Human identity and the quest to replace passwords continued

Sirvan Almasi
Department of Computing
Imperial College London
London, UK
sirvan.almasi17@imperial.ac.uk

William J.Knottenbelt
Department of Computing
Imperial College London
London, UK
wjk@imperial.ac.uk

***Abstract*—Identification and authentication are vital components of many online services. For authentication, password has remained the most practical solution despite it being a weak form of security; whilst public-key cryptography is more secure, developing a practical system has been an impediment to its adoption. For identification there is a lack of widely adopted protocol that isn't dependent on trusted third parties or a centralised public-key infrastructure. The Fiat–Shamir identification scheme solved this issue but by assuming there was a single identity issuer to begin with, whilst in reality you would have multiple identity issuers. Here we introduce deeID, a blockchain-based public-key infrastructure that enables multiple Fiat–Shamir identity issuers for identification. At the same time the system is used for authentication that enables better than password security whilst remaining practical. We achieve a decentralised network of identities that is used for identification and authentication, thus providing a common protocol for organisations to use when communicating with respect to a user (identity holder).**

***Index Terms*—Identification, Authentication, Blockchain, PKI**

I. Introduction

Authentication is when a user, A, can prove to the verifier that they are user A, but another user cannot prove to the verifier that they are user A. Identification is when a user, A, can prove to the verifier that they are A, but the verifier or anybody else cannot prove that they are A. Both of these concepts form an integral part of our interaction with online services and underpin the security of that interaction, too. Password is the dominant authentication scheme despite its flaws; it is especially prone to user errors [1], such as: re-using them, forgetting them and simply using weak passwords. Bonneau et al. in [2] demonstrated, in a comprehensive and systematic analysis, why password has remained the dominant end-user authentication to this date. In this paper we add to their work by developing a blockchain-based identification and authentication system named deeID. Hence, the title of this paper *'... the quest to replace passwords continued'*. The overall aim of this paper is to marry the notions of identity, authentication and blockchain. Blockchain is proving to have many more important applications beyond its primary conceived application of digital cash (Bitcoin [3] since 2009). Using it as a public-key infrastructure (PKI) and identity management is two important examples that have tremendous implications for web applications. The likes of Namecoin [4], Blockstack [5] and Certcoin [6] are examples of organisations attempting to recreate DNS, identity and certification services using blockchain. Other examples that have used the blockchain as a PKI include [7] for IoT devices and [8] for identity management.

The role and security of our identity in the digital ecosystem has taken a centre stage as the significance of online services grows and more of us, increasingly, use digital products and services. Whilst the username and password combination has remained an established authentication method due its accessibility and deployability as highlighted in [2], it remains a weak and insecure method [9] and it is continuously propped up by forcing users to adopt certain behaviours, e.g. length of password, character combinations and added on factors such as mobile phone verification. This, to a certain extent, has made passwords less human friendly and in certain cases has made it more insecure as it encourages users to use the same password across different sites [10].

Identification or entity authentication is important because it is a routine task for most of us. We swipe in through gates and doors to access facilities and use passwords to login to our emails and social media accounts. Identities are proven via three methods: (1) What we possess, (2) What we know and (3) What we are, e.g. bio-metrics. In majority of online authentication schemes, we use method (2), which is what we know, that is we know a specific password twinned with an email account. The application, that we are authenticating with, ensures we own the email account by testing if we can access the contents of that email account. Crude authentication schemes, usually via phone calls, use simple information like knowledge of our own date of birth or our home address; which then grant access to resources and account details. Hence why phone scams and identity related scams are prevalent [11]. We naturally progress into what we possess and what we are in authentication. From an industry point of view we are most concerned with a system that is decentralised by default. Because we want to eliminate the ongoing use of trusted third parties. Therefore, one of our objective is to *propose a decentralised entity authentication system using identification schemes for the real-world that is more secure than passwords, is easy to deploy and use.*

Proofs of identity are usually layered where each layer is a proof that is typically derived from another proof. We do not own a single identity proof but many of them, such as: passport, birth certificates and driving licence. Certain organisations usually require multiple proofs of identity; one reason for such a requirement is the idea that the cost of faking all these various documents provides a barrier for majority of individuals to attempt a forgery. Therefore, we have two things to take away: (1) The more work done and more interactions we have the more trust and reputation we gain. The cost of forgery provides a barrier to creating fake identities, and (2) Proofs of identity are layered. The more proof we have (in numbers) then the more trust can be put upon our claim to the identity. Multiple identity providers is challenging with the Fiat–Shamir scheme, but as discussed is necessary for the real world. A reason why identity theft is an issue is because the requirements and challenges to the claim are shallow. Currently, most online services require the knowledge of a user's name, address and date of birth to open a credit card or take out an insurance policy. Phone based authentication are also shallow in that, again, only basic personal details are required to access the account. Thus, this brings us to another objective: *provide a dynamic method for organisations to issue individuals with an identity*

We should also note the increasing problem of synthetic identity fraud [12]. This is where criminals create fictional identities and legitimate financial and credit history for the identity. This adds credibility and makes it appear to be real to a challenger and most organisations. Though, it is only a stepping stone for the criminals to have access to bigger credit and finances. This is partly encouraged by the lack of a common and trusted service that all organisations could use for identity verification.

The benefits of the proposed system are numerous. Firstly, the most obvious application is the replacement for the username and password combination for authentication. With a secure, up to date and decentralised key-store, it is a practical alternative (especially considering potential cost savings with respect to lost passwords highlighted in [13]). Secondly, an easy method to prove ones identity online by using identity providers (e.g. your bank can be a trusted identity provider). Thirdly, it can help to reduce phone scams, identity fraud and other related crimes by ensuring that access is less dependent on common personal information but based on a combination of 'what we possess' (mobile phone with strong private keys), or 'what we know' combined with 'what we have' (fingerprint access to our mobile phones). Lastly, it brings flexibility and a regular standards to organisations, countries and continents where there is a gap in standards and trust.

In the Background section we provide a background of important concepts (identity, Fiat–Shamir scheme, blockchain-based identity schemes). In the Architecture section we give the requirements for our conceived system followed by an overview of its implementation. Next, we evaluate the deeID system using the comparison framework provided by Bonneau et al in [2]. Before concluding the paper we will discuss our implementation's contributions, applications and limitations in the Discussion section.

II. Background

In this section we will explore the definition of identity and its representation, existing authentication schemes, and identification cryptosystems.

A. Identity

The Oxford dictionary defines the word *identity* as *The fact of being who or what a person or thing is.* By identity we mean a truly unique representation of an entity (whether human or not). One way of uniquely representing a human identity is through concatenation of the person's characteristics, such as: hair colour, genetics, height, and so on. If we gather enough characteristics we can uniquely represent the person with a high probability. This representation is merely a set of characteristics which can form into a very long string. Such representation is prone to errors as identification is a process of many steps. Recording such characteristics can be difficult as one may argue over the colour of hair or eyes. Then we must think about verifying the identity and the practicality of doing so, too. Therefore, with respect to an identification protocol, we must consider the following:

1) Capture: Unique 'bits' about the entity and the process of doing so.
2) Recording: Can it be recorded on a machine?, how is it recorded? and the efficiency of doing so.
3) Security: Is it tamper-proof?, is there a risk to impersonation and other relevant threats?
4) Verification: Can we verify the identity?, is the process error free? and is the process efficient enough?

Humans identify one another through human readable strings and visual clues (first names, surnames, etc.). Though it is efficient enough at small-group levels (family level for example); this is not unique and alone cannot be used for authentication. Then we have biometrics, such as fingerprints and other physical features that uniquely identifies us. We also have a unique set of characters and numbers, such as passport number and serial numbers, which represent a unique record - usually accompanied by a photo and your full name, therefore, a mix of highly probable unique and human readable formats.

B. Authentication Methods

The username/email and password combination has remained the go-to standard for authentication. Other iterations such as password managers, two factor authentication (2FA), and federated systems have made advances in improving its security and usability. Bonneau et al. in [2] provide a unique framework that allows us to compare different authentication schemes. In their paper they provide a unique insight on why password has remained to be such a practical system despite its flaws. The comparative framework in [2] compares different web authentication schemes based on 25 factors which are categorised under three headings: usability, deployability and security. As expected most other schemes perform better in security, mix results on usability, however all the other schemes do worse than passwords on deployability. Overview of their comparative result is visualised in Figure 4 along with the comparative result of deeID.

Federated protocols such as OpenID [14], OAuth [15] and SAML [16] provide a greater level of standardisation, their adoption by tech-giants like Google [17] and Facebook [18] has increased their adoption amongst developers as more users find it more convenient to have have one login credential. Nevertheless, they are still using password at their foundation. Therefore, as noted above, password is a weak form of authentication and security remains to be a concern. Hardware tokens and phone-based schemes that use special cryptographic keys are significantly more secure than passwords. Though simple categorisation does not mean an automatic better security; implementation remains to be important too. As indicated in the results of [2] (also shown in Figure 4), MP-Auth [19] and IronKey [20] are shown to be less secure than their counterparts in the same category.

Biometric-based authentication schemes are an easy to use schemes. However, the scheme is held back due to technology and the general noise when capturing the data (not deterministic). Biometric schemes generally require a piece of hardware that is only useful for that application and thus makes it unpractical for us to carry around all the time. Though, mobile phones have helped by providing face and finger print scanners in recent years. We should also see an increased use of it as mobile phone adoption increases. Given that it is much more difficult to steal biometric data [21] and its ease of use, we should see more of it being used in conjunction with other schemes.

C. Blockchain Based Identity Systems

The cryptocurrency invention initiated a race to create decentralised applications and amongst them were and still are identity management systems, each utilising either Ethereum [22] or Bitcoin as their blockchain platform. Organisations are coming together under foundations such as the Decentralized Identity Foundation [23] to create a standardised ecosystem. Standards are also being developed at the same time, evident by Decentralized Identifiers (DID) [24].

