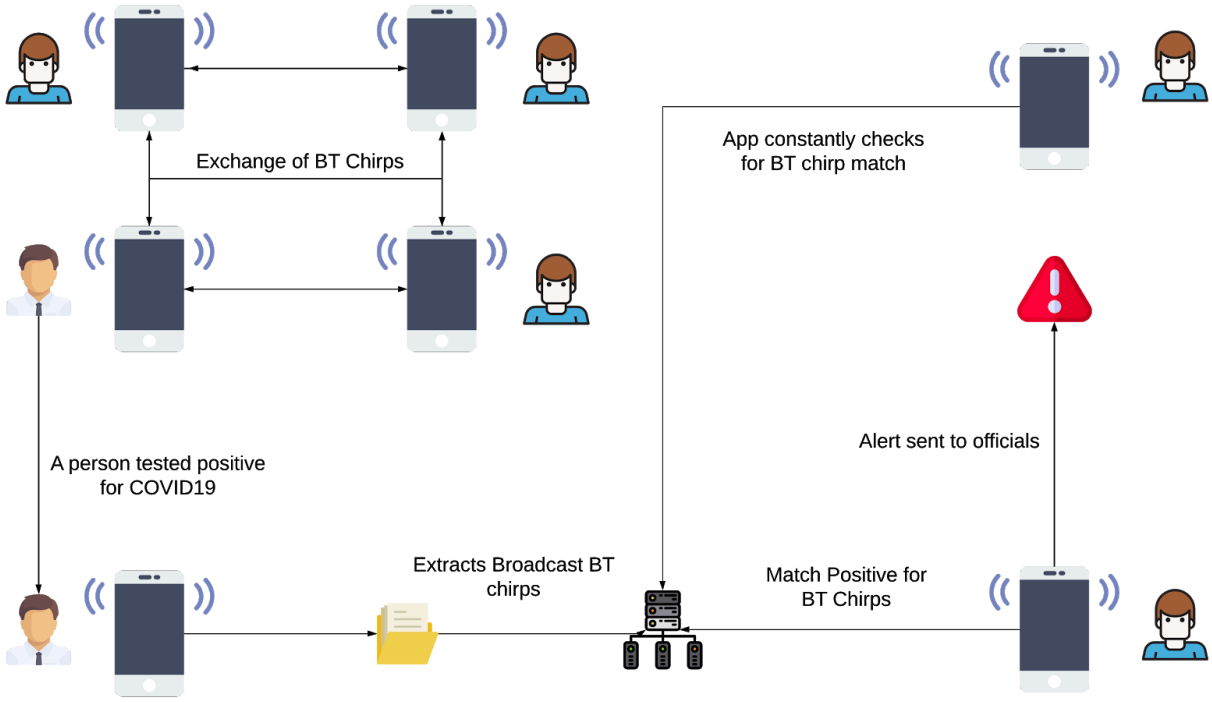
Privacy-Preserving and Interoperable Contact Tracing And Exposure Analysis Using Blockchain or PPICT-EAB suggests a system that can be used by citizens on a large scale. It would allow secure, privacy-preserving and interoperable contact tracing and exposure analysis of citizens. The system is designed to help governments to conduct automated contact tracing and exposure analysis by lowering risks associated with privacy and security for individuals.

Privacy-preserving contact tracing

Identifying the people who have come in close contact with an infected person is the main goal of contact tracing. When a person installs the application, the Bluetooth of their smartphone starts broadcasting random codes (a.k.a chirps) to nearby Bluetooth devices. The nearby Bluetooth devices capture these chirps and log them into the application. Likewise, all smartphones under close proximity share chirp between each other. PPICT-EAB primarily focuses on defining the protocol for exposure analysis and interoperability of citizen daya. PPICT-EAB contact tracing is inspired from contact tracing methodology defined by Bluetrace protocol, which satisfies the need for privacy preserving contact tracing.

By exchanging chirps over Bluetooth the user application log encounters with each other. Chirps can not reveal the identity of the users in order to protect their privacy. However, these messages can not include static identifiers, to prevent third parties from tracking users over time. When a person is tested positive for COVID-19, government agencies can request the

user to upload the broadcast chirps from their mobile application. Each chirp consists of a UserID, created time, and expiry time encrypted symmetrically with AES-256-GCM and then Base64 encoded. Each chirp is generated with a random Initialisation Vector.



