Using decentralized social trust as an alternative way to prove someone’s address

Mesquita Borba Maranhao M, Suzana; Seigneur, Jean-Marc

Using Decentralized Social Trust as an Alternative Way to Prove Someone's Address

Suzana Mesquita de Borba Maranhão Moreno
Geneva School of Economics and Management
University of Geneva
Geneva, Switzerland
Suzana.Mesquita@etu.unige.ch

Jean-Marc Seigneur
Geneva School of Economics and Management
University of Geneva
Geneva, Switzerland
jean-marc.seigneur@unige.ch

***Abstract*—The traditional way to prove someone's address using formal documents like utility bills may not be feasible for some people, like those living in very poor neighborhoods, because they do not have these documents. In this paper, we propose an alternative way to prove someone's address using a decentralized social trust solution. Because our design choices, this solution is able to work offline and does not need a logically centralized repository of all issued proof-of-address, in oppose to what would be achieved by using existing accretionary ID solutions. We validated this proposal by building a mobile application, using it in a real experiment in a Brazilian favela, and collecting mobile data. We also interviewed 20 people to complement our validation and help to guide the next steps of this work. The experiment showed that the solution is viable and easy to use. It is possible to adopt an approach like the one proposed to prove other facts, like gender, sex and income. These proofs may be used for different initiatives, like social programs, purpose-driven lending or other decentralized finance services.**

Keywords—Proof-of-address, Decentralized Social Trust, Accretionary Proofs, Mobile Application, Certifiers.

I. Introduction

We are required to prove where we live in different situations in our routine. For example, when opening a bank account, registering children in a public school or applying for a job. The traditional way of proving someone's address often relies on using formal documents like a utility bill[1] or another document issued by an official entity linked to the government sector or a trustful institution in a jurisdiction [1].

Presenting these documents may be a difficult requirement to satisfy in certain scenarios. Some people live in places with no formal address or with no access to utility services. This is for example the case in Brazilian favelas [2]. Even if the chief of the household has access to a paper-based proof in his/her name, other residents may not be able to reuse the same document. Also, many paper-based proofs require a recent document to guarantee that the person still lives in the place. For example, in Brazil it is common to ask for a proof-of-address issued up to 3 months before.

There are additional drawbacks of this paper-based approach when dealing with digital processes. First, a utility bill is sometimes received as a physical paper and it is necessary to digitalize the information by typically taking a photo or scanning the paper. It may be an issue for poor people that do not know how or do not have an equipment to do it. Second, even if the utility bill is received in digital format (for example, by email), it may the case that it is just the equivalent of a paper bill, and the address is an unstructured text in the file. Third, these documents are also in general easily faked, e.g., a user can change an image to forge a proof that he/she resides in a different place.

Some institutions have well-defined processes to verify these documents. This is the case for financial service providers, that are required to have KYC (know your customer) and AML (anti-money laundering) processes in place [3]. These are high-cost processes, leading to more expensive financial services [4]. This issue is even bigger in the context of cross-border business processes, since the premise of the well-known trustworthy institution may not be easily checked anymore, e.g., an international business partner may not know how to evaluate the credibility of an overseas issuer of a document.

The requirement of proof-of-address is correlated to financial exclusion. In fact, of those without formal financial accounts, 25% attribute their exclusion to lacking the necessary documentation such as proof-of-address [5]. To balance the need for financial inclusion with the need for AML compliance, FATF recommendations do not require paper-based proofs in low-risks transactions [3]. This enables alternative ways to comply with financial service requirements.

This paper discusses alternative ways to prove someone's address and proposes a solution by using decentralized social trust [6]. The proposal is especially useful to be applied in some scenarios previously discussed, like people in poverty without formal documents trying to comply with digital process requirements. The remainder of this paper is divided as follows. Section 2 discusses concepts related to digital identity systems and related work. Section 3 details the technical proposal of this paper, Section 4 discusses some details of its implementation while Section 5 explains how the proposal was validated with a real experiment and users' interviews. Section 6 examines conclusions derived from the work and future work. Finally, Section 7 presents some references linked throughout the paper.

II. Background Concepts and Related Work

This section positions this work by briefly describing digital identity system and comparing it with existing solutions.

A. Digital identity systems

Digital identity systems are typically divided in the following phases: (a) enrollment, which includes identity proofing, de-duplication (ensures the uniqueness of each

[1] Some people may argue that the definition may not be clear in some cases, for example a child of a separated couple who stays some days with each parent. This is in fact an issue and it is possible that one person can be able to proof that he/she lives in more than one place.