Numerous papers such as [25] have implemented variations of identity solutions on the blockchain. Zhu and Badr [26] in their survey provide further information on blockchain-based identity systems. Therefore, we will not do a survey as such here. Many of the notable implementations are commercial or experimental projects, such as uPort [27] and Sovrin [28]. Taking these two as an example, they both try to do the same function but using very different technology and philosophy to tackle decentralisation, security and trust. uPort uses the Ethereum blockchain as its foundation, and therefore its consensus, security and growth is bounded by the Ethereum chain. Sovrin has taken a different approach, it uses the Hypderledger [29] stack and it is a permissioned ledger. With respect to consensus, the integrity of the chain is maintained by the so called stewards. Therefore, one has to raise the question of whether if it really is self-sovereign at all? The transparency of a cheated system is irrelevant if the honest participants cannot do anything about it.

The closest related work, to our knowledge, is the work of Boontaetae et al. [8]. They have similarly utilised the blockchain (Ethereum in their implementation) as a PKI system and identities are issued by trusted sources known as Trusted Source Certificate Authorities (TSCA). the TSCAs are responsible for verifying the end-user's identity via some offline means and then signing their cryptographic keys using their own. Consequently, storing the relevant keys in a universal smart-contract (named RDI). The cryptographic keys are known to be the hot and cold keys, where the hot-key represents an identity and the cold-key is responsible for revoking the hot-key. several issues begin to emerge at this point - signing keys en mass proves to be a fatal security problem. If a key gets compromised then the entirety of the signed-keys must be re-signed by a new key. Another issue is the representation of identity, it is fair to say that an identity is represented by a key. But *who* is that key? and what human readable attributes can we attach to that key?. This hasn't been discussed enough in the paper - legitimacy of an identity can be important in its own. However, most applications will require cross-referencing and at times some human readable attributes too.

D. Identity-based Cryptosystems

The drawbacks of using a public-key cryptosystem are: its practically, certification and trusted party requirements. This is where the idea of identity-based cryptosystem comes in. It essentially allows one to use one's identity (e.g. name, date of birth, place of birth etc.) as a public-key. In such a system everyone would have access to some function f. This function would allow one to compute the public-key from some string which is unique to the individual.

The famed cryptographer Adi Shamir[1] first introduced an identity-based cryptosystem and signature scheme in his 1984 paper *"Identity-based cryptosystems and signature schemes"* [30]. This novel technique is still based on public-key cryptosystems, however, with a *twist* as he describes it himself. Instead of generating and publishing a public-key, the user can use a set of identities and features unique to themselves (e.g. name, date of birth, place of birth etc.) as a proxy for the public-key. The implementation scheme provided by Shamir is only an identity-based signature scheme in this paper.

Shamir describes the scheme ideal for closed groups of users, however, our ecosystem requires an open system, but he also goes on to say it is practical on the national scale.

Key properties as described in his paper [30]:

- A trusted key generation centre is required to generate user's secret key and issue in the form of smart card.
- Advantage is the ease of use, *"...it can be used effectively even by laymen who know nothing about keys or protocols."*

Security of the system depends on the following:

- Security of the underlying cryptographic functions
- Secrecy of the privileged information at the key generation centres.
- Thoroughness of identity checks at the key generation centres before a smart card is issued.
- Precautions taken by the user to prevent loss, duplication or unauthorised use of their card.

Shamir only provides a signature implementation scheme for his identity-based system idea. Shamir does return to this idea in subsequent papers in the following years after 1984. In 1986, Fiat and Shamir give us the first concrete solution [31]. They note that *"The new identification scheme is a combination of zero-knowledge interactive proofs [32] and identity-based schemes [30]"*. In this paper [31] they provide both the signature and proof of identity scheme that Shamir proposed in the 1984 paper [30]. Moreover, Shamir along with Feige and Fiat produce a further study on *"Zero-Knowledge Proofs of identity"* [33], this paper discusses and differentiates between zero-knowledge *"proofs of membership"* and *"proofs of knowledge"*.

Feige-Fiat–Shamir Identification Protocol [33] This is the algorithm as described in their paper in 1988 *"Zero Knowledge Proofs of identity"* [33]. We have made some adjustments to the notation to be consistent throughout this paper. Its security is dependent on the intractability of computing the square roots mod n.

Assuming the interaction is occurring between two entities, the *prover A* and the *verifier B*.

A's key generation protocol is as follows:

1) Choose k random numbers $S_1, ..., S_k$ in $\mathbb{Z}_n$
2) Choose each v_j (randomly and independently) as $\pm 1 \cdot (S_j^2)^{-1} \mod n$
3) Publish $v_1, ..., v_k$ and keep $S_1, ..., S_k$ secret

The generation and verification process (i.e. interactions) takes place as follows via a four step process: Repeat t times:

1) A (the prover), picks a random R, and sends $X = \pm R^2 \mod n$
2) B (the verifier), sends a random boolean vector $E_1, ..., E_k$
3) A sends the value $Y = R \cdot \prod_{E_j=1} S_j \mod n$
4) B verifies that $X = \pm Y^2 \cdot \prod_{E_j=1} v_j \mod n$

In their last remarks [33], Feige, Fiat and Shamir note: *"An interesting modification can eliminate the public key directory and lead to a key-less identification scheme"*. We use the identity string to public-key transformation provided in [31] to create public-keys in the system. Implementation of the scheme is given in the Architecture section.

The implementation provided by Fiat and Shamir [31] is for proving ones identity through an interactive manner. However, there had not been any *encryption* schemes until the works of Boneh–Franklin [34] and Cliffor Cock's scheme [35] in 2001.

The systems described above provide an important layer on top of public-key cryptography, public-keys are not the most user-friendly concept, for a national system we require a more friendly scheme that even the laymen can understand and operate. Moreover, the major advantage of the Fiat–Shamir protocol is that it can work with systems that are limited in power and memory.

There are other protocols similar to the Fiat–Shamir protocol, such as the Guillou-Quisquater (GQ) identification scheme [36] (difference is a reduction in the number of messages exchanged and the memory requirements for user secrets). The Schnorr identification protocol [37] depends on the intractability of the discrete logarithm problem unlike the Fiat–Shamir and GQ protocol.

1) Attacks on Identification Protocols: There are various security threats to identification protocols:

- **Impersonation:** Pretending to be a different person
- **Replay attack:** Use of information from a previously used protocol process, e.g. re-use of a poor signature.
- **Interleaving attack:** Multiple concurrent execution of the protocol and using selective information for possible attacks.
- **Reflection Attack:** Attack on a challenge-response method such as the interactive Feige-Fiat–Shamir

[1] https://www.britannica.com/biography/Adi-Shamir

protocol, essentially tricking the challenger into providing itself with the answer.

- **Forced Delay:** Requires intercepting a message and delaying it until some later time (thus not a replay attack).
- **Chosen-text Attack:** Strategically choosing challenges in an attempt to extract information about the provers secret information.

The above was mainly inspired from [38] p.417, and other sources.

III. Architecture

In this section we will lay out the requirements for our conceived system and provide an overview of its implementation.

A. System Requirement Overview

Below are the key requirements of our conceived system.

Beyond passwords.: Our problems requires a system that goes beyond passwords and hence does not require the user to remember complex strings and sequence of numbers for authentication. The consequence of this is that the user will have to use something other than 'what they know', hence, 'what they possess' or 'what they are'. Therefore, having studied identification schemes and use of public/private key pairs - storing such secure keys on a device such as a mobile device will be a requirement.

Compatibility: Compatible with existing browsers and existing technology that use the email/password combination for authentication. The user must not download new browsers. Developers should be able to use existing JavaScript or Python web stacks to implement deeID as an authentication scheme.

Key store management.: Shared and equal access to public-keys that link users to identities. This is assuming that the user can have multiple keys (e.g. for multiple devices and keys for encryption).

Dynamic identity: Ability for anyone to issue identities to individuals. Utilise and implement the Fiat–Shamir scheme for multiple identity issuers.

Privacy by design: Cannot record any personal details on the public chain.

Unlinkable authentication: Colluding actors cannot determine if our anonymous identity is the same across their servers and systems.

B. Implementation of deeID

The proposed system, deeID, is a combination of technologies that enables a dynamic and decentralised identity system with functions compatible with existing services. Although the proposed system can be built on various devices, even a unique independent device solely for this application, we have assumed the implementation on a typical mobile device running Android or iOS. deeID uses the Fiat–Shamir identification cryptosystem with blockchain acting as a PKI.

