

RESEARCH ARTICLE

Double-Blind Proof of Existence for Decentralized Identities

JAYAMINE ALUPOTHA Institute of Computer Science, University of Bern, 3000 Bern, Switzerland
Research and Innovation Centers Division, Rabdan Academy, Abu Dhabi, United Arab Emirates

e-mail: jayamine.alupotha@gmail.com

ABSTRACT Decentralized identities return control of identities to the identity owners. Although current work enhances the privacy of these publicly stored identities using encryption and zero-knowledge proofs, decentralized identities can still be abused due to the following problems:

- *identity holders*, e.g., blockchain peers, can profile identity owners by looking at “who is reading which identity data”, and
- *identity verifiers*, e.g., applications and websites, learn private data about owners, like their monetary values and previous transactions during the identity linking.


In the worst case scenario, the identity holders and verifiers *collaboratively profile* users to learn more information. As a practical solution, we introduce the notion of **Double Blind Proofs of Existence (DBPoE)**, which shows that an opened DID is committed in one of the constant-sized multi-generator Pedersen commitments (33 Bytes at 128-bit security), and *nothing else*. Hence, our DBPoE *double-blinds* identity holders and identity verifiers to mitigate private information leakage. Equally importantly, our multi-generator commitment-based DBPoE is **more resistant** to *graph analysis* than other one-of-many proofs, e.g., ring signatures, which we show mathematically using the maximal flow problem. Our DBPoE protocol has a size complexity of $O(\log_2(N) + m)$ when the real commitment is hidden in N commitments and m generators are used, e.g., when $m = 4$, a DBPoE of 1000 commitments is only 3 KB.

INDEX TERMS Decentralized identities, blockchain, privacy-preserving cryptography.

I. INTRODUCTION

Many privacy policies state that users' data will be shared with “*our affiliates and other trusted businesses or persons*” but do not explicitly state with whom. This simple statement takes away users' privacy entirely and creates a legal loophole in what data holders can do with their users' data. Unfortunately, these types of data sharing have become a common practice, not only in social media platforms but also in sensitive applications like banking and healthcare systems.

Many examples show that buyers can deanonymize them [1], [2] even though the aggregated or anonymous data were shared. It is noteworthy that these cases [1], [2] were only identified because the datasets were public. There is no way to verify whether so-called anonymization is sufficient or not when the data selling happens in private.

The associate editor coordinating the review of this manuscript and approving it for publication was Wei Huang .

Many privacy enthusiasts promote *user-controlled data management* as a countermeasure since users take action and try to protect their data *without* relying on central authorities.

Decentralized Identity Management (DIM): The first step to the user-controlled data management is decentralized identity management [3], [4], [5], [6], [7], [8] where users control their identity information, and these identities are stored in decentralized storage rather than a centralized authority, e.g., blockchain network. A decentralized identity has an identifier called DID (Decentralized Identifier) and an identity document containing more details about the identity. We separate participants in DIM according to their roles.

- **Identity Owner** – The identity belongs to the owner. An identity owner can be an individual, an organization, or even a physical object.
- **Identity Vendor (Issuer)** – While some DID details could be self-issued, other details may be issued by an external party, e.g., a social media platform that issues

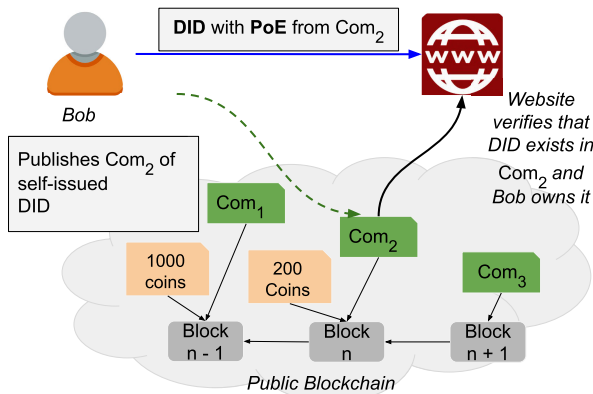


FIGURE 1. An example of traditional DIM.

usernames or a vehicle department that issues licenses. Sometimes, a DID document can combine multiple attributes issued by different vendors.

- **Identity Holder** – The entity who holds identities and/or identities’ proofs of existence, e.g., an identity registry or a blockchain network.
- **Identity Verifier** – The entity who verifies the following: (1) the existence (2) the ownership, e.g., a website that verifies identities for log-in.

Traditional Decentralized Identity Management: In DIM, the protocol works as follows. First, identity owners

- decide what to include in their DID document,
- secure the DID documents, e.g., using encryption or zero-knowledge proofs for sensitive information, and
- send the secured identity or its short proof to the holder.

Since this immutable storage is expensive, owners may store *short proofs* of the secured identity like certificates, hashes, or commitments and keep the secured identity with them. Later, the owners can use these identities for other applications by simply showing the identity and its existence records on blockchains or registries to identity verifiers. Therefore, the identity owner can decide what information to reveal, to whom, and when. For example, as shown in Figure 1, Bob registers his DID in a commitment Com_2 . Later, he can use DID to create an account on a website. For that, Bob sends a proof of existence PoE along with Com_2 and DID. Then the website checks if

- 1) Com_2 is in the blockchain,
- 2) DID is in Com_2 via PoE (**existence**), and
- 3) Bob owns DID via PoE (**ownership**).

Many DID proposals rely on blockchain networks as their identity holders due to their *availability* and *trustless immutability*. However, this transparency creates new ways to *profile and gain private information* about identity owners. In this paper, we address the following two privacy issues by introducing Double Blind Proof of Existence.

- 1) **Identity Holders** like **blockchain peers** gain information from who reads which data. For example, when the website reads Bob’s DID, the blockchain peer could derive that Bob is a user of the website

(Figure 1). This unintended information gain is an issue for blockchains as well since many users do not maintain their blockchain copy but rely on *resourceful peers* [9], [10], [11] via application interfaces. Those peers gain a significant amount of private data about users and their activities. Similar to centralized identity holders, these user data are also shared with third parties, e.g., “we process and collect may be transferred between companies, Services, and employees affiliated with us as a normal part of conducting business”. Even though these resourceful peers use anonymizing techniques, deanonymization could be possible [1], [2].

- 2) **Identity verifiers**, e.g., websites, individuals’ applications, or organizations’ clients, learn more information about individuals and organizations when DID are directly published on the blockchain or linked to the blockchain, e.g., previous transactions, monetary values, or relations to others [12], [13], [14], [15], [16], [17], [18]. From Figure 1, the website learns that Bob owns 200 coins.

In short, encrypting data, zero-knowledge proofs, or private information retrieval cannot solve the above-mentioned privacy issues. Also, these two problems must be solved simultaneously since both roles (blockchain peer/database and identity verifier) may be played by the **same entity** or they share data.

More importantly, these solutions must be **affordable** and should not overload the database or the blockchain; otherwise, solutions become impractical due to the cost (storing data on blockchains is expensive), and may not be affordable to identity owners, eventually forcing them to use cheap yet no-privacy solutions which turn owners into their products.