XXX-X-XXXX-XXXX-X/XX/$XX.00 ©20XX IEEE

identity in the system) and credential issuance; (b) authentication, when one can assert an ID to access a service or to transact and (c) authorization to determine which services the authenticated party is authorized to access [7]. Instead of being a one-time event, the enrollment step may be a process when dealing with accretionary ID. Accretionary ID allows undocumented users to establish an initial ID with little or no supporting information. Such IDs would start with relatively little confidence that one is who they say they are but additional supporting proofs may be added over time, increasing ID trustworthiness [7]. The activities to create a proof-of-address are considered a part of enrollment in a digital identity system.

B. *Existing solutions*

There are some existing alternative ways to proof where someone lives without using official documents. A first one is using a declaration of address, which may be signed by a third-party [8] [9]. This approach is linked to a specific jurisdiction and many times it requires notarized signatures. Still, it is not a digital native solution, demanding some steps to be integrated with digital processes in a structured way.

A second approach is to make use of location technologies, like GPS (Global Positioning System). These technologies can help to determine the location of a device and be used to infer that the user of the device is in a specific place [10]. One could assume that the user of the device lives where the device usually stays at night. This approach is not jurisdiction-dependent and it is natively digital. However, even though authentication technologies may be used in the device to check the person holding it at a specific moment, proving that a device is in a place is not the same as proving that the user is also there. Finally, there are some ways to fake location results. [11] [12].

A third approach is to use an accretionary ID. There are many examples of models using decentralized identity and verifiable credentials [13] [14] that enable the user to prove ID-related topics about himself/herself [15], including proof-of-address. Since these solutions are trying to approach the entire identity system, they need to solve the issue of de-duplication [7] using a logically centralized solution. For that, these solutions can use a database or a distributed ledger technologies (DLT) [16]. Some examples of DLT-based identity systems are BlockCerts [17], Kaytrust [18] and Rem ID [19]. DLT brings some particularities to these solutions. First, connectivity is a requirement to settle transactions. Second, in the majority of DLT networks, the sender of a transaction needs to individually pay for his/her transaction. To avoid this usability issue, some apps and/or their used DLT network may have an additional layer to relay users' transactions (sometimes called metatransactions) [20]. This approach is still new, introducing technical risks and software complexity. An additional point to these three cited solutions is that they allow the user to request a certificate to a permissioned list of issuers like, for example, a university or a government entity. It has not been found in these apps a function where a normal user has a native process to become a new certifier.

This paper proposes an accretionary proof-of-address digital solution based on decentralized social trust aiming to have no predefined list of certifiers. This work does not approach to prove that a person is really who he/she says he/she is nor to solve de-duplication of users. If necessary, some additional proofs, paper-based or using alternative ways – e.g., proof-of-humanity[2], BrightID[3], should be used. Because of these design choices, this solution is able to work offline and does not need a logically centralized repository of all issued proof-of-address, in oppose to what would be achieved by using existing accretionary ID solutions.

A drawback of not having a logically centralize repository is that it is not possible to revoke previously issued certificates. The technical solution described in this paper minimizes this issue by including a timestamp in the issued certificates. External applications may define custom expiration dates for these certificates. This is not enough for all scenarios and existing works discussing how to enable revocation in PKI (Public Key Infrastructure) management system [21] [22] are a source for inspiration. In the future, we will develop a plug-in to register issued certificates to a logically centralized repository like a DLT. This plug-in should be used in some scenarios, including when the revocation list is essential, overcoming the initial drawback. Note that the social interaction necessary for issuing certificates will still work offline. The plug-in can be triggered at a later moment, only when the user is connected.

Finally, our solution can work in a complementary way with the approach of location technologies, increasing the overall trustworthiness of the solution.

III. Technical Proposal

The technical proposal of this paper is an application that enables people to certify where other people live by using decentralized trust. The trust is anchored at people with high reputation inside a community, named as Certifier Manager. A Certifier Manager can be for example a priest, a police officer, a bank, a representative of the government or an NGO. These high-trust users delegate trust by creating new certifiers and by issuing certificate for other users.

The main functions of this proposal are: (1) to assign a Certifier Manager, (2) to make other people Certifiers, (3) to issue a proof-of-address and (4) to manage its own certificates. These functions are explained below.

The first function is to assign a new Certifier Manager. In the bootstrap, the proposed solution relies on a specific role called Admin to select who should be a Certifier Manager and give him/her a secret code. Using a mobile app, the chosen person asks to be a Certifier Manager by entering his/her personal information and the secret code. This action generates a request in a backend service and an Admin approves the request by using a master key to sign the Certifier Manager certificate. Future versions will change the Admin's role by fully automating who can be entitled as Certifier Managers using a computational trust engine [6]. Note that many Certifier Managers may co-exist on the app.