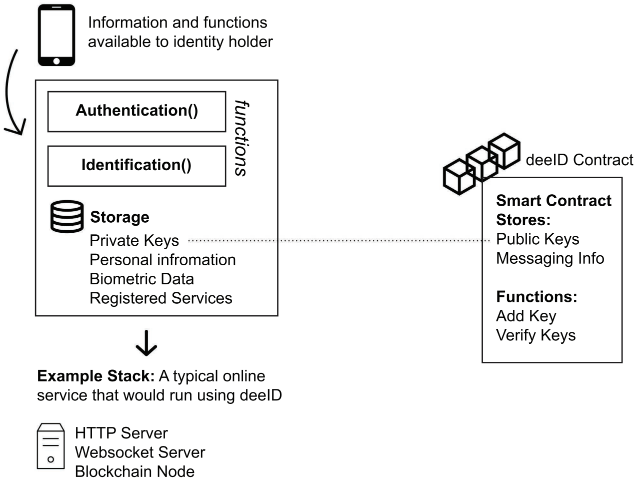


Figure 1. Top view of the deeID architecture

Figure 1 provides an overview of the system. The essential components are:

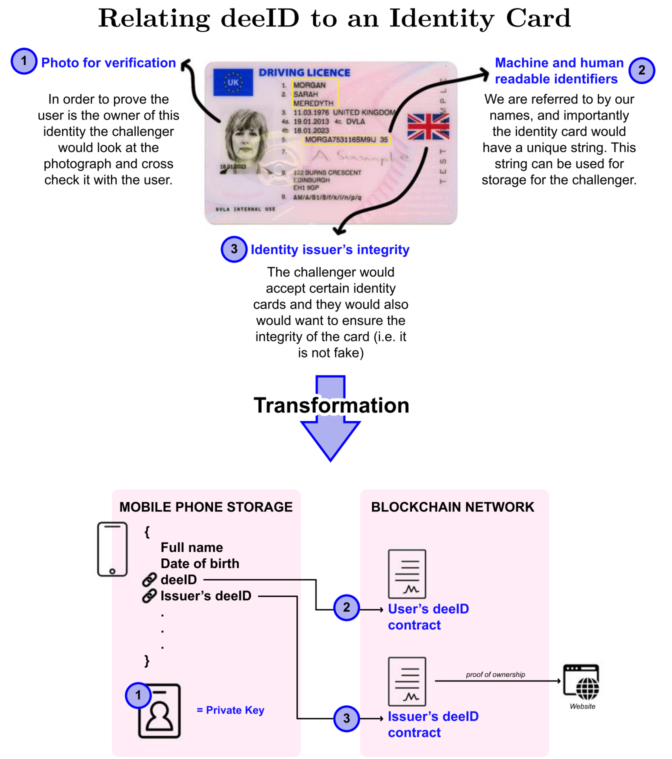
- **Distributed Ledger:** Our implementation uses Ethereum and we have developed several smart-contracts that act as portals and identity managers.
- **Mobile phone:** A device such as a mobile phone that can store private-keys and interact with services and servers over the internet (partly to ensure compatibility with existing web applications). We built a mobile phone application that allows the user to authenticate and identify (Fiat–Shamir scheme) themselves. The application uses QR codes to interact with web applications and then talk to servers (using WebSocket connection) to carry out the necessary functions (e.g. interactive Fiat–Shamir scheme).
- **Libraries:** Required libraries that developers can install so that their users can use deeID to authenticate themselves on the developer's application.

A better way to demonstrate the capability of deeID would be to use an identity card as an analogy. Figure 2 visualises this comparison; there are three main components to an identity card:

- **Photograph for verification:** We typically use the photograph of the individual on the identity card to identify the user. Meaning that the assumed valid ID card belongs to the individual that is trying to prove their identity to the challenger. In deeID this is done through the ownership of a private-key associated with an identity. Via the Fiat–Shamir scheme one would be able to prove their ownership of the private-key and therefore identity.
- **Human and machine readable strings:** We require a unique string that would uniquely identify the card. This string can also be used for record keeping with respect to the identity holder. Along with this unique string we have the full name of the user as well.

Though it is typically not unique. However, it is what we use to identify with one another at a human level. In deeID, the smart-contract on the network with its unique address represents the unique machine-readable identity.

- **Identity issuer's integrity:** The integrity of the identity card comes through trusting the physical card itself and its physical properties. This trust also assumes that the verifier trusts the organisation that has issued the card. We find organisations that require different proofs of identity, e.g. passport and driving license in the UK. Therefore, we are concerned with different issuers and their reputation. Reputation in itself would require a full research paper, in the current state of deeID we have used a simple link on the identity issuer's smart-contract, which links to their web address if they have one proving that the owner has access to both.



In order to prove the user is the owner of this identity the challenger would look at the photograph and cross check it with the user.

We are referred to by our names, and importantly the identity card would have a unique string. This string can be used for storage for the challenger.

The challenger would accept certain identity cards and they would also would want to ensure the integrity of the card (i.e. it is not fake)

Figure 2. At the top we have the main components of an identity card and at the bottom we have the equivalent functions in deeID.

1) Public-key Management: Management of public-keys is an important aspect of deeID which defines much of its capabilities. In our problem statement we stated that one contribution of this paper is the extension of the Fiat–Shamir scheme to multiple identity providers. The way to do so without taking anything away from the scheme was to use the idea of decentralised PKI. Numerous studies have contributed to decentralised PKI (DPKI) [39] [40] [41] [42] and here we represent a simple solution built on Ethereum for deeID.

The use of DPKI goes beyond the Fiat–Shamir scheme in deeID. The user can use it to associate other public-keys to their identity, e.g. RSA keys for encryption and public-keys used on other devices that links back to the identity. Generally, DPKI has been proposed to be used for alternative DNS solutions [43], IoT [44] and much more. The challenges to the adoption of blockchain as a PKI is similar to that of any other blockchain application: consensus schemes, practicality, scaling and standardisation.

We assume that the user has ownership control of their deeID smart-contract and thus any changes within the contract are not malicious. Furthermore, the integrity of the the public-keys fall under the actual deeID and individual public-keys are not issued certificates as such. So, if a verifier has no knowledge of the user's deeID then they must first identify the deeID user (through the Fiat–Shamir scheme).

Adding new public-keys to the deeID smart-contract is a simple process of invoking the `addKey()` function with the following arguments: `title`, `key` and `comment`. The criteria for adding the key is that the user must be the owner of the deeID - which in our implementation has been cross-checking the sender's address with the address of the owner (stored on the smart-contract). A real world-application could use any of the keys stored on the deeID to verify the integrity of the sender. Lastly, verifying a specific key would involve querying the blockchain with respect to the public-key and the deeID, in our implementation we had a function, `getKey()`, which would return all the details related to that key.

Other blockchain-based PKIs are more traditional in that they use certifications, e.g. CertLedger [39] and CertCoin [45]. On deeID the authenticity and security comes from the identification of the deeID itself which can depend on the identity issuer and the verifier's willingness to trust the issuer or not. Whilst in the current implementation of deeID this verifier and issuer trust is not quantified, we envision that one would be able to add a quantifiable reputation layer.

2) Authentication: With respect to current web applications, the role of password is mainly for authentication. deeID provides an alternative authentication option to passwords that are compatible with current technologies with minimum development overhead. The components of the authentication system are: a mobile phone running deeID application and the web application running a blockchain node with the necessary deeID libraries for it to interact with the mobile phone.

One can authenticate themselves via four different methods:

- **Explicit link to deeID:** With this method we are using the deeID, i.e. identity smart-contract, to authenticate the user. Essentially using the PKI method. Once the user has a public-key stored on their deeID contract then they can use that to authenticate themselves using their deeID. The user and the verifier have to query the chain for the existence of the specific public-key used to sign the authentication message. Lastly, because only the deeID is used to authenticate, the user can store multiple public-keys on the contract and thus use multiple-devices with different keys for authentication. The Figure in Appendix E provides a visual sequence of main events that are required in this authentication process. At the end the typical session creation occurs, we have simply shown "Login successful" to re-iterate the unique points related to our protocol.

- **Generate a public-key:** One can also avoid the use of the blockchain all-together and simply generate a unique public-private key once they register with a website and use this public-key for authentication. The benefit of this is that the user can remain anonymous to the application. This also provides *unlinkable authentication*. This means that two or more colluding servers are unable to determine if the user is using their platform. Therefore, providing some privacy if required. Though the colluding servers could use other means such as using IP address to guess if they have the same users.
- **Fiat–Shamir identification:** The user can simply use the Fiat–Shamir scheme to authenticate themselves too. With this method you're also sharing your identity and its proof; this is therefore not useful if the user wishes to remain anonymous.
- **Strong password:** Just like with the second method, one can use their mobile phone like a password manager and generate strong passwords to authenticate themselves with a given service.

Auth Method	Chain Storage	Query Chain	Interactive	Multi Device
Explicit link to deeID	Yes	Yes	No	Yes
Generate a public-key	No	No	No	No
Fiat–Shamir iden'	Yes	Yes	Yes	No
Strong password	No	No	No	No

Table I
Methods of authentication with deeID.

Table I summarises the above authentication methods. Although the choice of which authentication method to use is not entirely dependent on the user, we believe, ultimately, online behaviour will dictate which method is best suited to a given application. The differences between the authentication methods are primarily the use of blockchain and ease of use (e.g. multi-device and requirement of a mobile phone).

3) Issuing an identity: We have described the system as dynamic and self-serving. This means that anyone and anything can issue an identity. This is done through the Fiat–Shamir scheme as described in the Background section. Essentially, each issuer would have two large primes numbers that are kept secret, the product of these two are kept on the issuer's blockchain identity, i.e. smart-contract, so, when an identity holder claims to have been issued an identity, the verifier can check the existence of the product of the two large prime numbers within the issuers smart-contract or deeID$_{contract}$.

Below we go through an example of how an identity is issued by any other participant on the network. The first box states the requirements and the necessary steps required before two participants can create identities for each other. In our implementation we created a flask (Python based) app that allowed a user to create an identity. Therefore, most organisations and current systems are able to do so without much development overhead.

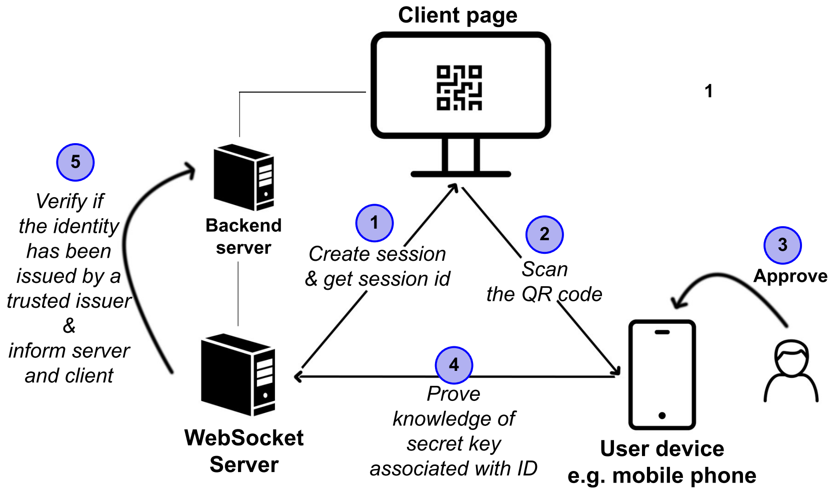
In our implementation we created a Python function (described as f throughout the paper) that any developer can use to transform an identity string to a Fiat–Shamir public-key given the string, indices and n. A simple numerical example is given in Appendix B.