A. OUR CONTRIBUTION

We propose the notion of Double Blind Proofs of Existence (DBPoE) to prove the existence of a hidden DID in hiding commitments without directly showing where it is to *mitigate* private information leakage. Our DBPoE implementation on SECP256k1 Elliptic Curve Library is public on <https://github.com/DData-core/secp256k1-did>.

More formally, our special DID commitments with Double Blind PoEs provide the following properties.

- **Hiding Succinct Decentralized Identifiers:** Our constant-sized commitments can hold many DIDs while hiding them with a secret key. More precisely, we use m -generator Pedersen commitments, e.g., when $m = 3$, $C = g^k h_1^{f(DID)_1} h_2^{f(DID)_2} \in \mathbb{G}$ is a Pedersen commitment of two DID documents when the secret key $k \in \mathbb{Z}_q$, and g, h_1, h_2 are nothing-up-my-sleeve generators of multiplicative group \mathbb{G} of prime order q . Here, $f()$ turns a DID document into an element of \mathbb{Z}_q .
- **Double Blinding Proof of Existence (from the Zero-Knowledge Argument):** DBPoE only shows that the disclosed DID document is committed *in one of the given commitments* without revealing

- in which commitment it is residing,
- which generator was used to commit, and
- other DID documents of the same commitment.

Hence, DBPoEs mitigate the discovery of the corresponding commitment in the long term, even if verifiers share/sell DBPoE data with each other. We discuss the **graph analysis** and why our system is more resistant to them than other one-of-many proofs due to the multi-generator commitments in Section V.

Our DBPoE is built from modifying Groth-Kohlweiss Protocol [19] for multi-generator commitments. The asymptotic size complexity of our DBPoE is only $O(\log_2(N) + m)$ when N is the number of commitments and m is the number of generators.

- **Unforgeable Ownership with PoE:** A valid PoE can be only created if the creator knows the corresponding secret key of the commitment. Hence, PoEs cannot be forged even if the attacker knows all hidden DIDs. Thus, the applications can verify the authenticity of the owner by including a time-based or random challenge message, and the same DID can be used many times with fresh challenge messages.

B. PRACTICAL USE OF DBPoE

DIM has a wide range of architectures to fit into different applications. Here, we narrow them down to two types: identities for *organizations* and *individuals*. Organizations or public figures publish identities for their clients or audiences, while individuals only want to share their identities with necessary parties like websites or applications. Due to these different objectives, we propose two DBPoE-based DIM solutions and explain them with two case studies. As a solution to our third case study, we show how these two DBPoE-based DIM solutions can *coexist* to build a DIM suitable for many applications.

1) CASE STUDY: DBPoE-BASED SELF-SOVEREIGN IDENTITIES TO REPLACE FEDERATED IDENTITIES

Many Internet users have roughly 60-150 different digital identities. Federated Identity Management (FIM) [20], [21] was introduced to make identity management easy for users by allowing them to reuse an identity issued by a reputable vendor, e.g., reusing a social media profile or email. However, this simplicity comes at *the cost of privacy* since the vendor can trace user activities across multiple applications.

Why blockchain-based DIM? Many free account applications like shopping websites or social media require users to have some form of legitimacy, like an email address or a phone number, to prevent attackers from overloading the system with *fake accounts*. DIDs can replace these federated identities since blockchains add a **monetary-based legitimacy** to self-issued identities with immutability. Note that this kind of monetary legitimacy can only be bought with decentralized systems since if the user pays someone or some organization for identities, users' applications will contact the centralized entity, and all user activities are again visible

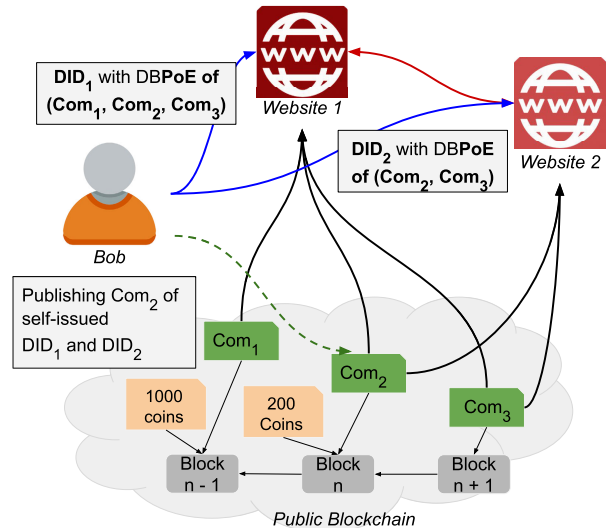


FIGURE 2. Cost-effective and double-blind Decentralized Identities from multi-generator commitments. Here, DID₁ and DID₂ cannot be linked together or to the commitment com₂ when they do not have any linkable information even if Website 1 and Website 2 collaborate with each other.

to that centralized entity. Hence, blockchain-based DID is a better option to reduce the number of fake accounts while giving control to the identity owners, not to a centralized vendor as explained in [22], [23], and [24].

Issues with blockchain-based DIM. While this setup solves the centralized tracing, it creates the previously mentioned problems.

- Many users (e.g., mobile applications) do not keep their own blockchain copy but rely on resourceful peers via application interfaces. Hence, peers learn identity owners' activities when applications or DID verifiers request DIDs from those peers.
- A blockchain includes other data like monetary values or relations to other identities. The application or DID verifiers learn the user's data on the blockchain when a user shares DID directly.

DBPoE-based Solution. We propose off-chain DBPoE-based DIM for individuals where multiple DIDs of the same identity owner are hidden in a multi-generator Pedersen commitment. Later, the identity owner can open one of the DIDs to an identity verifier by sending multiple commitments with an offline DBPoE to prove that the opened DID is in one of the commitments.

We modify the previous example in Figure 1 with our DBPoE solution (see Figure 2). Here, Bob stores two DIDs with no linkable information in com₂. Then, he sends DIDs and their double-blind DBPoEs with commitment sets, including com₂, to websites to create accounts. Once the websites receive DBPoEs, they verify the existence and ownership using these shared commitment sets. However, unlike traditional DIM, they can not identify the exact commitment even if the websites share information. Thus, Bob can hide how many coins he has. Also, a website's blockchain peer(s) only sees that the website is reading

multiple commitments and cannot exactly identify which commitment's owner is a user of that website.¹

These multi-generator Pedersen commitments are only 33 bytes, and our DBPoE has logarithmic size complexity. Hence, the identity owners can use large commitment sets at an affordable price to gain more privacy, e.g., storing a 33-byte commitment only costs ≈ 0.01 USD in Ethereum, and an offline DBPoE of 1000 commitments is only 3 KB.

2) CASE STUDY: DBPoE-BASED DECENTRALIZED IDENTITY FOR SOFTWARE INTEGRITY

Software companies issue certificates to prevent malicious changes to the software during communication. These certificates' signatures can be used as legal evidence against the company in case of malicious activity.