The master key is an asymmetric cryptographic key pair managed by a backend service. The public key of the Admin must be well-known so the signature in a Certifier Manager certificate may be validated by anyone. The mobile app also manages an asymmetric cryptographic key pair for its user. The private key of this latter key pair is stored only in the

[2] https://www.proofofhumanity.id/

[3] https://www.brightid.org/

device with the goal to sign certificates while the public key is used to identify the mobile app user. See more details on key pairs and certificate format in Subsection 3.A.

The second main function is to make other people Certifiers. A Certifier is chosen by the Certifier Manager to assign proof-of-address to regular users. There is a set of steps to make this happen in online or offline modes. If both parties (a user who wants to be a Certifier and a Certifier Manager) are online, the flow can go in the online mode.

The online mode is presented in Figure 1. The first step is done by the person who wants to be a Certifier (referred here as candidate), by filling his/her name and timestamp using the mobile app. The filled data is saved on his/her own local device linked with the user public key and automatically synchronized with the remote storage too. There are some ways to forward the saved data to the Certifier Manager. This solution opted for the creation of a simple QRCode [23] containing the value of the candidate's public key. The Certifier Manager uses the mobile app to read this QRCode, seeks on the remote server the necessary additional information, creates the certificate body using a predefined format (see subsection 3.A) and signs the desired certificate. This new certificate is saved in the Certifier Manager local device and automatically synchronized with the remote server. In the last step, the candidate uses the mobile app to synchronize his/her local storage with the remote server to finally get his/her certificate.

A careful reader may notice that a central repository was used and may imagine that this repository can also be used for de-duplication. However, this central repository is used to help in the certificate exchange and does not need to have all certificates, as better discussed in the offline mode.

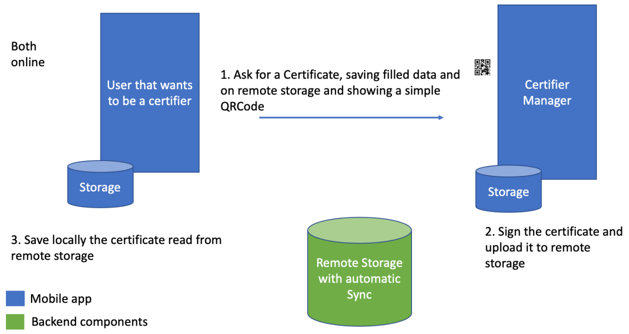

Both online
User that wants to be a certifier
Storage
1. Ask for a Certificate, saving filled data and on remote storage and showing a simple QRCode
Certifier Manager
Storage
3. Save locally the certificate read from remote storage
Remote Storage with automatic Sync
2. Sign the certificate and upload it to remote storage
Mobile app
Backend components

Figure 1: Certificate exchange - online mode

If any part is not online, the exchange needs to happen in offline mode, as represented in Figure 2. This mode allows people to interact with each other without the need for a central authority and without connectivity.

As it happened in the online mode, the candidate fills his/her name and timestamp using the mobile app in the first step. The filled data is saved on his/her own local device linked with the user's public key. Since it is the offline mode, the candidate needs to create a QRCode containing all necessary information to the Certifier Manager (in contrast, in online mode, the Certifier Manager could get some information reading them directly in the remote server). An important issue here is to minimize the number of bytes to be encoded to generate a QRCode manageable by low-capacity phones, as discussed in the next session. The Certifier Manager uses the mobile app to read this QRCode, creates the certificate body using a predefined format and signs the desired certificate. This new certificate is saved in the Certifier Manager local device. Since it is the offline mode, the Certifier Manager uses another QRCode to return the information to the candidate. A simple approach would be to show the entire certificate in the QRCode but this is not the best way because it is possible to create a smaller QRCode containing only information the candidate still does not have, which is the Certifier Manager's signature in the new certificate. In the last step, the candidate uses the mobile app to read the QRCode and extracts the signature of the certificate. The candidate does not have the public key of the Certifier Manager, but it is possible to compute it based on the original information sent to the Certifier Manager to be signed (i.e., name, timestamp and the public key of the candidate) plus the received signature. This is then used to mount the certificate using the predefined format and save it locally. In sum, the entire process happens offline, and each part can synchronize with the remote server when connectivity is available.

The third main function is to issue proof-of-addresses. The description of this function is similar to the second one and may occur in online or offline mode. The main difference is that the user also needs to type his/her own address as part of his/her personal data. A proof-of-address certificate may be issued by any user that has a Certifier's or Certifier Manager's certificate to a regular user of the mobile app.