Requirements

Steps required before an entity can issue an identity:
Person **A**: The individual wishing to get a new identity from another person or organisation. Person **B**: Identity issuer.

1) **A**: Must create a deeID representation on the blockchain, e.g. on Ethereum, create a unique contract as per the common standards.
2) **B**: Must also create a deeID representation on the network.
3) **B**: Create a random modulus which is a product of two large prime numbers, $n = pq$, and n should be at least 512bits.
4) **B**: Store n on the network. In our implementation it is stored in a data structure on the deeID contract.

Verifying an issued identity



Client page
1
5
Verify if the identity has been issued by a trusted issuer & inform server and client
Backend server
1
Create session & get session id
2
Scan the QR code
3
Approve
4
Prove knowledge of secret key associated with ID
WebSocket Server
User device e.g. mobile phone

Figure 3. Visualisation of how a traditional web page can authenticate a user's identity.

Example of issuing an identity

1) **A:** Hand over the required identity bits, $I_1 = \{\text{Fullname, DoB, deeID, City of Birth}\}$
2) **B:** Add a random string to I_1, and let that be I hereafter.
3) **B:** Verify the identity of A (e.g. face to face by looking at passport etc.)
4) **B:** If successful, create the credentials as follows:
   a) Create the public-keys: $v_j = f(I, j)$ where v_j is the public-key, f is some pseudo-random function *(expansion of this is given in Appendix A and construction of such a function is explained in [46])* and j are small random values; to create the public keys we pick k distinct j values for which v_j is a quadratic residue mod n, i.e. solution exists for $x^2 = v_j \mod (n)$
   b) Calculate the secret keys for A by taking the smallest $s_j = \sqrt{(v_j^{-1})} \mod (n)$.
   c) So the set of k public keys and associated j values would give us two sets of values: $V = \{v_{j(1)}, v_{j(2)}, ..., v_{j(k)}\}$ which is our public keys, $J = \{j_1, j_2, ..., j_k\}$
5) Now B can hand over $\{I, J, S, V\}$ to A. V is technically not required because they can re-calculate it themselves if they have I and J.

When an identity is issued, the state of the blockchain would not change; therefore there is no blockchain related costs associated with issuing new identities. The issuer would have the product of p and q, n, within their smart-contract already, if not then the issuer would have to create a transaction and place n within their `keys` array on the smart-contract, in this scenario there is an associated cost with this transaction.

4) Verifying a true identity: By verifying a true identity we mean that we are concerned with the actual real life identity of the person. You can think of this as checking ones passport to get the real identity of the person we are concerned with. So, we have covered how you can issue an identity to someone, now we will go over how you can verify that identity. To give context, this would typically happen when one is applying for a credit card, insurance policy or car finance online. Figure 3 provides an overview of how we will go about this. We assume that the verifier (the backend and WebSocket servers) will have a set of trusted identity issuers deeID addresses. Therefore, when they ask for an identity they will ensure that the provided identity is provided by one of the trusted issuers. The actual authentication happens with the web server and the user's device (mobile phone), thus, for this to happen the user has to first interact with the client, capture the required information and then carry out the identification protocol with the WebSocket server. Upon the completion of the verification the server can update the client and carry out with the intended functions (e.g. registration and creating a session for the user).

5) Phishing protection: Phishing is a common threat and it manifests itself through mediums such as telephone, email and SMS. The problem is an identification one. Phishing attacks are successful due *operator mistakes*. This means that we mistake what is before us to be the same as what we thought it was (e.g. similar looking URLs). If we assume that an honest sender of a message cannot have malicious messages then our problem reduces to simply identifying the sender. In our protocol this process is simple; let us assume that your bank sends a message to you with a link. In order to avoid phishing attacks the system would automatically check the signature of the message and cross that with the services that you trust (can also cross it with the blockchain). To simplify the process we have locally stored set of deeID addresses so that we do not have to keep verifying deeID addresses with the blockchain.

IV. Evaluation

In order to evaluate our protocol we have used the framework proposed in [2], this allows us to compare the protocol with different types of authentication schemes in detail.

A. Common security threats

Here we will evaluate the identification and authentication schemes through comparison and analysis of their behaviour with respect to security threats.

1) Fiat–Shamir Scheme: The scheme's security is well described in the protocol's paper [31]. The security of the cryptosystem is based on the fact that it is computationally difficult to factor large prime numbers.

2) Phishing: This is a common threat that involves social engineering techniques to hijack identity and sensitive information such as ones password. The threat channels are commonly: SMS or any instant messaging platform, emails, miss-typed or look-alike URLs. The consequences of phishing attacks are: account hijacking, identity theft, financial loss and much more. There are countermeasure techniques [47], however the lack of adoption of a single standard hinders the growth of these techniques.

With respect to passwords, receiving notifications to change your password is a common technique used to steal the password. Upon registration the deeID mobile application creates a link in its local storage. This store is essentially a dictionary of services that the user has a connection with. Upon communicating with the registered service that user's device would carry out a check to confirm whether they have an existing link to this service or not. We assume in this scenario that the user's device is secure. The system avoids relying on the user for the entire process of authentication (recognising the website, inserting information and so on), this is similar to the *PhoolProof* system [48].

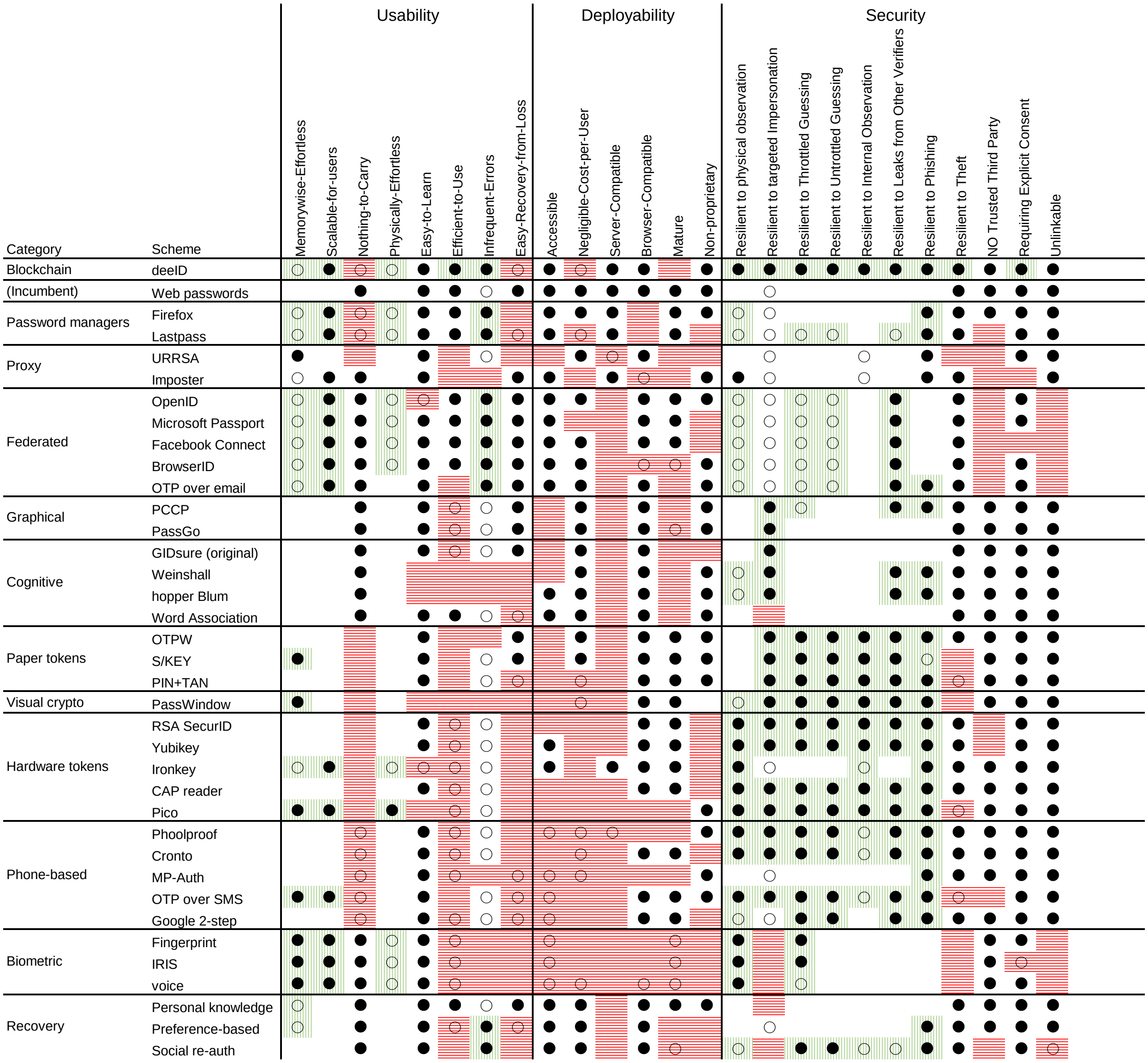
B. Semi-structured Evaluation of User Authentication Schemes Framework

In *'The Quest to Replace Passwords'* [2], the authors suggest a framework that compares various authentication schemes, with a strong focus on web authentication. In this section we will put deeID to the test.