Why Blockchain-based DIM? Providing software integrity is one of the exhaustively discussed use cases of blockchains [25], [26]. Once the company self-issues an identity or obtains an identity from traditional PKI (Public Key Infrastructure), they share the certificate on a public website or individually with clients. However, unlike traditional PKI, publishing their identities on immutable blockchains makes double identities visible to everyone. Hence, users can see if the company is trying to be malicious to a target, e.g., the company targets Alice with malicious software but shares proper software with others. Also, the company can identify if someone tries to pretend to be them with similar information. These features are difficult to achieve in traditional PKI but easy with blockchain-based PKI due to the immutability and transparency.

Issues with blockchain-based DIM. However, directly publishing certificates creates the following problem.

- Software buyers learn the company's previous transactions, monetary values, or relations to others through the blockchain history [12], [13], [14], [15], [16], [17], [18].

DBPoE-based Solution. We propose a layer 3 decentralized registry DBPoE-based DIM where the blockchain(s) holds hiding commitments, but the registry dedicated to identities holds certificates *through* DBPoE. As a result, the certificates are public, but they are not linked to any blockchain account. For organizations, publishing DBPoE is more expensive than for individuals' offline communication; however, it also gives them visibility with more privacy and security.

3) CASE STUDY: DBPoE-BASED METAVERSE IDENTITIES TO SOLVE INTEROPERABILITY

Metaverses [27] interconnect many different Internet applications and websites. The biggest issue of metaverses is interoperability since it is not secure or private to share everything with a centralized vendor. Still, we want to connect all metaverse builders to give the best user experience. For example, users can build their own 3D homes and workplaces,

¹We only assume the application layer anonymity in this paper, not the network or the physical layer anonymity.

shop at a digital shopping mall, and play video games using the same account. These applications, i.e., 3D homes, shopping websites, and video games, may be from different vendors, but the user's identity information, like the username and the avatar, has to be shared.

Why Blockchain-based DIM? Blockchains provide the transparency and immutability we need for self-issued identities with monetary-based legitimacy. Hence, Blockchain-based DIM can bring different entities together without a centralized authority. Many metaverse projects are currently building DIM solutions for identities and also for NFT (Non-Fungible Tokens) to connect with content creators (prefer [27], [28] for more details).

Issues with blockchain-based DIM. However, directly publishing identities creates the following privacy issues.

- Blockchain peers who sell APIs to applications or websites learn user activities by looking at who is reading which identities.
- Also, those applications and websites learn users' monetary values and previous activities on the blockchain when the user links the DID.

DBPoE-based Solution. Metaverse identities can belong to individuals or organizations (or public figures). Individuals do not need publicity, while organizations must publish their identities. Hence, we propose a layer 3 decentralized registry where identities can be confidential or public. All public identities with DBPoEs are published on the layer 3 decentralized registry, and owners share confidential identities with a DBPoE directly with applications, as shown in Figure 3.

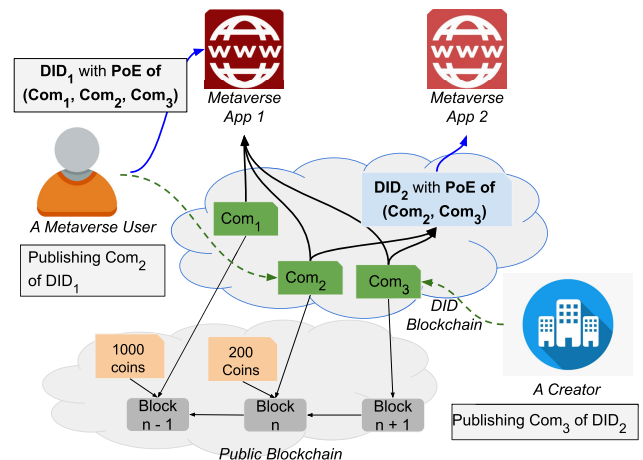


FIGURE 3. Solving interoperability of metaverses when public DIDs and confidential DIDs coexist.

a: ROAD-MAP

We discuss preliminaries, including multi-generator commitments and argument of zero-knowledge, in Section II. DID documents, DID cores, and what to commit from DID documents are explained in Section III. Then, we show how to create a double-blind PoE for commitments without directly

opening DIDs in Section IV. Why DBPoE is stronger than other one-of-many proofs against graph analysis is explained in Section V. Then, we evaluate the performance of DBPoE implementation in Section VI. Finally, we discuss DBPoE solutions, related work, and future work in Section VII.

II. PRELIMINARIES

A. NOTATION

We use “:=” and “=” for assigning, e.g., $a := b$ means that a was assigned to b . For a cyclic group $\mathbb{G} = \langle g \rangle$, g denotes a generator of \mathbb{G} . $\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$ is a ring of modular integers in the range $[0, q - 1]$ for modulus q . Bold letters like \mathbf{a} denote vectors of n elements such that $\mathbf{a} = [a_i]_{i=0}^n = \{a_0, a_1, \dots, a_{n-1}\}$. $m \xleftarrow{\$} \mathcal{M}$ denotes that m is drawn uniformly at random from a set \mathcal{M} . $\epsilon(\lambda) = 1/o(\lambda^c)$ is a negligible function $\forall c \in \mathbb{N}$. We use pp , λ , and \mathcal{A} for public parameters, security level, and p.p.t. adversaries, respectively.

Definition 1 (Discrete Log Problem): For $\mathbb{G} = \langle g \rangle$ of prime order q , $\text{Adv}_{\mathbb{G}}^{\text{DL}}$ for an adversary \mathcal{A} is defined as,

$$\text{Adv}_{\mathbb{G}, \mathcal{A}}^{\text{DL}} := \Pr[y \stackrel{?}{=} g^x \mid y \xleftarrow{\$} \mathbb{G}, x \xleftarrow{\$} \mathcal{A}(y)].$$

The DL problem is (τ, ϵ) -hard if $\mathcal{A}(\tau, \epsilon)$ runs it at most τ times and $\text{Adv}_{\mathbb{G}, \mathcal{A}}^{\text{DL}} \leq \epsilon$.

Definition 2: (Decisional Diffie-Hellman (DDH) Log Problem) For $\mathbb{G} = \langle g \rangle$ of prime order q , $\text{Adv}_{\mathbb{G}, \mathcal{A}}^{\text{DDH}}$ $\leq \epsilon(\lambda)$ for an adversary \mathcal{A} is defined as,

$$2 \left| \frac{1}{2} - \Pr \left[x \stackrel{?}{=} x' \mid \begin{array}{l} (a, b, c) \xleftarrow{\$} \mathbb{Z}_q^3, (y_0, y_1) := (ab, c) \\ x \xleftarrow{\$} [0, 1], x' \xleftarrow{\$} \mathcal{A}(g^a, g^b, g^{y_1}) \end{array} \right] \right|.$$

B. PEDERSEN COMMITMENTS

The original Pedersen commitments [29] and their security properties are explained here. Let COM be the Pedersen commitment scheme [29], defined as follows,

- $\text{COM.set}(\lambda)$: return $pp = (\mathbb{G}, g, h, q)$ \blacktriangleright where $\mathbb{G} = \langle g \rangle = \langle h \rangle$ is a group of prime order $q \in \{0, 1\}^\lambda$, and the discrete logarithms of g and h relative to each other are unknown — they are “nothing-up-my-sleeve” (NUMS) group generators.
- $\text{COM.cmt}(pp, v, k)$: return $C = g^k h^v \in \mathbb{G}$ \blacktriangleright commits value v and key k
- $\text{COM.opn}(pp, C, v, k)$: $C \stackrel{?}{=} g^k h^v \in \mathbb{G}$ \blacktriangleright opens commitment C

COM have the following security properties. Here, we use \mathcal{V}_{pp} for the value space.