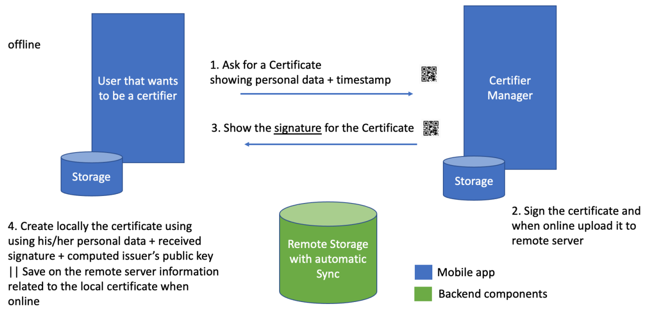

offline
User that wants to be a certifier
Storage
1. Ask for a Certificate showing personal data + timestamp
3. Show the signature for the Certificate
Certifier Manager
Storage
2. Sign the certificate and when online upload it to remote server
4. Create locally the certificate using using his/her personal data + received signature + computed issuer's public key || Save on the remote server information related to the local certificate when online
Remote Storage with automatic Sync
Mobile app
Backend components

Figure 2: Certificate exchange - offline mode

The last main function is the user's support to manage his/her own certificates. The mobile app offers the user the possibility of viewing his/her own certificates as well as the certificates of the Certifiers involved in the signature of his/her certificates. This is known as the parent path rule. For example, if user A has a proof-of-address certificate that was issued by Certifier B, whose Certifier's certificate was issued by the Certifier Manager C, the user A will be able to see in his/her device not only her proof-of-address certificate but also the Certifier's certificates of users B and C. At first, the user A does not have in his/her own local storage the information of all these certificates, but the mobile app will fetch the necessary data from the remote server to present it when necessary. After the first fetch, these certificates will also be saved on the local storage of user A. Besides these certificates linked to the parent path rule, a user does not have access to view any other certificate. Two points about this visualization should be highlighted. First, since the Certifier Manager can also issue proof-of-Address certificates directly, the app needs to check if the issuer is a Certifier or Certifier Manager to provide the correct navigation. Second, it is true that the remote database has an essential role to fetch Certifier's certificates, but it can be improved in the future by for example adding a function to enable the importing and exporting of certificates.

Still related to the function of managing his/her own certificates, a user may also remove certificates previously issued to him/her. After being removed from his/her local storage, future synchronizations will also remove the certificate from the remote server. A user can never remove certificates from the remote server not issued to him/her. Neither can an issuer remove previously issued certificates for other users.

In the bootstrap, this technical solution can be compared to a PKI. Indeed, the roles Admin, Certifier Manager and Certifier may be seen as a hierarchical chain of trust for a domain-specific application. In the future, we envision to remove the Admin role and to introduce non-hierarchical relationships.

A. *Certificate Format*

Both the user who is requesting a certificate and the Certifier have the mobile app installed in their device, so they use the same rules to create and read certificates in predefined formats. The Figure 3 shows the format of a proof-of-address certificate on the left. The signature field is determined by the signer using his/her private key to sign the content data field containing subject id (public key of the user who asked for the certificate), subject name, subject address and timestamp. The field signerId is equal to the certifier's public key and it is essential to verify that the signature is correct.

The word "subject" was inspired in the W3C DID subject definition [13] and refers to the person who requested the certificate. The proposed solution uses the timestamp from the device of the user asking for the certificate. In this way, the Certifier can confirm if it is a valid timestamp when signing the certificate. Another option would be to determine the timestamp considering an external source of clock like an online clock. Unfortunately, it could not work on the offline mode, so it was not used. The timestamp can be used by external applications to define expiration dates to these certificates, as already discussed. The address is stored as structured data. It can be even improved in the future to deal with structured subfields and GPS coordinates.

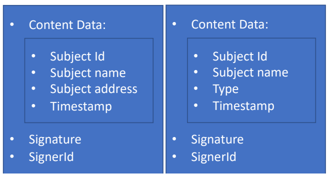


Figure 3: Schema of a proof-of-address certificate (left) and a Certifier certificate (right)

Figure 3 also shows the format of a Certifier's certificate on the right. Comparing with proof-of-address certificate, the Certifier's certificate does not have the subject address and has a field named "type". This field is used to indicate if this certificate is for a Certifier or Certifier Manager.

We decided not to include meta-fields in these formats. For example, there is no indication of what type of key or algorithm was used to generate the signature. Our proposal is to generate a very light format able to be easily exchanged offline and suitable to be in a second moment used by an application which faces limitation in terms of storage like public DLT applications. We follow the software design paradigm called "convention over configuration" [24]. New fields may be added in the future if one wants to use the format not following the convention. A source of reference to follow if we decide to add new fields is W3C Verifiable Credentials [14].

All asymmetric key pairs and signatures are compatible with Ethereum blockchain network [25]. The public key is stored in compressed form to minimize storage needs [26]. The choice of Ethereum-compatible key pair opens new ways to expand the proposal in at least two ways. First, to expand the app to a DLT wallet able to send transactions containing information about the certificates to blockchain applications. E.g., to comply with KYC/AML requirements in decentralized finance applications [27]. Second, to create a business model – e.g., using a *tokenomics* linked to decentralized finance applications – to motivate Certifiers to behave honestly and increase their own reputation because they can earn more with fees generated by their issued certificates than by cheating.