Starting with *Usability benefits*, we believe deeID provides *Meomorywise-Effortless* benefits, there is no need to remember a password or a secret. However, for more security we encourage users to have a pin-protected device. The application is highly scalable for the users, they can store as many credentials as they want without any extra burden, therefore better than passwords. However, the users are required to carry a device with themselves, such as a mobile phone device, since this is an object that they carry around with themselves anyway, we rate it as *Quasi-Nothing-to-Carry*. The scheme is also *Physically-Effortless*, they are only required to aim their device for example and then press a button to grant or decline a request. The scheme is also *Easy-to-learn* and intuitive, it requires scanning an image and pressing a button. Also it is far more efficient than a password, *Efficient-to-Use*, the user is only required to scan and press a button, no need to type or remember a password for example. *Infrequent-Errors*, mainly compared against biometric schemes, the process is very reliable and there are no false-positives. The scheme, however, is not so easy with respect to recovery of lost private keys, depending on which keys are lost, usually there are no recovery methods but to replace the lost credentials with the identity provider.

Moving onto *Deployability benefits*: The scheme is highly *Accessible*, and provides *Negligible-Cost-per-User*, beyond the access of a mobile-phone device the user is not required to spend anymore money, this is same for the verifier. The scheme is also *Server-Compatible*, meaning with their existing technology they are able to run the scheme. At the prover's end, they do not require to change or amend their browsers, therefore it is *Browser-Compatible*. It is not proprietary technology, however, it isn't mature at this stage.

On *Security-benefits*: It is resilient to physical observations, as the secret is not observable and all authentications do not require the user to reveal the secret keys. It is also resilient to targeted impersonation, knowledge of ones personal details will not mean that they can break the authentication scheme. Given that the scheme is challenge-response based, and it is machine-to-machine (mobile phone to server), the rate of failure due to any other reason than wrong key is low, therefore the chance of guessing and getting it right is very low. Unthrottled-guessing depends on the identity provider and the length of their keys. Resilient-to-Internal-Observation is very low, since the user does not interact with the private keys directly, therefore key logging and other methods have no chance of revealing the secret-key. Given the zero-knowledge concept of the Fiat–Shamir concept, anything that the verifier knows cannot help reveal any information about the user's secret key.The scheme is also resilient to phishing attacks, again due to ZK, no information about the secret-key is revealed. The scheme is less resilient to theft, however with the use of a PIN on the device, the scheme is qusi-resilient-to-theft. *No-Trusted-Third-Party*:



Category	Scheme	Usability								Deployability						Security										
		Memorywise-Effortless	Scalable-for-users	Nothing-to-Carry	Physically-Effortless	Easy-to-Learn	Efficient-to-Use	Infrequent-Errors	Easy-Recovery-from-Loss	Accessible	Negligible-Cost-per-User	Server-Compatible	Browser-Compatible	Mature	Non-proprietary	Resilient to physical observation	Resilient to targeted Impersonation	Resilient to Throttled Guessing	Resilient to Untrottled Guessing	Resilient to Internal Observation	Resilient to Leaks from Other Verifiers	Resilient to Phishing	Resilient to Theft	NO Trusted Third Party	Requiring Explicit Consent	Unlinkable
Blockchain	deeID	○🟩	●🟩	○🟥	○🟩	●	●🟩	●🟩	○🟥	●	○🟥	●	●	🟥	●	●🟩	●🟩	●🟩	●🟩	●🟩	●🟩	●🟩	●🟩	●	●🟩	●
(Incumbent)	Web passwords			●		●	●	○	●	●	●	●	●	●	●		○						●	●	●	●
Password managers	Firefox	○🟩	●🟩	○🟥	○🟩	●	●	●🟩	🟥	●	●	●	🟥	●	●	○🟩	○					●🟩	●	●	●	●
	Lastpass	○🟩	●🟩	○🟥	○🟩	●	●	●🟩	○🟥	●	○🟥	●	🟥	●	🟥	○🟩	○	○🟩	○🟩		○🟩	●🟩	●	🟥	●	●
Proxy	URRSA	●		🟥		●	🟥	○	🟥	🟥	●	○🟥	●	🟥	🟥		○			○		●	🟥	🟥	●	●
	Imposter	○	●	●		●	🟥	🟥	●	●	🟥	●	○🟥	🟥	●	●	○			○		●	●	🟥	🟥	●
Federated	OpenID	○🟩	●🟩	●	○🟩	○🟥	●	●🟩	●	●	●	🟥	●	●	●	○🟩	○	○🟩	○🟩		●🟩		●	🟥	●	🟥
	Microsoft Passport	○🟩	●🟩	●	○🟩	●	●	●🟩	●	●	🟥	🟥	●	●	🟥	○🟩	○	○🟩	○🟩		●🟩		●	🟥	●	🟥
	Facebook Connect	○🟩	●🟩	●	○🟩	●	●	●🟩	●	●	●	🟥	●	●	🟥	○🟩	○	○🟩	○🟩		●🟩		●	🟥	🟥	🟥
	BrowserID	○🟩	●🟩	●	○🟩	●	●	●🟩	●	●	●	🟥	○🟥	○🟥	●	○🟩	○	○🟩	○🟩		●🟩		●	🟥	●	🟥
	OTP over email	○🟩	●🟩	●		●	🟥	●🟩	●	●	●	🟥	●	🟥	●	○🟩	○	○🟩	○🟩		●🟩	●🟩	●	🟥	●	🟥
Graphical	PCCP			●		●	○🟥	○	●	🟥	●	🟥	●	🟥	●		●🟩	○🟩			●🟩	●🟩	●	●	●	●
	PassGo			●		●	○🟥	○	●	🟥	●	🟥	●	○🟥	●		●🟩						●	●	●	●
Cognitive	GIDsure (original)			●		●	○🟥	○	●	🟥	●	🟥	●	🟥	🟥		●🟩						●	●	●	●
	Weinshall			●		🟥	🟥	🟥	🟥	🟥	●	🟥	●	🟥	●	○🟩	●🟩				●🟩	●🟩	●	●	●	●
	hopper Blum			●		🟥	🟥	🟥	🟥	●	●	🟥	●	🟥	●	○🟩	●🟩				●🟩	●🟩	●	●	●	●
	Word Association			●		●	●	○	○🟥	●	●	🟥	●	🟥	●		🟥						●	●	●	●
Paper tokens	OTPW			🟥		●	🟥	🟥	●	🟥	●	🟥	●	●	●		●🟩	●🟩	●🟩	●🟩	●🟩	●🟩	●	●	●	●
	S/KEY	●🟩		🟥		●	🟥	○	●	🟥	●	🟥	●	●	●		●🟩	●🟩	●🟩	●🟩	●🟩	○🟩	🟥	●	●	●
	PIN+TAN			🟥		●	🟥	○	○🟥	🟥	○🟥	🟥	●	●	●		●🟩	●🟩	●🟩	●🟩	●🟩	●🟩	○🟥	●	●	●
Visual crypto	PassWindow	●🟩		🟥		🟥	🟥	🟥	🟥	🟥	○🟥	🟥	●	●		○🟩	●🟩	●🟩	●🟩	●🟩	●🟩	●🟩	🟥	●	●	●
Hardware tokens	RSA SecurID			🟥		●	○🟥	○	🟥	🟥	🟥	🟥	●	●	🟥	●🟩	●🟩	●🟩	●🟩	●🟩	●🟩	●🟩	●	🟥	●	●
	Yubikey			🟥		●	○🟥	○	🟥	●	🟥	🟥	●	●	🟥	●🟩	●🟩	●🟩	●🟩	●🟩	●🟩	●🟩	●	🟥	●	●
	Ironkey	○🟩	●🟩	🟥	○🟩	○🟥	○🟥	○	🟥	●	🟥	●	●	●	🟥	●🟩	○			○🟩		●🟩	●	●	●	●
	CAP reader			🟥		●	○🟥	○	🟥	🟥	🟥	🟥	●	●	🟥	●🟩	●🟩	●🟩	●🟩	●🟩	●🟩	●🟩	●	●	●	●
	Pico	●🟩	●🟩	🟥	●🟩	🟥	○🟥	○	🟥	🟥	🟥	🟥	🟥	🟥	●	●🟩	●🟩	●🟩	●🟩	●🟩	●🟩	●🟩	○🟥	●	●	●
Phone-based	Phoolproof			○🟥		●	○🟥	○	🟥	○🟥	○🟥	○🟥	🟥	🟥	●	●🟩	●🟩	●🟩	●🟩	○🟩	●🟩	●🟩	●	●	●	●
	Cronto			○🟥		●	○🟥	○	🟥	🟥	○🟥	🟥	●	●	🟥	●🟩	●🟩	●🟩	●🟩	○🟩	●🟩	●🟩	●	●	●	●
	MP-Auth			○🟥		●	○🟥	🟥	○🟥	○🟥	○🟥	🟥	🟥	🟥	●		○					●🟩	●	●	●	●
	OTP over SMS	●🟩	●🟩	○🟥		●	🟥	○	○🟥	○🟥	🟥	🟥	●	●	●	●🟩	●🟩	●🟩	●🟩	○🟩	●🟩	●🟩	○🟥	🟥	●	●
	Google 2-step			○🟥		●	○🟥	○	○🟥	○🟥	🟥	🟥	●	●	🟥	○🟩	○	●🟩	●🟩		●🟩	●🟩	●	●	●	●
Biometric	Fingerprint	●🟩	●🟩	●	○🟩	●	○🟥	🟥	🟥	○🟥	🟥	🟥	🟥	○🟥	🟥	●🟩	🟥	●🟩					🟥	●	●	🟥
	IRIS	●🟩	●🟩	●	○🟩	●	○🟥	🟥	🟥	○🟥	🟥	🟥	🟥	○🟥	🟥	●🟩	🟥	●🟩					🟥	●	○🟥	🟥
	voice	●🟩	●🟩	●	○🟩	●	○🟥	🟥	🟥	○🟥	○🟥	🟥	○🟥	○🟥	🟥	●🟩	🟥	○🟩					🟥	●	●	🟥
Recovery	Personal knowledge	○🟩		●		●	●	○	●	●	●	🟥	●	●	●		🟥						●	●	●	●
	Preference-based	○🟩		●		●	○🟥	●🟩	○🟥	●	●	🟥	●	🟥	🟥		○					●🟩	●	●	●	●
	Social re-auth			●		●	🟥	●🟩	🟥	●	●	🟥	●	○🟥	🟥	○🟩	🟥	●🟩	●🟩	○🟩	○🟩	●🟩	●	🟥	●	○🟥

●= offers the benefit; ○= almost offers the benefit; *no circle* = does not offer the benefit.
🟩= better than passwords; 🟥= worse than passwords; *no background pattern* = no change.
We group related schemes into categories. For space reasons, in the present paper we describe at most one representative scheme per category; the companion technical report [1] discusses all schemes listed.