Definition 3 (Completeness): When $pp \leftarrow \text{COM.set}(\lambda)$, COM for any λ is complete if

$$\Pr \left[\text{COM.opn}(pp, \text{COM.cmt}(pp, v, r), v, r) \mid \begin{array}{l} v \leftarrow \mathcal{V}_{pp} \\ r \leftarrow \mathbb{Z}_q \end{array} \right] = 1.$$

Definition 4 (Hiding and Binding): For any p.p.t. adversary \mathcal{A} , COM is hiding and binding if,

$$\text{Adv}_{\text{COM}}^{\text{HID}} = \Pr \left[\begin{array}{l} (v_0, v_1) \xleftarrow{\$} \mathcal{V}_{pp}, b \xleftarrow{\$} \{0, 1\} \\ b \stackrel{?}{=} b' \mid \begin{array}{l} r \xleftarrow{\$} \mathbb{Z}_q \\ C \leftarrow \text{COM.cmt}(pp, v_b, r), \\ b' \leftarrow \mathcal{A}(pp, C, v_0, v_1) \end{array} \end{array} \right] - \frac{1}{2} \leq \epsilon(\lambda)$$

$$\text{Adv}_{\text{COM}}^{\text{BND}} = \Pr \left[\begin{array}{l} v_0 \stackrel{?}{\neq} v_1 \wedge \\ g^{r_0} h^{v_0} \stackrel{?}{=} g^{r_1} h^{v_1} \mid (v_0, r_0, v_1, r_1) \leftarrow \mathcal{A}(pp) \end{array} \right] \leq \epsilon(\lambda).$$

We recollect the security theorem of [29] here.

Theorem 1: 2-generator Pedersen commitments are complete, perfectly hiding, and computationally binding if the DL problem and DDH problems are hard.

C. MULTI-GENERATOR PEDERSEN COMMITMENTS

Instead of original Pedersen commitments, we use m -generator commitments where m could be larger than 2. When $m = 2$, this protocol is equivalent to the original Pedersen commitments. This m contributes to the privacy against graph analysis in our DBPoEs. Hence, how to select m and its impact will be explained in Section V. Let COM be a m -generator Pedersen commitment scheme.

- $\text{COM.set}(m, \lambda)$: return $pp = (\mathbb{G}, g, (h_1, \dots, h_{m-1}), q)$ \blacktriangleright where $\mathbb{G} = \langle g \rangle = \langle h_t \rangle$ for any $t \in [1, m - 1]$ is a group of prime order $q \in \{0, 1\}^\lambda$, and g and (h_1, \dots, h_{m-1}) are NUMS group generators.
- $\text{COM.cmt}(pp, v_1, \dots, v_{m-1}, k)$: return $C = g^k \prod_{t=1}^{m-1} h_t^{v_t} \in \mathbb{G}$
- $\text{COM.opn}(pp, C, v_1, \dots, v_{m-1}, k)$: $C \stackrel{?}{=} g^k \prod_{t=1}^{m-1} h_t^{v_t} \in \mathbb{G}$

Multi-Generator Pedersen commitments have the following security properties.

Definition 5 (Completeness): When $pp \leftarrow \text{COM.set}(\lambda)$, COM for any λ is complete if $\mathbf{v} \leftarrow \mathcal{V}_{pp}^{m-1}$ and

$$\Pr[\text{COM.opn}(pp, \text{COM.cmt}(pp, \mathbf{v}, r), \mathbf{v}, r) \mid r \leftarrow \mathbb{Z}_q] = 1.$$

Definition 6 (Hiding and Binding): COM is hiding and binding if $\text{Adv}_{\text{COM}, \mathcal{A}}^{\text{HID}} = \epsilon(\lambda)$ and $\text{Adv}_{\text{COM}, \mathcal{A}}^{\text{BND}} = \epsilon(\lambda)$ for any p.p.t. adversary \mathcal{A} , when,

$$\text{Adv}_{\text{COM}}^{\text{HID}} = \Pr \left[\begin{array}{l} (v_0, \neq v_1) \in \mathcal{V}_{pp}^{2 \times m-1} \leftarrow \mathcal{A}(pp) \\ b \stackrel{?}{=} b' \mid \begin{array}{l} b \xleftarrow{\$} \{0, 1\}; r \xleftarrow{\$} \mathbb{Z}_q \\ C \leftarrow \text{COM.cmt}(pp, \mathbf{v}_b, r), \\ b' \leftarrow \mathcal{A}(pp, C, v_0, v_1) \end{array} \end{array} \right] - \frac{1}{2}$$

$$\text{Adv}_{\text{COM}}^{\text{BND}} = \Pr \left[\begin{array}{l} v_0 \stackrel{?}{\neq} v_1 \wedge \\ g^{r_0} \prod_{t=1}^{m-1} h_t^{v_{0,t}} \stackrel{?}{=} \left(\begin{array}{l} v_0, r_0 \\ v_1, r_1 \end{array} \right) \leftarrow \mathcal{A}(pp) \\ g^{r_1} \prod_{t=1}^{m-1} h_t^{v_{1,t}} \end{array} \right]$$

Theorem 2: Multi-generator Pedersen commitments are complete, perfectly hiding, and computationally binding of the DL and DDH problems are hard for each generator.

D. ZERO-KNOWLEDGE ARGUMENT

Our DBPoEs are arguments of zero-knowledge. An argument of zero-knowledge can cryptographically prove some statement without revealing any other information. In our case, the statement proves that an identity exists in the given commitments without revealing other identities, the index of the commitment, the index of the generator, etc. In general, the interactive version of our protocol contains three algorithms: the setup set which produces a Common Reference String (CRS) pp , the interactive prover \mathcal{P} , and the interactive verifier \mathcal{V} . Each interaction between \mathcal{P} and \mathcal{V} of inputs s and t is denoted as $tr = \langle \mathcal{P}(s), \mathcal{V}(t) \rangle$, which can be 1 if \mathcal{V} accepts; otherwise, 0. We call a relation \mathcal{R} is a polynomial-time-decidable ternary relation of $pp \in \{0, 1\}^*$, a witness $w \in \{0, 1\}^*$ of a statement $u \in \{0, 1\}^*$ such that $(pp, u, w) \in \mathcal{R}$. From this, we can define a CRS-dependent language $\mathcal{L} = \{x | \exists w : (pp, x, w) \in \mathcal{R}\}$ to be the set of statements that have a witness in \mathcal{R} . We formally define the Argument of Knowledge below.

Definition 7 (Argument of Knowledge): A protocol $(\text{set}, \mathcal{P}, \mathcal{V})$ is called argument of knowledge for relation \mathcal{R} if it is complete (Definition 8), sound (Definition 10), and has computational witness-extended emulation (Definition 9).