IV. Implementation of the Proposal

Figure 4 presents the main conceptual technical modules of the implementation. There are three layers. The lower one is responsible for managing data, and is composed of the certificates, personal data used to constitute the certifications and wallets. The mobile application creates the asymmetric keys of the user wallet when the user first loads the application, which introduced a significant delay in some devices. This was minimized by showing a specific screen with a spinner and an explanatory message. Once created, the private key linked to generated wallet is saved locally and is loaded every time the user reloads the application.

The middle layer is called Business Rules and contains three components. The service component contains business rules mainly linked to generating, managing and validating certificates. The internationalization component enables the app to work with multiple languages (and actual implementation in English and Portuguese). The Logger/Analytics component is responsible for collecting statistics about the use of the application. The collected data is stored in Firebase [28] and periodically transferred to an external analytics database.

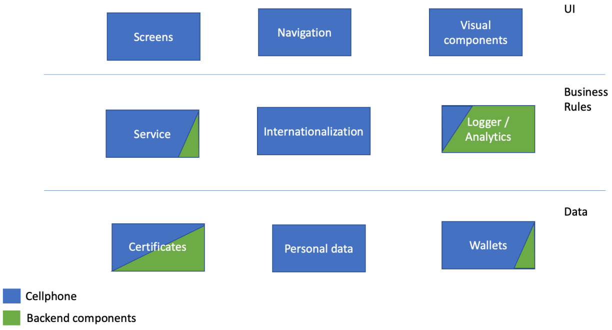


Figure 4: Main technical modules

The upper layer is the user's interface. The mobile app has some tabs with internal independent navigation. Visual components are reused on many screens.

The requirement that this app may be expanded to deal with different types and formats of certificates was a relevant input in the implementation. Any inclusion, change or removal of certificates mainly impact the technical components named Service and Certificates as well as the UI layer in Figure 4.

There are two main set of functions in the backend. One is to perform the Admin role of approving Certifier Managers (including parts of the components named Service, Certificates and Wallets). As discussed, Admin is a temporary role, used to bootstrap the model. The second part is called Logger/Analytics, which was used to monitor the system and it is not essential to the overall system. So, in the future, the backend as described in this paper may be removed to the overall solution.

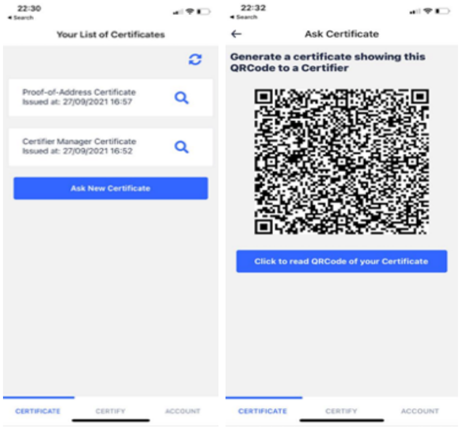
The mobile app was coded using React Native, so it can be used on Android and IOS, both smartphones and tablets. Admin functions were implemented using nodeJS. From Firebase, it was used the database Firestore and Statistics modules. The analytics database was Google Cloud Big Query.

QRCode was used as a tool to transfer information between two devices, both in online and offline modes. QRCodes can be used in many sizes, bigger sizes may store more data [23]. Many mobile devices come with applications able to read QRCodes, but the proposed solution includes support to generate and read QRCodes inside the mobile app. In this way, information can be easily integrated with other features of the software. The used default QRCode size is 256 x 256, the same used as default configuration in easyqrcode [29] and in Jean-Marc work [30]. Bigger sizes would introduce visualization issues on small-screen devices.

The first initial tests were executed using the React Native simulator Expo Go4 on an iPhone 11, an iPhone XR and an Android LG K22 cellphone. The tests confirmed that the more information is coded in the format, the more difficult it is to work in practice. Tests showed that more than 300 characters introduce a significative delay mainly when generating a QRCode. This result was consistent with some QRCode generators that advise using QRCodes with less than 300 characters[5] [6]. As a result, the QRCodes source data were reviewed to make sure that data transferences contain fewer characters than this maximum reference of 300.

The Figure 5 shows on the left side a screen of the developed app with a list of certificates that a user has on his/her device. There are two certificates. The first one is a certificate of Certifier Manager. Because he/she has this certificate, this user can sign new proof-of-addresses or Certifiers certificates to people using the app on other devices. The second certificate presented in the left image is a proof-of-address, which was signed by another Certifier (or Certifier Manager) using the app on another device. It is possible to see the details of these certificates by clicking on the magnifying glass icons on the right side of each certificate.