Figure 4. Visual evaluation of various schemes and our new blockchain-based scheme (deeID)

the scheme does not rely on trusted third-parties beyond the initial identity offer, therefore they have no control and influence thereafter, if they become compromised it only means that identities should no longer be accepted by the newly untrusted-third-party. The scheme requires Explicit consent from the user for any authentication (physically pressing a button after logging in to their device). Since there is no authenticator, they cannot link whether the same user is logging in to various services. However, if the verifiers do collude and share user information, then of course they will be able to know that they have the same set of users. The point being that with just the authentication scheme itself, they will not be able to link users.

V. Discussion

Human identification is an integral part of the security of the digital ecosystem. Though it is a missing element at the moment which is leading to account breaches, identity theft and much more. In this paper we introduced the Fiat–Shamir scheme and have generalised it so that it could work on a larger scale with *multiple* identity issuers. Our solution has been to use the blockchain as a decentralised PKI. Blockchain-based PKIs has shown promises in research and applications. Emergence of blockchain-based PKIs have come about mainly due to the weak-link security problem with respect to certificate authorities and the X.509 standard [49] (other main PKI standard is OpenPGP [50]). Consequently, this has led to new approaches such as the Google Certificate Transparency project [51]. Blockchain PKI models such as [52] attempt to overcome the weaknesses of the current PKI standards.

Essentially we have a p2p network and we have to represent the human identities on this network. How should the identity be represented? and what mechanisms are required to interact with the identity? Going back to our four rules: a) Capture: each identity must have a unique representation on the network. Each deeID contract has a unique address. b) Recording: It must be machine and human readable. The deeID contract address is unique and machine-readable. The user can also prove their identity and give their human readable names to the challenger. c) Security: can we protect from impersonation and other attacks? Best current method is using public-key cryptography and *what we posses* rather than *what we know* or *what we are.* d) Verification: Using the mobile phone device and the Fiat–Shamir scheme we can verify identities easily. Similar to uPort we have represented the identity by a smart-contract which is uniquely identified in the network. Therefore, each identity has an interactive code on the network (managed by the identity holder themselves). So, here, identity is no longer a piece of string such as your full name, but a totally unique digital representation on a network. The benefit of this is that it is a single-source of truth that everyone can use and agree upon. So, your bank, hospital and school can all use the same identity. Using the same identity and authentication scheme can provide data sharing which in itself has huge benefits. We believe merging the Fiat–Shamir scheme with the existing blockchain identity methods is a novel method for an identification system.

The benefit of using the Fiat–Shamir scheme over simply signing an identity string is the privacy and security it provides. Moreover it saves transactions and space on the network. However, it is still important to keep our 'full names', the basic string identity, because we humans still need to communicate with one another and be able to identify each other. Which is not possible if we are identified by a 512bit unique identity string. There have been studies such as [53], *'Towards reliable storage of 56-bit secrets in human memory'*, where the authors tried to see if we humans can remember long bits of string. The conclusion was that humans are able to learn cryptographic secrets (56-bit in this case), though it requires significant learning periods and other limitations such as recalling times.

The issue of how do we trust the identity issuer falls upon the verifier. At the moment it is conceived to be a simple linear trace back of identities until the final link must somehow be verified. In our implementation we have a simple link to an official website or regulatory-body portal. However, there is considerable work to be done in improving this, e.g. one can quantify trust and provide recommendation to the verifier whether they are trust-worthy. A reputation-based network can provide some metric for verifiers to work from. However, one must consider the ethical implications of quantifying reputation, especially when we are considering real identities.

Using deeID as an alternative to password for authentication provides many benefits beyond better the raw security of password itself. One way of having ones password and personal details compromised is when the either the website (third-party libraries), browser or OS are compromised; deeID simply reads a QR code and then transfers (can be encrypted too) the relevant information to another server using over a WebSocket connection. Thus, the browser, OS and scrapers on the website won't be able to steal entered information. British Airways is reportedly being fined £183 million [54] for the 2018 breach of their site where thousands of customer's personal information were stolen. The attackers compromised a script on the site [55] and were able to capture data as it was being entered on the page. Even *CVV* (3 numbers at the back of bank card) numbers that no organisation stores were stolen. deeID can inherently overcome security threats such as these, too. We are using public-keys and signature schemes as opposed to password for authentication, we have four different ways of authenticating ourselves. Some are achieved by allowing the the user to store various public-keys on their identity smart-contract. The user may also remove keys if they believe it has been compromised or is no longer valid.

In our evaluation we followed the framework described in [2] to do a comparative analysis of deeID with existing authentication schemes. Schemes such as: passwords, password managers, federated systems, hardware, mobile phone based systems, biometrics and more. The deeID protocol uses cryptographic keys that are unpractical for humans to read, copy and transfer themselves without the help of a machine. Moreover, the computation required for these keys is impossible for humans to do. Thus, the use of a machine such as a mobile phone is required. Consequently, our protocol blurs the lines between hardware tokens, phone-based systems, federated and password management systems as shown in figure 1.

Our evaluation based on the framework from [2] indicates that between the three categories of usability, deployability and security, our protocol betters federated systems in security, betters hardware tokens and phone-based systems in deployability. And these two systems are the two closest that seem to better passwords based on the comparative factors. Our protocol is inherently a password manager too, as the device used to manage keys and do the necessary computation on behalf of the identity holder. It can also store and manage passwords. But, this is merely a backward-compatible feature, as we envision a future that does not use password-based schemes.

It should be noted that the framework in figure 1 is designed for the practicality of a human as it has factors such as 'nothing to carry', 'physical-effortlessness' and 'memory-wise-effortless'. Therefore, the protocols are very human-centric at the moment. Nevertheless, as we progress into the domains of AI and IoT we can improve upon these protocols and frameworks to cater to other non-human entities too. Also note that no other system can better passwords that are stored and retrieved easily in our brains but as the number of passwords to remember grows then we start running into problems and thus other schemes such as password managers' benefits become more apparent.

The applications of deeID are numerous. The clear use case is for authentication and an alternative to password. Next, identification via the Fiat–Shamir protocol for online services such as e-commerce. Preventing SIM swap attack is another use case of deeID. SIM swap attack is where an attacker uses social-engineering (operator error and weakness) to convince the user's mobile carrier to swap the SIM to the attacker's SIM. The consequence of this is that the attacker can hijack accounts that use two factor authentication (2FA). Notable hacks include the hijacking of celebrity Instagram accounts [56] and individuals losing millions of pounds of cryptocurrency as their exchange accounts were linked to their mobile phone number using 2FA. This is a growing threat that is leading to more organised crime [57]. Preventive techniques include using a PIN or better 2FA [58]. However, these are patchy and preventive measures; a standardised solution through the use of decentralised identity like that of deeID across mobile network carriers would be a better solution. Explicitly linking the user's deeID to their SIM and having a hardware linked crypto keys - meaning that it is physical and malicious attackers cannot carry out their attack without access to the physical elements. When the SIM is linked to the user's deeID (do this when the user acquires the SIM), the attacker would have to identity themselves which is far more difficult that knowing the user's name, address and date of birth. This brings us to identity theft. An attacker can steal an identity and order a credit card by only knowing the victims name, address and date of birth. Credit checks, for example, should be a push mechanism from the identity owner rather than a pull by knowing the victim's basic personal information. With deeID the process of opening a credit account can be as follows: The credit agencies (Experian [59], Equifax [60] and TransUnion [61]) would encourage their customers to link their data to their deeID. Once a company does a credit check through these credit agencies, the agency would notify the identity owner and request approval rather than automatically granting access. In reality, the economics of this is far more complex, however having a middle layer between the company asking for credit reports and the agency would allow savvy online users to protect their identity.

The system does have its limitations. Most of them are related to the availability of research and technology rather than strong theoretical barriers. To begin with, with respect to identification, trusted third party has not been completely eliminated in this scenario. One can also ask whether if it is truly possible to get rid of the trusted third party at all? nevertheless, in this system the trusted third party, i.e. identity issuer, is only required at the beginning of the identity life-cycle. It is here that we would like to introduce the notion of dynamic trusted third party as opposed to a static one. One may think of the miners in the Bitcoin or Ethereum network as dynamic trusted third parties between two individuals transacting. In the Fiat–Shamir scheme, the identity issuer is a trusted third party but it is only required at the beginning.

A security issue with any decentralised system is the potential threat from powerful entities such as state actors. If there is enough will power then they will be able to bring down most systems either directly or indirectly. This is important for a system that touches on security and privacy. Related to this is the consensus protocol of the blockchain system used. Our implementation used Ethereum and its proof-of-work consensus scheme. This consensus scheme has limitations in energy usage, trust of the miners. We believe alternative consensus schemes such as Proof-of-Stake or more experimental ones such as Proof-of-Reputation [62] are better suited to a network exclusively built for identity management. The deeID system is a general one - meaning, it can be be implemented on other blockchain technologies, adopted on either permissioned or permisionless network.

Using a blockchain system on a mobile phone is also a challenge as we do not verify the integrity of the network on the mobile phone. In our implementation, we do the signing and transaction operations on the mobile phone but do not run a node as such on the device. In this context, the user would have to rely on another trusted node. An implementation of a light node would be another way of overcoming this challenge. This could form a further branch of future works. Revoking an identity could happen in reality. Currently, with the Fiat–Shamir scheme, if an issuer issues x number of identities based on the public n (product of p and q) then revoking a single identity means revoking identities for all individuals using n. This is clearly a challenge and issuing and re-issuing identities if one identity need revoking is not efficient.