Definition 8 (Completeness): $(\text{set}, \mathcal{P}, \mathcal{V})$ is complete if

$$\Pr \left[\begin{array}{l} (pp, u, w) \in \mathcal{R} \\ \langle \mathcal{P}(pp, u, w), \mathcal{V}(pp, u) \rangle = 1 \end{array} \middle| (u, w) \leftarrow \mathcal{A}(pp) \right] = 1$$

when $pp := \text{set}(1^\lambda)$ and for any p.p.t. adversary \mathcal{A} .

Definition 9 (Computational Witness-Extended Emulation): Let \mathcal{A}_1 and \mathcal{A}_2 be interactive non-uniform p.p.t. adversaries. The protocol $(\text{set}, \mathcal{P}, \mathcal{V})$ has computational witness-extended emulation if there is an p.p.t. emulator \mathcal{E} for all deterministic polynomial time prover \mathcal{P}' such that the following holds

$$\left| \begin{array}{l} \Pr \left[\begin{array}{l} \mathcal{A}_1(tr) \stackrel{?}{=} 1 \\ tr := \langle \mathcal{P}'(pp, u, s), \mathcal{V}(pp, u) \rangle \end{array} \middle| \begin{array}{l} pp := \text{set}(1^\lambda) \\ (u, s) \leftarrow \mathcal{A}_2(pp) \end{array} \right] \\ -\Pr \left[\begin{array}{l} \mathcal{A}_1(tr) \stackrel{?}{=} 1 \wedge tr \stackrel{?}{=} 1 \\ (tr, w) \leftarrow \mathcal{E}^O(pp, u) \end{array} \middle| \begin{array}{l} pp := \text{set}(1^\lambda) \\ (u, s) \leftarrow \mathcal{A}_2(pp) \end{array} \right] \end{array} \right| \leq \epsilon(\lambda).$$

for an Oracle $\mathcal{O} = \langle \mathcal{P}'(pp, u, s), \mathcal{V}(pp, u) \rangle$, and allows to rewind the protocol to any point and continue with new random coin tosses for the verifier.

Definition 10 (n-Special Soundness): Let there be an efficient extraction algorithm \mathcal{X} that can compute the witness from n “valid” transcripts, $[(x_i, s_i)]_{i=1}^n$ such that $\forall x_i \neq x_j$ when $i \neq j$ with the same initial message a of the prover. Then $(\text{set}, \mathcal{P}, \mathcal{V})$ is n -special sound if \mathcal{X} outputs a valid witness

for \mathcal{R} such that $\text{Adv}_{\text{DK}, \mathcal{A}}^{\text{KS}}$ is,

$$\Pr \left[(pp, u, w) \stackrel{?}{=} \mathcal{R} \middle| \begin{array}{l} (u, a, [(x_i, s_i)]_{i=1}^n) \leftarrow \mathcal{A}(pp) \\ w \leftarrow \mathcal{X}(u, a, [(x_i, s_i)]_{i=1}^n) \end{array} \right] = 1 - \epsilon(\lambda)$$

when $pp := \text{set}(1^\lambda)$ and for any p.p.t. adversary \mathcal{A} .

This special-soundness guarantees that a transcript can be only created by a prover who knows the witness.

In this paper, we are more interested in non-interactive protocols where only the last transcript is exchanged between the prover and the verifier. To simulate them cryptographically, we turn into the public coin argument of special honest verifiers where all messages from verifiers are chosen from a public uniformly random tape independently from the prover’s messages. In other words, there is a simulator that can simulate the whole argument if the verifier knows random challenges in advance. We define the public coin zero-knowledge argument below.

Definition 11 (Public Coin Argument of Zero-Knowledge): Public coin argument of zero-knowledge of $(\text{set}, \mathcal{P}, \mathcal{V})$ is a special honest verifier zero-knowledge argument if there exists a p.p.t. simulator \mathcal{S} such that

$$\left| \begin{array}{l} \Pr \left[\begin{array}{l} (pp, u, w) \in \mathcal{R} \\ \wedge \mathcal{A}_1(tr) \stackrel{?}{=} 1 \end{array} \middle| \begin{array}{l} pp := \text{set}(1^\lambda) \\ (u, w; \rho) \leftarrow \mathcal{A}_2(pp) \\ tr := \langle \mathcal{P}'(pp, u, w), \mathcal{V}(pp, u; \rho) \rangle \end{array} \right] \\ -\Pr \left[\begin{array}{l} (pp, u, w) \in \mathcal{R} \\ \wedge \mathcal{A}_1(tr) \stackrel{?}{=} 1 \end{array} \middle| \begin{array}{l} pp := \text{set}(1^\lambda) \\ (u, w; \rho) \leftarrow \mathcal{A}_2(pp) \\ tr \leftarrow \mathcal{S}(pp, u; \rho) \end{array} \right] \end{array} \right| \leq \epsilon(\lambda)$$

for non-uniform p.p.t. adversaries \mathcal{A}_1 and \mathcal{A}_2 .

In real-life, applications are secured from these arbitrary verifiers in the CRS model using standard techniques like [30] with a small overhead.

E. GROTH-KOHLWEISS PROTOCOL

Our DBPoE protocol is an improvement of Groth-Kohlweiss proofs [19] that allows proving that one-out-of-many commitments have a value of “0” with a logarithmic sized complexity, i.e., the communication cost is $O(\log N)$ when there are N commitments. We explain the interactive Groth-Kohlweiss protocol in Figure 4. Let $\delta_{i,j}$ be 1 if $i = j$; otherwise, 0. Also, i_l presents the l th bit of $i < 2^n$ such that i_0 is the least significant bit of i . Let n be $\lceil \log_2(N) \rceil$. For any $j \in [0, N - 1]$ and $[a_l]_{l=0}^n$, we can precompute the following polynomial coefficients $[[p_{i,l}]_{l=0}^n]_{i=0}^N$ such that

$$\begin{aligned} \forall i \in [0, N) : P_i[X] &= \prod_{l=0}^n (\delta_{i_l, j_l} X + (-1)^{\delta_{0, i_l}} a_l) \\ &= \delta_{i,j} X^n + (p_{i, n-1} X^{n-1} + \dots + p_{i, 1} X^1 + p_{i, 0}) \end{aligned} \quad (1)$$

The core of the protocol is that polynomials’ X^n coefficient will be non-zero if and only if $i = j$. Using this, Groth-Kohlweiss protocol proves that there is a “0” value commitment without revealing its index j .