These chirp codes from all infected people will be uploaded to a common database. Every citizens' application checks with this common database to see if any of the listed chirps matches with the logged received chirps in the application. If there is a match, the application will notify the user about the potential contact with an asymptomatic carrier and alert them to contact health officials via the app.

Privacy-preserving exposure analysis

PPICT-EA suggests a privacy preserving exposure analysis using user location data from GPS. Through this approach, the user's GPS data is not transmitted to the server side. Instead, the user application pulls the batches of hot zone information from the server to perform a decentralized exposure analysis.

Geo-fencing

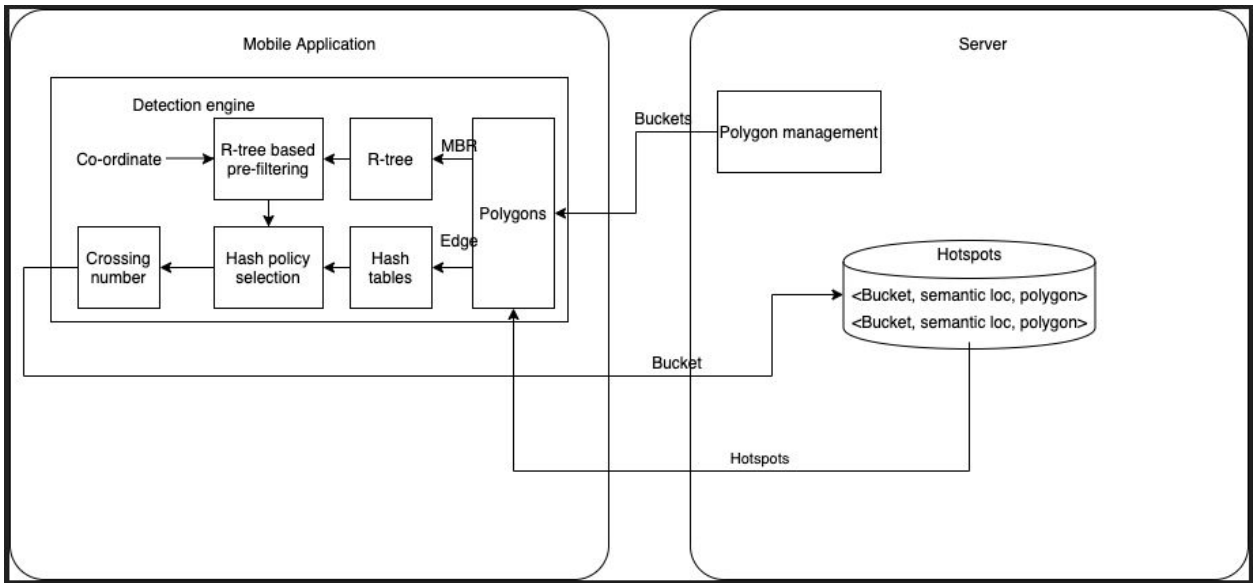
The proposed framework consists of two main parts, which include:

Polygon management: Polygons of geo-fences are kept in two structures with different details. MBRs of polygons, as rough approximations of polygons, are stored in an R-tree5 . Edges of a polygon are stored in hash tables. To allow the updation of polygons, each polygon possesses its own hash tables. Several hash tables are used in parallel. Inside a hash table, a large bucket is split into sub-buckets and sorted, and these buckets/sub-buckets are organized in a bucket tree.

Pairing engine: R-tree is used to quickly detect whether a point is inside the MBR of any polygon in the filtering stage. Buckets in all tables associated with the point are found when a point is inside the MBR of a polygon, and the one with least number of edges is chosen. Then, a scheme corresponding to the hash policy is used to perform the CN algorithm (binary search if

edges in this bucket are sorted and thorough search of all edges in this bucket otherwise) that determines whether the point is really inside the polygon.

In a standard geo-fencing application, point positions are changed much more frequently than those of polygons. Taking this into account, we try to build hash tables for polygons during system spare time and shift some computation cost from the refining stage (which has a real-time requirement) to this spare time. The cost of building hash tables is justified by the fact that hash tables, once built, can be used for many points. The figure below shows the architecture of geo fencing used for exposure analysis.