Figure 5 on the right side shows the user asking for a proof-of-address certificate using the app on offline mode. The way to get on this screen is by clicking on "Add New Certificate" on the image on the left, filling in the necessary personal data and clicking for generating a QRCode with the user's personal data. The presented QRCode codifies the necessary data to be read by a Certifier. Using the tab "Certify" (note that the app has three tabs on the bottom – "CERTIFICATE", "CERTIFY" and "ACCOUNT"), a Certifier reads this presented QRCode using another device, signs the proof-of-address certificate and generates as response another QRCode. Then, using the button presented on the right image ("Click to read QRCode of your Certificate"), the initial user is finally able to read his/her proof-of-address certificate without needing any network access.



Proof-of-Address Certificate
Issued at: 27/09/2021 16:57

Certifier Manager Certificate
Issued at: 27/09/2021 16:52

Generate a certificate showing this QRCode to a Certifier

Figure 5: List of certificates (left) and QRCode representing a request of a new certificate (right).

V. Validation with Tests and Interviews

This section describes our validation of this work, performed by running a real experiment with the solution presented in Section 4 in a Brazilian favela called Rocinha and by an additional set of interviews. This informal settlement is in an expensive area of the city so, although there are few formal streets, there is a high number of residents who have smart phones.

A. Experiment

Our pilot was executed by deploying the proposed app on Android smartphones, deploying the nodeJS backend component using Heroku, using Google platform to run remote database and statistics and using remote Mongo Atlas Database to store additional backend data.

We agreed with a Priest from the Church called *Nossa Senhora Aparecida* in a big Brazilian favela to go there after a mass and talk with the local community. There were three requirements to participate in this experiment as a user: be at least 16 years old, have an Android mobile phone and go to the church regularly.

We first invited the Priest, who was happy to participate in this initiative. He said he believes that the Church should help to improve the access to public services and to create a volunteer solidarity network. He agreed to be a Certifier Manager and he also said he believes other Priests would be interested to participate as well. The Priest installed the mobile app and could perform the role as planned. He selected three people as Certifiers. The Certifiers agreed with other people to give them proof-of-address certificates. In total, twenty people downloaded the app to their personal mobile phones. Even though the church had Internet connection, they could select online or offline mode as they wished. No one found an error in the app, but they asked a few questions to understand the flow. The experiment finished with all certificates

[4] https://expo.dev/client

[5] https://goqr.me/

[6] https://www.qrcode-monkey.com

generated in their cellphones and most of them synchronized with the remote server, as expected.

The users pointed out some improvements to be made. First, it was not clear to understand the offline mode. They could not easily understand why there are two QRCodes and what they represent. Some of them gave up on the offline mode (that used to be the default one) and changed to the online one. They suggested we configure the online mode as the default option and improve the explanation in the offline mode. Second, two of them were not able to read and suggested to include audio explanations. Third, in some low-capacity mobile phones, the app took a significant time to generate the QRCode and some people did not understand that they should wait and not press any available button before it finishes. These are good insights to be considered as future improvements to this mobile app.

B. Interviews

We requested all 20 users to also answer a small interview. We wanted to better understand if they have a way to prove their identification and their address, if they understood the proposal and how they can use a proof-of-address certificate. To collect insights about how to evolve the app, we also asked who they believe would perform the role of Certifier Manager in the favela and what other types of proofs they believe this app could offer in the future.

All interviewees answered that they have formal identification and they claimed that it is easy to get a government paper-based formal identification in their community. However, only seven of them said they have a paper-based proof of their address based on utilities or formal bank communication. The majority of these seven ones stated that these communications were linked to another address, mainly their jobs or relatives' houses. All users said they could get an address declaration in the community association in their neighborhood provided they go there, pay their fee and bring someone to vet for them.

After they confirmed they understood the proposal, we asked who they believe would be able to perform the role of Certifier Manager. As expected, many of them stated that they believe the Priest can perform this role. In addition, most of them indicated that the community association would be a very good example of Certifier Manager. This indicates that they believe that the existing way of issuing a paper-based address declaration may evolve to a digital proof using an app like what we are proposing. In addition, some of them referred to friends, family and neighbors as possible Certifier Managers.

We also asked how they could use a proof-of-address in their daily lives giving them three non-exclusive options. 13 people selected the option of applying for a new job, 6 people selected the option of applying for studying and 12 people selected the option of opening a bank account or using a financial service. This result is depicted in Figure 6.