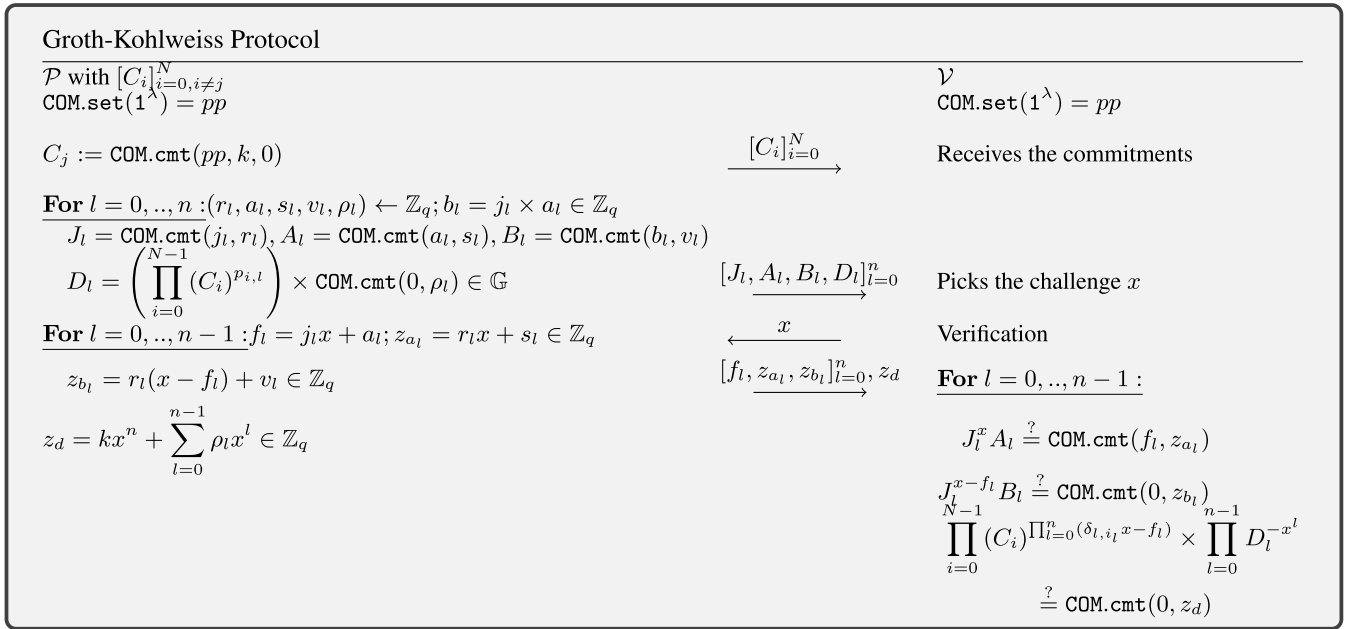


FIGURE 4. Groth-Kohlweiss protocol [19]: Proving one commitment has a value “0” out of N commitments.

DBPoE should be unmalleable, i.e., they have the quasi-unique response property, that is, given an accepting proof, an adversary cannot find a new valid response for the same initial message a and challenge x as follows.

Definition 12: (Quasi-Unique Response) The protocol of $(\text{set}, \mathcal{P}, \mathcal{V})$ has quasi unique responses if for any p.p.t. adversary \mathcal{A} if

$$\Pr \left[\begin{array}{c} \mathcal{V}(pp, u, x, s) \wedge \\ \mathcal{V}(pp, u, x, s') = 1 \wedge \\ s \neq s' \end{array} \middle| \begin{array}{c} pp := \text{set}(1^\lambda) \\ (u, a, x, s, s') \leftarrow \mathcal{A}(pp) \end{array} \right] \leq \epsilon(\lambda)$$

Theorem 3: Groth-Kohlweiss protocol is complete and provides zero-knowledge argument (see [19]).

Theorem 4: Non-interactive Groth-Kohlweiss protocol that replaces verifier’s challenges with random-oracle challenges is complete, has quasi unique responses, and provides public coin zero-knowledge argument.

III. DID CORES AND INTERNET PERSONAS

This section explains what to include in a DID commitment and why an identity owner may need to maintain separate DID documents.

First, we have to identify what should be embedded in commitments. A decentralized identity is associated with a DID document, e.g., the following is an example of W3C DID standardization version 1 [31].

```
{ "@context": [
  "https://www.w3.org/ns/did/v1", \ldots
],
  "id": "did:example:123456789abcdefghi",
  "verificationMethod": [
    {
      "id": "did:example:123#_Qq0UL2Fq651Q0Fj\ldots",
      "type": "JsonWebKey2020",
      "controller": "did:example:123",
      "publicKeyJwk": {
```

```
  "crv": "Ed25519",
  "x": "VCpo2LMLhn6iWku8MKvSLg\ldots",
  "kty": "OKP",
  "kid": "_Qq0UL2Fq651Q0Fjd6TvnY\ldots"
    }
  ], {
    "id": "did:example:123456789abcdefghi#keys-1",
    "type": "Ed25519VerificationKey2020",
    "controller": "did:example:pqrstuvwxyz\ldots",
    "publicKeyMultibase": "\ldots"
  }
}
```

Each DID document contains a Decentralized Identifier (DID), a cryptographically generated self-issued identifier that does not require centralized authority. For example, did:example:123456789abcdefghi and did:web:example.com are two DIDs that use a static identifier and a website name for unique identification, respectively.

In general, DID documents contain essential data like associated public keys, personal claims, and assertions of verifiable credentials. However, not all data are required to commit to the commitment. For example, some personal claims may not be required for all identity verifiers. Still, if we directly commit the entire DID document, it must be presented when we open the commitment. Also, knowing some associated public keys, but not all, may be sufficient to authenticate the users and their verifiable claims through a time-stamped signature. Hence, the DID document’s data required for a particular identity verifier is called DID-core, e.g., the previous DID document could also be the DID core of did:example:123456789abcdefghi#keys-1 for a website. Therefore, we commit hashes of DID cores in Pedersen commitments, which we call DID commitments. Since we use m -generator Pedersen commitments, a single DID commitment can hold at most $(m - 1)$ DID cores.

A. INTERNET PERSONAS AND ANONYMITY

A DID document holds personal information about a person, especially with verifiable credentials. For example, the following is a verifiable credential (verifiable university certificate) issued for `did:example:123456789abcdefghi`.

```
{ "@context": [
  "https://www.w3.org/2018/credentials/v1", \ldots
],
  "id": "http://uni.edu/credentials/3732",
  "type": [
    "VerifiableCredential",
    "UniversityDegreeCredential"
  ],
  "issuer": "https://uni.edu/issuers/",
  "issuanceDate": "2010-01-01T19:23:24Z",
  "credentialSubject": {
    "id": "did:example:123456789abcdefghi",
    "degree": {
      "type": "BachelorDegree",
      "name": "Bachelor_of_Science_and_Arts"
    }
  },
  "proof": {
    "type": "Ed25519Signature2020",
    "created": "2022-02-25T14:58:43Z",
    "verificationMethod": "https://uni.edu/issuers/#key-1",
    "proofPurpose": "assertionMethod",
    "proofValue": "\ldots"
  }
}
```

While these types of verifiable credentials may link digital identities to the real world, many DIDs may represent an online persona without any links to the physical world. In fact, due to the excessive information selling between applications, one may use multiple internet personas to create boundaries and limit tracing. For example, an individual may use different internet personas for work, personal browsing, or different social media platforms. Hence, it is common to have multiple DID documents with different DID identifiers with no unique identifiers.