VI. Conclusion

In this paper we have broken down the idea of identity, its representation and definition and explored how one can one uniquely identify themselves. We have compared our blockchain-based identity system with existing systems and typical password systems. We have argued that deeID provides better security than federated, mobile phone-based, and hardware-based systems which in turn provide better security than passwords. Deployability is where most schemes fall back on compared to passwords. However, this is something that cannot be beaten due to 'knowing something' is more deployable than all other systems. Given that our system can inherently be a password manager too, it therefore strikes a better balance between the three major comparison factors of usability, deployability and security.

deeID can also be built on top of other blockchain technologies. Our novel contribution is improving upon identity representation through smart-contracts, allowing the blockchain as a cryptographic key management and a trusted-source of identities and providing a mechanism for people on the network to provide identities to one another. This was achieved by allowing the Fiat–Shamir scheme to have multiple identity issuers through the blockchain. This is most useful in the real world where multiple organisations can issue identities to the same individual.

Future work can be separated into four different pillars, (1) General algorithm optimisation of the Fiat–Shamir cryptosystem. (2) We believe that an identity system such as the one proposed in this paper could become an integral part of future blockchain systems that utilise Proof-of-Authority consensus schemes. A deeID based Proof-of-Authority that is Byzantine Fault Tolerant with some quantifiable method for reputation to manage the network, (3) We have skewed our approach towards human identity, the definition can be debated, however one can extend the scheme to go beyond humans and make it a universal identity system for entities that include IoT devices, animals, fictional entities, and even cyborgs with vision of a decentralised galactic identity network. (4) Lastly, our design has been blockchain implementation agnostic, therefore one can think about interoperability of the scheme and how identities across networks can interact with one another.

Acknowledgement

Sirvan Almasi is funded by the Engineering and Physical Sciences Research Council under the Financial Computing CDT.

References

[1] Dinei Florencio and Cormac Herley. A Large-scale Study of Web Password Habits. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, pages 657–666, New York, NY, USA, 2007. ACM. event-place: Banff, Alberta, Canada.

[2] J. Bonneau, C. Herley, P. C. v Oorschot, and F. Stajano. The Quest to Replace Passwords: A Framework for Comparative Evaluation of Web Authentication Schemes. In *2012 IEEE Symposium on Security and Privacy*, pages 553–567, May 2012.

[3] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system.

[4] Namecoin. https://www.namecoin.org.

[5] Blockstack. https://blockstack.org.

[6] Conner Fromknecht and Dragos Velicanu. CertCoin : A Name-Coin Based Decentralized Authentication System 6 . 857 Class Project. 2014.

[7] Guilherme Vieira Pinto, João Pedro Dias, and Hugo Sereno Ferreira. Blockchain-based PKI for crowdsourced IoT sensor information. In Ana Maria Madureira, Ajith Abraham, Niketa Gandhi, Catarina Silva, and Mário Antunes, editors, *Proceedings of the Tenth International Conference on Soft Computing and Pattern Recognition (SoCPaR 2018)*, Advances in Intelligent Systems and Computing, pages 248–257. Springer International Publishing.

[8] P. Boontaetae, A. Sangpetch, and O. Sangpetch. RDI: Real digital identity based on decentralized PKI. In *2018 22nd International Computer Science and Engineering Conference (ICSEC)*, pages 1–6.

[9] Robert Morris and Ken Thompson. Password security: A case history. 22(11):594–597.

[10] Anupam Das, Joseph Bonneau, Matthew Caesar, Nikita Borisov, and Xiaofeng Wang. The tangled web of password reuse.

[11] Identity theft statistics report. https://www.experian.com/blogs/ask-experian/identity-theft-statistics.

[12] Combating synthetic identity fraud | McKinsey. https://www.mckinsey.com/business-functions/risk/our-insights/fighting-back-against-synthetic-identity-fraud.

[13] Greg Wolfond. A blockchain ecosystem for digital identity: Improving service delivery in canada's public and private sectors. 7(10):35–40.

[14] OpenID foundation website. https://openid.net.

[15] OAuth community site. https://oauth.net.

[16] Security assertion markup language. Page Version ID: 905598209.

[17] Using OAuth 2.0 to access google APIs | google identity platform. https://developers.google.com/identity/protocols/OAuth2.

[18] Facebook. https://www.facebook.com.

[19] Mohammad Mannan and Paul C. van Oorschot. Leveraging personal devices for stronger password authentication from untrusted computers. *Journal of Computer Security*, 19:703–750, 2011.

[20] IronKey. https://www.ironkey.com/en-US.

[21] L. O'Gorman. Comparing passwords, tokens, and biometrics for user authentication. 91(12):2019–2020.

[22] Ethereum. https://ethereum.org.

[23] DIF - Decentralized identity Foundation. https://identity.foundation.

[24] Decentralized Identifiers (DIDs) v1.0. https://www.w3.org/TR/did-core/#a-simple-example.

[25] Y. Liu, Z. Zhao, G. Guo, X. Wang, Z. Tan, and S. Wang. An identity management system based on blockchain. In *2017 15th Annual Conference on Privacy, Security and Trust (PST)*, pages 44–4409.

[26] Xiaoyang Zhu and Youakim Badr. Identity management systems for the internet of things: A survey towards blockchain solutions. 18(12):4215.

[27] uPort.me. https://www.uport.me.

[28] Sovrin. https://sovrin.org.

[29] Hyperledger – Open Source Blockchain Technologies. https://www.hyperledger.org.

[30] Adi Shamir. Identity-based cryptosystems and signature schemes. In *Advances in Cryptology*, Lecture Notes in Computer Science, pages 47–53. Springer, Berlin, Heidelberg.

[31] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Advances in Cryptology — CRYPTO' 86*, Lecture Notes in Computer Science, pages 186–194. Springer, Berlin, Heidelberg.

[32] S Goldwasser, S Micali, and C Rackoff. The knowledge complexity of interactive proof-systems. In *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing*, STOC '85, pages 291–304. ACM.

[33] Uriel Feige, Amos Fiat, and Adi Shamir. Zero-knowledge proofs of identity. *Journal of Cryptology*, 1(2):77–94, June 1988.

[34] Dan Boneh and Matt Franklin. Identity-based encryption from the weil pairing. In *Advances in Cryptology — CRYPTO 2001*, Lecture Notes in Computer Science, pages 213–229. Springer, Berlin, Heidelberg.

[35] Clifford Cocks. An identity based encryption scheme based on quadratic residues. In *Cryptography and Coding*, Lecture Notes in Computer Science, pages 360–363. Springer, Berlin, Heidelberg.

[36] L. C. Guillou and J.-J. Quisquater. A Practical Zero-knowledge Protocol Fitted to Security Microprocessor Minimizing Both Transmission and Memory. In *Lecture Notes in Computer Science on Advances in Cryptology-EUROCRYPT'88*, pages 123–128, New York, NY, USA, 1988. Springer-Verlag New York, Inc. event-place: Davos, Switzerland.

[37] Claus P. Schnorr. Efficient Identification and Signatures for Smart Cards. In *Proceedings of the Workshop on the Theory and Application of Cryptographic Techniques on Advances in Cryptology*, EUROCRYPT '89, pages 688–689, Berlin, Heidelberg, 1990. Springer-Verlag. event-place: Houthalen, Belgium.

[38] A. J. Menezes, Paul C. Van Oorschot, and Scott A. Vanstone. *Handbook of applied cryptography*. CRC Press series on discrete mathematics and its applications. CRC Press.

[39] Murat Yasin Kubilay, Mehmet Sabir Kiraz, and Hacı Ali Mantar. CertLedger: A new PKI model with Certificate Transparency based on blockchain. *Computers & Security*, 85:333–352, August 2019.

[40] L. Dykcik, L. Chuat, P. Szalachowski, and A. Perrig. BlockPKI: An Automated, Resilient, and Transparent Public-Key Infrastructure. In *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 105–114, November 2018.

[41] A. Yakubov, W. Shbair, and R. State. BlockPGP: A Blockchain-Based Framework for PGP Key Servers. In *2018 Sixth International Symposium on Computing and Networking Workshops (CANDARW)*, pages 316–322, November 2018.

[42] Christos Patsonakis, Katerina Samari, Aggelos Kiayias, and Mema Roussopoulos. On the Practicality of Smart Contract PKI. *2019 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPCON)*, pages 109–118, April 2019. arXiv: 1902.00878.

[43] E. Karaarslan and E. Adiguzel. Blockchain Based DNS and PKI Solutions. *IEEE Communications Standards Magazine*, 2(3):52–57, September 2018.

[44] A. Singla and E. Bertino. Blockchain-Based PKI Solutions for IoT. In *2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC)*, pages 9–15, October 2018.

[45] Conner Fromknecht, Dragos Velicanu, and Sophia Yakoubov. A Decentralized Public Key Infrastructure with Identity Retention. *IACR Cryptology ePrint Archive*, 2014:803, 2014.

[46] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to Construct Random Functions. *J. ACM*, 33(4):792–807, August 1986.

[47] Ahmed Aleroud and Lina Zhou. Phishing environments, techniques, and countermeasures: A survey. *Computers & Security*, 68:160–196, July 2017.

[48] Bryan Parno, Cynthia Kuo, and Adrian Perrig. Phoolproof phishing prevention. In Giovanni Di Crescenzo and Avi Rubin, editors, *Financial Cryptography and Data Security*, Lecture Notes in Computer Science, pages 1–19. Springer Berlin Heidelberg.

[49] X.509 - Information technology - Open Systems Interconnection - The Directory: Public-key and attribute certificate frameworks. https://www.itu.int/rec/T-REC-X.509, urldate = 2019-11-30.

[50] David Shaw, Lutz Donnerhacke, Rodney Thayer, Hal Finney, and Jon Callas. OpenPGP Message Format. https://tools.ietf.org/html/rfc4880.