Bucket Selection and Hybrid Hashing Policy are referred in section 3.2 and 3.3 in detail in the paper: Efficient Geo-Fencing via Hybrid Hashing: A Combination of Bucket Selection and In-Bucket Binary Search (http://www.tang.cs.uec.ac.jp/publication/15-TSAS-GeoFencing.pdf)

The mobile application of a user periodically saves their location coordinates in the phone as an array <semantic location>, mobile phone also has a list of polygons and Buckets pre-loaded in the application using R-tree we find if the point is inside of which polygon and then find buckets associated to the point. This bucket is then sent to the server and a list of polygons of hotspots are sent back which is close to the identified bucket. This list is saved into the application and till the bucket changes the coordinate of the user is checked across this received list of the polygons to notify if the person is inside a hotspot or not in the mobile application.

Interoperability

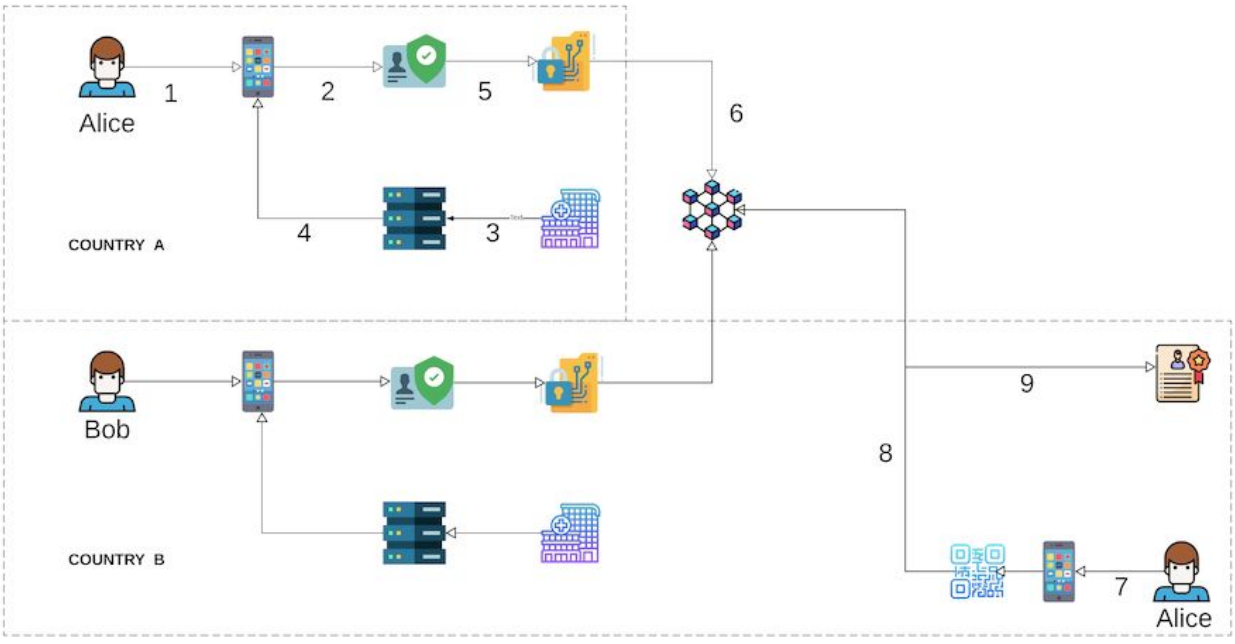
Automated contact tracing and exposure analysis can help a country to flatten the curve. But, for the country to open up borders, they should be confident that none of the incoming travelers are infected with COVID. For that, it is essential to store the citizens' health status data of every country or at least a consortium of countries in a common database. Obviously, storing all data in a common DB would raise data security concerns. No country would agree to store their citizens' data in a server located in a different country simply because they don't trust each other. This is where the importance of decentralized ledger technology comes into play.

As defined by Investopedia, A distributed ledger is a database that is consensually shared and synchronized across multiple sites, institutions or geographies. It allows transactions to have public "witnesses," thereby making a cyberattack more difficult. The participant at each node of the network can access the recordings shared across that network and can own an identical copy of it.