Recall that multi-generator DID commitments with DBPoE give the same privacy as a commitment of a single DID core. In other words, *a verifier cannot link two DID cores together even if they are committed in the same DID commitment if there are no unique identifiers between DID cores*. Hence, this property is cost-effective for identity owners and promotes privacy-enhancing techniques for individuals rather than cheap yet no-privacy solutions.

IV. DOUBLE BLIND PROOFS OF EXISTENCE

In this section, we explain how to commit DID cores in a way that one DID core can be opened without revealing anything other than the statement, *“the opened DID core is committed in the commitments”*, which hides the secret key, other DID cores of the owner’s commitment, the index of the particular DID commitment, or even the **index** of the DID core. We call this zero-knowledge opening, **Double Blind Proofs of Existence (DBPoE)** of a decentralized identity.

The protocol works as follows (see Figure 5). First, the identity owner publishes the DID commitment, e.g., as a blockchain transaction or registers the commitment in the DID registry. When the identity owner decides to use one of the DID cores in the commitment, he/she sends the

commitment to the identity verifier \mathcal{V} , mixed with other commitments. Then, \mathcal{V} sends a challenge message w to the identity owner. After that, the owner sends a DBPoE of *DID core* DC to \mathcal{V} , which \mathcal{V} verifies against the challenge message w . This challenge may contain application names or timestamps to recognize the identity owner and prevent reusing the DBPoE of other applications or old DBPoEs (see Figure 5).

The owner does not need the openings of other commitments to create a DBPoE; hence, the real DID commitment can be hidden in others’ DID commitments. In that way,

- 1) commitment holders, i.e., blockchain peers and database owners, can not directly identify identity owners’ applications or services, and
- 2) identity verifiers, i.e., applications or services, are unable to trace identity owners’ activities on the blockchain.

Note: These types of one-out-of-many proofs are susceptible to graph analysis in the long run, especially when many DIDs use the same applications or the applications **exchange/sell** DBPoE data with each other. In the next section, we explain how our DBPoE counter-protects against graph analysis for long-term use with m -generator Pedersen commitments even when identity verifiers *collaboratively* try to discover the real DID commitment.

Let DBPoE be the DBPoE protocol.

- $\text{DBPoE.prove}(pp, [C_i]_{i=1}^N, j_N, j_m, k, [DC_i]_{i=1}^{m-1}, w) :$ return π or error \perp \blacktriangleright create a Double Blind PoE π to show that *DID core* DC is committed in one of $[C_i]_{i=1}^N$.
- $\text{DBPoE.verify}(pp, [C_i]_{i=1}^N, \pi, DC, w) :$ return 0/1 \blacktriangleright verify whether DC_i is committed in $[C_i]_{i=1}^N$ or not.

The DBPoE protocol between the prover \mathcal{P} (identity owner) and the verifier \mathcal{V} is explain in Figure 5.

Expected Security Properties: We expect DBPoE to be complete, Existentially Unforgeable against Chosen Challenges (EUF), and to have the argument of zero-knowledge.

Definition 13 (Completeness): DBPoE is complete for some $N \in \mathbb{N}$ and $m \in \mathbb{N}$ if, equation (2) as shown at the bottom of the next page.

We expect PoEs to be unforgeable such that even though the adversary knows the DID core, they should not be able to forge another proof for a *fresh challenge*. We focus on the strong unforgeability where the adversary can request many challenges for the same DID commitment set from the Oracle Π . The adversary’s goal is to generate a fresh pair of proof and a challenge that the oracle has not returned. Otherwise, PoE is unforgeable, or applications can check the existence of DID cores soundly with fresh challenges. The complete definition is stated below.

Definition 14 (Existential Unforgeability Against Chosen Challenge Attacks (EUF)): DBPoE is unforgeable against chosen challenges if $\text{Adv}_{\text{DBPoE}}^{\text{EUF}} = \Pr[\text{Game}_{\text{DBPoE}}^{\text{EUF}}(1^\lambda, N)] \leq \epsilon(\lambda)$ when N is size of the commitments.

$\text{Game}_{\text{DBPoE}}^{\text{EUF}}(1^\lambda, n) :$

$pp := \text{COM.set}(1^\lambda)$

$k \xleftarrow{\$} pp$

\triangleright generate random key

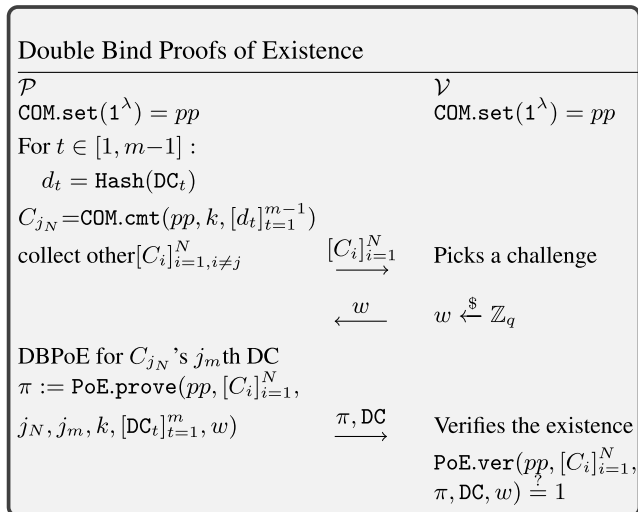


FIGURE 5. Double blind proof of existence protocol.

$[\text{Hash}(\text{DC})_t]_{t=1}^{m-1} \leftarrow \mathcal{A}(pp)$ ▷ get DID core from \mathcal{A}
 $j_N \xleftarrow{\$} [1, N]$ and $j_m \xleftarrow{\$} [1, m-1]$ ▷ indices
 $C_{j_N} := \text{COM.cmt}(pp, k, [\text{Hash}(\text{DC})_t]_{t=1}^{m-1})$ ▷ the commitment
 $[C_i]_{i=1, i \neq j_N}^N \xleftarrow{\$} pp$ ▷ collect other commitments
 $\mathbf{Q} = \{ \}$ ▷ query table
▷ \mathcal{A} requests multiple queries from Oracle Π
 $\{\pi', w'\} \leftarrow \mathcal{A}^{\Pi(\cdot)}(pp, [C_i]_{i=1}^N)$ ▷ \mathcal{A} generates a proof
 return $\{\pi', w'\} \notin \mathbf{Q} \wedge \text{DBPoE.verify}(pp, [C_i]_{i=1}^N, \pi, \text{DC}, w) \stackrel{?}{=} 1$
 Oracle $\Pi_{[C_i]_{i=1}^N, j_N, j_m, k, [\text{DC}_t]_{t=1}^{m-1}, \mathbf{Q}(w)}$:
 $\pi := \text{DBPoE.prove}(pp, [C_i]_{i=1}^N, j_N, j_m, k, [\text{DC}_t]_{t=1}^{m-1}, w)$
 $\mathbf{Q} = \mathbf{Q} \cup (\pi, w)$

We expect the argument of zero-knowledge in Definition 7 to achieve “double blind” property for our DBPoE such that the statement only says that “the opened DID core is committed in one of the given commitments”. Hence, the adversary can not figure out the real commitment and the generator more than the probability of $1/N$ and $1/(m-1)$, respectively, when the DID commitment set size is N , and m generators are used.