[51] Certificate Transparency. http://www.certificate-transparency.org, abstract = This site describes the Certificate Transparency effort being spearheaded by Ben Laurie, Adam Langley and Stephen McHenry. The effort is designed to significantly increase the security of the Public Key Infrastructure used by web sites and services., urldate = 2019-11-30.

[52] Artem S. Konoplev, Alexey G. Busygin, and Dmitry P. Zegzhda. A Blockchain Decentralized Public Key Infrastructure Model. *Automatic Control and Computer Sciences*, 52(8):1017–1021, 2018.

[53] Joseph Bonneau and Stuart Schechter. Towards reliable storage of 56-bit secrets in human memory. In *Proceedings of the 23rd USENIX Conference on Security Symposium*, SEC'14, pages 607–623. USENIX Association. event-place: San Diego, CA.

[54] British Airways faces record £183m fine for data breach. *BBC News*, July 2019.

[55] Jordan Bishop. This Is How 380,000 British Airways Passengers Got Hacked. https://www.forbes.com/sites/bishopjordan/2018/09/11/how-british-airways-got-hacked.

[56] Karissa Bell. Instagram users are reporting the same bizarre hack. https://mashable.com/article/instagram-hack-locked-out-of-account.

[57] Lorenzo Franceschi-Bicchierai. The SIM Hijackers. https://www.vice.com/en_us/article/vbqax3/hackers-sim-swapping-steal-phone-numbers-instagram-bitcoin, July 2018.

[58] How to Protect Your Phone Against a SIM Swap Attack. wired. https://www.wired.com/story/sim-swap-attack-defend-phone.

[59] Experian plc - Credit Agency. https://www.experianplc.com.

[60] Equifax UK - Credit Agency. https://www.equifax.co.uk.

[61] TransUnion - Credit Scores, Credit Reports & Credit Check. https://www.transunion.com.

[62] Qianwei Zhuang, Yuan Liu, Lisi Chen, and Zhengpeng Ai. Proof of Reputation: A Reputation-based Consensus Protocol for Blockchain Based Systems. In *Proceedings of the 2019 International Electronics Communication Conference*, IECC '19, pages 131–138, New York, NY, USA, 2019. ACM. event-place: Okinawa, Japan.

[63] Daniel Shanks. Five number-theoretic algorithms. *Proceedings of the Manitoba Conference on Numerical Mathematics and Computing.*, VII:51–70, 1973.

[64] Jan-Christoph Schlage-Puchta. On Shanks' Algorithm for Modular Square Roots. *arXiv:1105.1456 [math]*, May 2011. arXiv: 1105.1456.

Appendix A
Pseudo-random functions and the j values

In explaining the identification cryptosystem, Fiat and Shamir, in their paper [31] use the so called pseudo-random function along with a random seed, j. The pseudo-random function f maps an arbitrary string to the range of $[0, n)$, where n is the product of the prime numbers p and q. A pseudorandom function (PRF), essentially, emulates a random oracle in that the output of both should be *indistinguishable* by a polynomially bounded computation. Therefore, all outputs of the pseudorandom function will appear to be random but it of course is a deterministic function.

The j value which is used in calculating the public key, $v_j = f(I, j)$ (I is our identity string), is the random seed in our input to the pseudorandom function. Moreover it acts as an adjustment factor to get a public key, v_j, that is a quadratic residue $\mod n$.

Appendix B
From public-key to private-key using the Fiat–Shamir scheme

Here we will go through a numerical example of how to in our implementation we calculated the private-key key from the public-key as per the Fiat–Shamir scheme [31].

Remarks: The actual cryptosystem uses a function that maps the identity string to a rage of $[0, n)$. We will skip this step. Lastly, the cryptosystem uses a set of k keys, we will create one key, thus $k = 1$.

We start with our secret prime numbers of p and q

$$p = 7$$

$$q = 11$$

$$n = p * q = 77$$

lets assume that $v = f(I, k) = 64$, thus our public key is 64, which is a quadratic residue $\mod n$. Now our goal is to compute the smallest s:

$$s = \sqrt{v^{-1}} \mod n$$

Calculate $\bar{v} = v^{-1} \mod n$ using the extended euclidean algorithm:

$$\bar{v} = 64^{-1} \mod 77$$

$$\bar{v} = 71$$

Now we find the smallest square root for this value, $\bar{v}$.

$$71 \mod 7 = 1$$

$$71 \mod 11 = 5$$

Now calculating the square roots via the Tonelli–Shanks algorithm we obtain ±1 and ±4. Next, we use the Chinese-remainder theorem to combine the roots to find the square roots $\bar{v} \mod n$. And thus for our values of ±1 and ±4 we obtain 15. Thus $s = 15$. The python function that we implemented to create the Fiat–Shamir private-key is shown in Appendix D.

Appendix C
Tonelli-Shanks Algorithm

This is an algorithm developed by Daniel Shanks in 1973 [63] which has roots to the work of Alberto Tonelli in 1891. An algorithm to solve an equation of the following kind (where p is a prime):

$$x^2 \equiv n \mod p$$

Algorithm 1: Finding square roots modulo x prime p - [64]

```
1. Set $k = n, z = u^q, x = a^{(q+1)/2}, b = a^q$.
2. Let $m$ be the least integer with $b^{2^m} \equiv 1 \mod p$.
3. Set $t = z^{2^{k-m-1}}, z = t^2, b = bz, x = xt$.
4. If $b = 1$, stop and return $x$, otherwise set $k = m$
   and go to step 1.
```

Appendix D
Python function - generating Fiat–Shamir secret-key from public-key

```
'''
GENERATE SECRET KEY
    v = public key
    p and q = secret factors of n
    n = modulus, product of p & q
'''
def genSecretKey(v, p, q, n):
    v = egcd(v, n)[1] % n
    b1 = tonelli(v % p, p)
    b2 = tonelli(v % q, q)

    # Square root signs
    a = [b1, b2, b1, b2*-1, b1*-1,
        b2*-1, b1*-1, b2]
    j = 0
    smallest = -1
    for i in range(0,4):
        n = [p, q]
        c = [a[j], a[j+1]]
        cr = chinese_remainder(n, c)
        if (smallest<0):
            smallest = cr
        elif(cr < smallest):
            smallest = cr
        j +=2

    return smallest
```

Appendix E
Example of Authentication

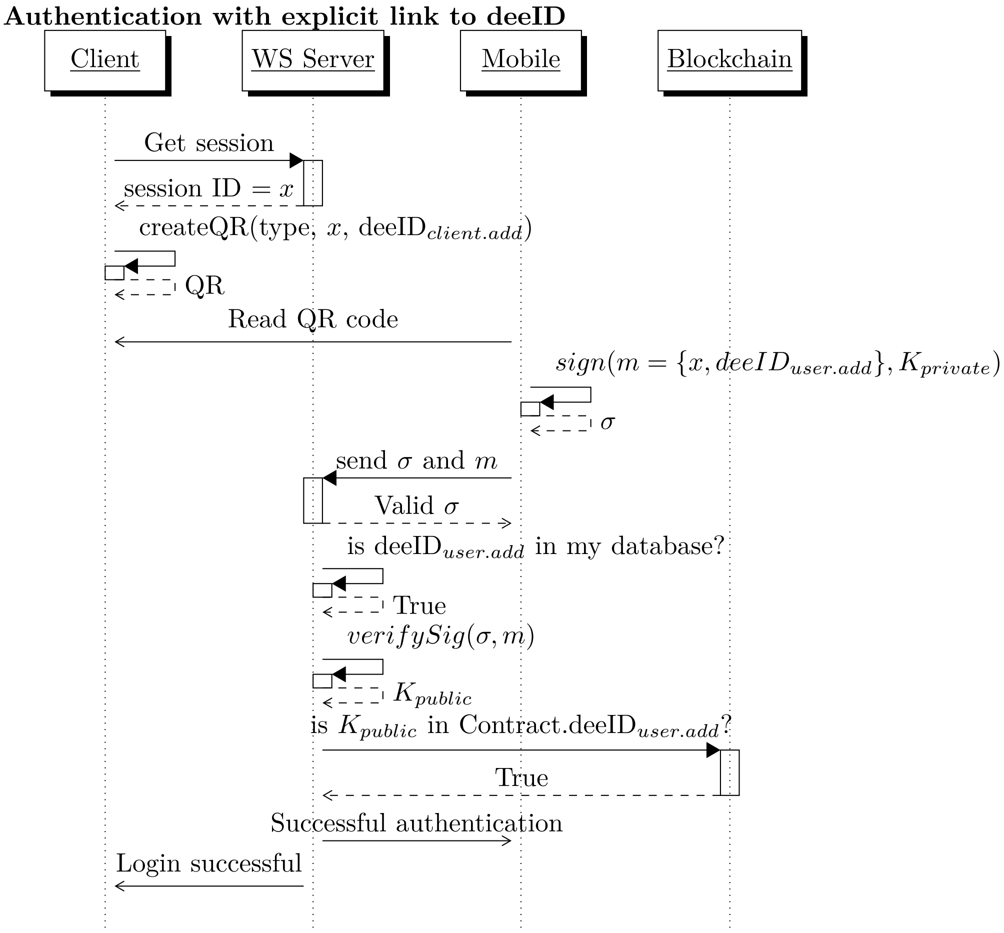

Authentication with explicit link to deeID
Client
WS Server
Mobile
Blockchain
Get session
session ID = x
createQR(type, x, deeID$_{client.add}$)
QR
Read QR code
$sign(m = \{x, deeID_{user.add}\}, K_{private})$
σ
send σ and m
Valid σ
is deeID$_{user.add}$ in my database?
True
$verifySig(\sigma, m)$
K_{public}
is K_{public} in Contract.deeID$_{user.add}$?
True
Successful authentication
Login successful

Figure 5. Sequence of authentication and the steps taken in order to authenticate whilst using deeID with one specific public-key.