When asked what new types of proofs this app can provide in the future, 14 people requested to include a proof of their income, 11 people asked to be marked as someone who is an informal worker, 8 people asked to recognize that they have done some voluntary work and 10 of them asked to acknowledge that they got a loan and have already paid back. One person suggested to include a proof of sex and race. These numbers are expressed in Figure 6. After the formal interview, during informal conversations, a person also suggested to include proof of age, the presence of a disease (e.g., HIV) or disability.

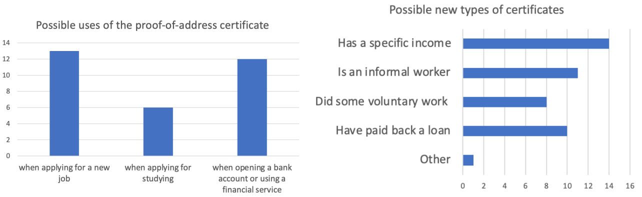


Figure 6: Some interview results

The variety of possible uses and possible new types of certificates on Figure 6 illustrate how the mechanism developed is generic to be used in many types of application.

To wrap up, all the users said they would like to have this app on their cellphones if there is an ecosystem in place to provide and use these proofs.

VI. Conclusions and Next Steps

In this paper, we proposed a solution to prove someone's address using decentralized social trust especially useful when applied to people in poverty without formal documents. The solution can generate accretionary proofs, works offline and aims to promote an open ecosystem involving certifiers and users.

Our solution stores the certifiers in the user's phone without needing a logically centralized data repository. Since the certificates were generated as a result of decentralized social trust, they seem to be an interesting input to different decentralized applications. For example, as an input to social programs, purpose-driven lending or other decentralized finance services.

The same trust mechanism may be applied to many other types of proofs, including proof-of-income, proof-of-volunteer-work, proof-of-good-payer, proof-of-sex, proof-of-race, proof-of-disease or proof-of-disability. In this specific case of proof-of-address, our solution can work in a complementary way with other approaches, like location technologies.

We validated our proposal by building a software solution, running a real experiment and doing interviews with users in a Brazilian favela called Rocinha. Since the community association in Rocinha provides a paper-based declaration of proof-of-address, it is possible to position our solution as a digitalization of an existing process (since we could include the community association as a Certifier Manager) while it also opens a broader range of new proof possibilities.

The software worked as expected in different types of cellphones and the people could understand the concept underlying the application, while they also suggested some improvements and future ways to go about it. As future work, we will develop a plug-in to optionally enable the registration of certificates in a locally centralized repository, for example, a DLT. We plan to use computational trust [6] to create a decentralized way to nominate someone as Certifier Manager without relying on the Admin role used during the bootstrap phase. Then, we will investigate how utility tokens can be used to further ensure that certifiers do not cheat because they would lose more by cheating than by behaving well. Finally, we will explore if the incentive to behave well will increase the overall trustworthiness of the signed certificates.

References

[1] S. M. B. M. Moreno, J.-M. Seigneur, and G. Gotzev, “A Survey of KYC/AML for Cryptocurrencies Transactions,” in *Handbook of Research on Cyber Crime and Information Privacy*, 2020. Accessed: Oct. 03, 2021. [Online]. Available: https://www.igi-global.com/chapter/a-survey-of-kycaml-for-cryptocurrencies-transactions/261722

[2] S. Schmidt, “Só 1% dos imóveis em favelas do Rio tem título de propriedade, entregue pela prefeitura,” *oglobo*, Mar. 14, 2022. [Online]. Available: https://oglobo.globo.com/rio/so-1-dos-imoveis-em-favelas-do-rio-tem-titulo-de-propriedade-entregue-pela-prefeitura-1-25371724

[3] FATF, “The FATF Recommendations.” 2019. Accessed: Dec. 10, 2019. [Online]. Available: https://www.fatf-gafi.org/publications/fatfrecommendations/documents/fatf-recommendations.html

[4] J. Moyano and O. Ross, “KYC Optimization Using Distributed Ledger Technology,” *Business & Information Systems Engineering*, vol. 59, pp. 411–423, 2017, doi: https://doi.org/10.1007/s12599-017-0504-2.

[5] B. Cooper, A. Esser, and M. Allen, “The use cases of central bank digital currency for financial inclusion: A case for mobile money,” Jun. 2019. Accessed: Oct. 03, 2021. [Online]. Available: https://cenfri.org/wp-content/uploads/2019/06/CBDC-and-financial-inclusion_A-case-for-mobile-money.pdf

[6] S. Jean-Marc, “Trust, Security and Privacy in Global Computing,” University of Dublin, 2005.

[7] USAID, “Identity in a Digital Age: Infrastructure for Inclusive Development,” Jul. 2021. Accessed: Mar. 25, 2022. [Online]. Available: https://www.usaid.gov/digital-development/digital-id/report

[8] J. Figueiredo, I. Arbi-Ackel, and H. Beltrão, “LEI Nº 7.115,” Aug. 29, 1983. http://www.planalto.gov.br/ccivil_03/leis/l7115.htm (accessed Oct. 04, 2021).