World countries or a consortium of countries can open up borders for each other based on a health-card based access system. Blockchain technology can facilitate the consortium to do the following,

- No country has ownership of the data, but data is distributed among all members of the consortium.
- Citizen data stored in the network is encrypted and a private key is required to access the data.
- A citizen is the only custodian of their private key.
- No one can access a particular citizen's data without the permission of the citizen

How such a system works, is explained in the infographic below.



Explanation: The above diagram shows how privacy-preserving interoperability can be attained. In the diagram, Alice is a citizen of country A.

Process 1: Alice installs the citizen side application to her smartphone.

Process 2: With automated contact tracing and exposure analysis, the system generates a health card for Alice. Health-card contains her health status, identification photo, and her name.

Process 3: The local health department uploads the recent pandemic data to their server. This data includes Bluetooth broadcast chirps of newly identified infected people, GPS data of newly identified hot zones.

Process 4: The server pushes the new data to Alice's application. This data is processed at Alice's application for contact tracing and exposure analysis.

Process 5: The health-card generated for Alice is encrypted by the system. And only Alice has the private key to decrypt this data.

Process 6: The encrypted data is stored in a blockchain network.

In Country B, the same processes apply to all citizens. When Alice travels from country A to country B,

Process 7: Alice can her health-card QR code at the security checkpoint.

Process 8: Upon scanning the QR code, the system receives a public key of Alice and sends a data access request to Alice. If Alice approves, the system fetches her health-card from Blockchain network

The system has three primary components:

1. smart contracts
2. citizen side application
3. authority side application

The health-card data stored on the blockchain network supports the concept of a self sovereign identity. This means that the users shall retain complete control over their own data. The data can be accessed through a private key that can be found in a secure enclave in the citizen side application. Users can gain access via local biometric authentication whenever the key is used to sign. Some of the main features of this system include,

- A request must be cryptographically signed to impart user data on behalf of the server application.
- Requests are sent as a JWT to your users via a QR, push, or another transport
- A URL will be included in the client app where the response is returned from the user. This can be a https url or a custom app url which receives the response.
- The application makes use of ECDSA (Elliptic Curve Digital Signature Algorithm) for it's public-key cryptography.

Symmetric encryption is used to encrypt the health-card data. This is carried out through the creation of personas; which map out a user's data to the user's public address. The data is stored in IPFS and encrypted. The IPFS file paths are stored in the blockchain network. Data can be shared with applications on an attribute by attribute basis, and

encrypt the dataset with the public key of the entity they'd like to share with, so that only they can decrypt the message. Data sharing can happen when an authority scans a QR code with data that follows an expected format which lets the user approve the request to share data, and submit the transaction to the blockchain so that the application can get the files from IPFS and decrypt the data.

Permissioned Data Retrieval

- In case we would like to share data with only identities X, Y and Z. First, we would need to create a random symmetric key k, and encrypt A symmetrically. We denote this ciphertext sym(k, A).

- After this, we must make the assumption that X, Y and Z each have a public encryption key. Let asym(U,V,d) denote the asymmetric encryption between identities U and V of some data d.

- Specifically asym(U,V,d) = sym(DH(U,V),d)

- Here DH(U,V) is a symmetric key generated from the public key of U and the private key of V using a Diffie-Hellman key exchange.

- Create a random and ephemeral public/private key pair R, and create the data blob made up of (sym(k,A), asym(X,R,k), asym(Y,R,k), asym(Z,R,k))

- This data blob can now be decrypted only by the identities X, Y and Z in order to retrieve A, and we store this data blob among our attestations where it can be retrieved. Note that since the key k is small, the overhead of encrypting to several identities is small, even though the attestation A might potentially be large.

References

1. http://www.tang.cs.uec.ac.jp/publication/15-TSAS-GeoFencing.pdf
2. https://bluetrace.io/static/bluetrace_whitepaper-938063656596c104632def383eb33b3c.pdf
3. https://github.com/pepp-pt/pepp-pt-documentation/blob/master/12-proximity-measurement/distance-measurements-and-classification-20200406.pdf