[9] “UIDAI,” *Unique Identification Authority of India | Government of India*. https://uidai.gov.in/ (accessed Feb. 26, 2020).

[10] Incognia, “Privacy policy | Incognia,” Jan. 11, 2021. https://www.incognia.com/policies/incognia-policy (accessed Oct. 04, 2021).

[11] H. Wen, P. Huang, J. Dyer, A. Archinal, and J. Fagan, “GPS Spoofing Countermeasures,” *Homeland Security Journal*, p. 8, 2003.

[12] N. Strout, “Government leaders worry about GPS spoofing, hacking,” *C4ISRNet*, May 17, 2019. https://www.c4isrnet.com/c2-comms/satellites/2019/05/17/government-leaders-worry-about-gps-spoofing-hacking/ (accessed Mar. 25, 2022).

[13] S. Manu, L. Dave, S. Markus, R. Drummond, S. Orie, and A. Christopher, “Decentralized Identifiers (DIDs) v1.0,” 2021. https://www.w3.org/TR/did-core/ (accessed Oct. 05, 2021).

[14] S. Manu, L. Dave, and C. Davi, “Verifiable Credentials Data Model 1.0,” 2019. https://www.w3.org/TR/vc-data-model/ (accessed Oct. 05, 2021).

[15] M. Allende López, *Self-Sovereign Identity: The Future of Identity: Self-Sovereignity, Digital Wallets, and Blockchain*. Inter-American Development Bank, 2020. doi: 10.18235/0002635.

[16] V. Buterin, “The Meaning of Decentralization,” *Medium*, Feb. 06, 2017. https://medium.com/@VitalikButerin/the-meaning-of-decentralization-a0c92b76a274 (accessed Mar. 26, 2022).

[17] Blockcerts, “Blockchain Credentials,” *Blockcerts*. http://blockcerts.org/ (accessed Oct. 04, 2021).

[18] “KayTrust - Manage digitals identities of your customers.” https://www.kaytrust.id/ (accessed Oct. 04, 2021).

[19] “REM ID.” https://wdi.net/rem/ (accessed Oct. 04, 2021).

[20] OpenZeppelin, “Sending gasless transactions - OpenZeppelin Docs.” https://docs.openzeppelin.com/learn/sending-gasless-transactions (accessed Mar. 26, 2022).

[21] ITU-T, “X.509.” Accessed: Feb. 26, 2020. [Online]. Available: https://www.itu.int/ITU-T/recommendations/rec.aspx?rec=X.509

[22] C. Tartan, C. Wright, M. Pettit, and W. Zhang, “A Scalable Bitcoin-based Public Key Certificate Management System,” Sep. 2022, pp. 548–559. Accessed: Sep. 28, 2022. [Online]. Available: https://www.scitepress.org/PublicationsDetail.aspx?ID=CX5C+Dd1O6k=&t=1

[23] W. Denso, “QRCode.” https://www.qrcode.com/en/codes/ (accessed Oct. 05, 2021).

[24] O. Russ, “Convention Over Configuration,” in *Design Patterns in Ruby*, Addison-Wesley Professional, 2007.

[25] W. Gavin, “Ethereum: a secure decentralised generalised transaction ledger istanbul version.” Oct. 04, 2021. [Online]. Available: https://ethereum.github.io/yellowpaper/paper.pdf

[26] A. Andreas, “Mastering Bitcoin,” in *Mastering Bitcoin 2nd Edition - Programming the Open Blockchain*, O’Reilly, 2017. Accessed: Oct. 05, 2021. [Online]. Available: https://github.com/bitcoinbook/bitcoinbook/blob/4cc04dc39d6503905e85af6409f16beb67bbdb73/ch04.asciidoc

[27] D. Sumedha, W. Sheila, and W. Kevin, “Decentralized Finance: (DeFi) Policy-Maker Toolkit,” 2021. Accessed: Oct. 05, 2021. [Online]. Available: https://www.weforum.org/whitepapers/decentralized-finance-defi-policy-maker-toolkit/

[28] “Cloud Firestore | Firebase Documentation.” https://firebase.google.com/docs/firestore (accessed Oct. 05, 2021).

[29] “EasyQRCode React Native.” Sep. 22, 2021. Accessed: Oct. 05, 2021. [Online]. Available: https://github.com/ushelp/EasyQRCode-React-Native

[30] S. Jean-Marc, L. Carlos, and M. Alfredo, “Secure User-Friendly Wi-Fi Access Point Joining,” Shanghai, 2013. Accessed: Oct. 04, 2021. [Online]. Available: https://archive-ouverte.unige.ch/unige:55387