A. OUR PROPOSAL

Our PoE proposal allows to commit multiple DID cores within the same DID commitment and only reveals the following zero-knowledge argument, “The opened DID core exists in the DID commitment set, and nothing else”. Our interactive protocol is stated in Figure 6, and the non-interactive protocol used for GitHub Implementation is stated below.

- 1: ► Creates a PoE for the j_m th DID core of $C_{j_N} = \text{COM.cmt}(pp, k, [\text{DC}_t]_{t=1}^{m-1})$ when the verifier’s challenge is w and DID commitment set is $[C_i]_{i=1}^N$.
- 2: $\text{DBPoE.prove}(pp, [C_i]_{i=1}^N, j_N, j_m, k, [\text{DC}_t]_{t=1}^{m-1}, w) :$
- 3: Here, ► $\delta_{i,j} = 1$ if $i = j$; otherwise, $\delta_{i,j} = 0$.
- 4: $c = \text{Hash}([C_i]_{i=1}^N)$
- 5: For $t = [0, m-1]$:
- 6: $d_t = \text{Hash}(\text{DC}_t) \in \mathbb{Z}_q; y_t \leftarrow \mathbb{Z}_q;$
- 7: For $t = [0, m-1]$:
- 8: $Y_t = \prod_{t'=1, t' \neq t}^{m-1} h_{t'}^{y_{t'}} \in \mathbb{G}$
- 9: $x_0 \leftarrow \text{Hash}([C_i]_{i=1}^N, [Y_t]_{t=1}^{m-1}, w, c, d_{j_m}) \in \mathbb{Z}_q$
- 10: For $t = [0, m-1]$:
- 11: $e_t = x_0 d_t + y_t \in \mathbb{Z}_q$
- 12: Arrange $N \times (m-1)$ commitments as follows:
- 13: For $i = [0, N]$: For $t \in [0, m-1]$:
- 14: $C'_{m \times i + t} = (C_i)^{x_0} h_t^{(-x_0 d_{j_m})} Y_t \left(\prod_{t'=0, t' \neq t}^{m-1} h_{t'}^{-e_{t'}} \right) \in \mathbb{G}$
- 15: $n = \lceil \log_2(N \times (m-1)) \rceil$
- 16: $j = j_N m + j_m$ ▷ index of the “0” commitment
- 17: For $l = [0, n]$:
- 18: $r_l, a_l, s_l, v_l, \rho_l \leftarrow \mathbb{Z}_q$
- 19: Find coefficients $[[p_i, l]_{i=0}^{N-1}]_{l=0}^{n-1}$ when
- 20: $\delta_{i,j} X^n + \sum_{l=0}^{n-1} p_{i,l} X^l = \prod_{l=0}^{n-1} (\delta_{i,j_l} X + (-1)^{\delta_{0,i} a_l})$
- 21: $J_l = \text{COM.cmt}(j_l, r_l)$
- 22: $A_l = \text{COM.cmt}(a_l, s_l)$
- 23: $B_l = \text{COM.cmt}(b_l, v_l)$ when $b_l = j_l \times a_l$
- 24: $D_l = \left(\prod_{i=0}^{N(m-1)-1} (C'_i)^{p_{i,l}} \right) \times \text{COM.cmt}(0, \rho_l)$
- 25: $x \leftarrow \text{Hash}(x_0, [C'_i]_{i=0}^{N(m-1)}, [J_l, A_l, B_l, D_l]_{l=0}^n, w, c, d_j) \in \mathbb{Z}_q$
- 26: For $l = [0, n]$:
- 27: $f_l = j_l x + a_l \in \mathbb{Z}_q$
- 28: $z_{a_l} = r_l x + s_l \in \mathbb{Z}_q; z_{b_l} = r_l(x - f_l) + v_l \in \mathbb{Z}_q$
- 29: $z_d = (k^{x_0}) x^n + \sum_{l=0}^{n-1} \rho_l x^l \in \mathbb{Z}_q$
- 30: $\pi = ([J_l, A_l, B_l, D_l, f_l, z_{a_l}, z_{b_l}]_{l=0}^n, z_d, [Y_t, e_t]_{t=0}^{m-1})$
- 31: ► Verify a proof for the DID core DC
- 32: $\text{DBPoE.verify}(pp, [C_i]_{i=1}^N, \pi, \text{DC}, w) :$
- 33: $([J_l, A_l, B_l, D_l, f_l, z_{a_l}, z_{b_l}]_{l=0}^n, z_d, [Y_t, e_t]_{t=0}^{m-1}) := \pi$
- 34: $c = \text{Hash}([C_i]_{i=0}^n)$
- 35: $d = \text{Hash}(\text{DC})$
- 36: $x_0 \leftarrow \text{Hash}([C_i]_{i=0}^n, [Y_t]_{t=0}^{m-1}, w, c, d) \in \mathbb{Z}_q$
- 37: Arrange $N \times (m-1)$ commitments as follows:
- 38: For $i = [0, N]$: For $t \in [0, m-1]$:
- 39: $C'_{m \times i + t} = (C_i)^{x_0} h_t^{(-x_0 d)} Y_t \left(\prod_{t'=0, t' \neq t}^{m-1} h_{t'}^{-e_{t'}} \right) \in \mathbb{G}$
- 40: $n = \lceil \log_2(N \times (m-1)) \rceil$
- 41: $x_0 \leftarrow \text{Hash}([C_i]_{i=0}^n, [Y_t]_{t=0}^{m-1}, w, c, d) \in \mathbb{Z}_q$
- 42: $x \leftarrow \text{Hash}(x_0, [C'_i]_{i=0}^{N(m-1)}, [J_l, A_l, B_l, D_l]_{l=0}^n, w, c, d) \in \mathbb{Z}_q$
- 43: Return 1 if the followings are equal, otherwise, return 0.
- 44: For $l = [0, n]$:
- 45: $J_l^x A_l \stackrel{?}{=} \text{COM.cmt}(f_l, z_{a_l})$
- 46: $J_l^{x-f_l} B_l \stackrel{?}{=} \text{COM.cmt}(0, z_{b_l})$
- 47: $\prod_{i=0}^{N(m-1)-1} (C'_i)^{\prod_{l=0}^{n-1} (\delta_{i,j_l} x - f_l)} \prod_{l=0}^{n-1} D_l^{-x^l}$
- 48: $\stackrel{?}{=} \text{COM.cmt}(0, z_d)$

$$Pr \left[\begin{array}{l} \text{DBPoE.verify}(pp, C_{j_N} := \text{COM.cmt}(pp, k, [\text{DC}_t]_{t=1}^{m-1}) \\ [C_i]_{i=1}^N, \pi, \text{DC}_{j_m}, w) \\ \pi := \text{DBPoE.prove}(pp, [C_i]_{i=1}^N, \\ j_N, j_m, k, [\text{DC}_t]_{t=1}^{m-1}, w) \end{array} \right] = 1 \quad (2)$$

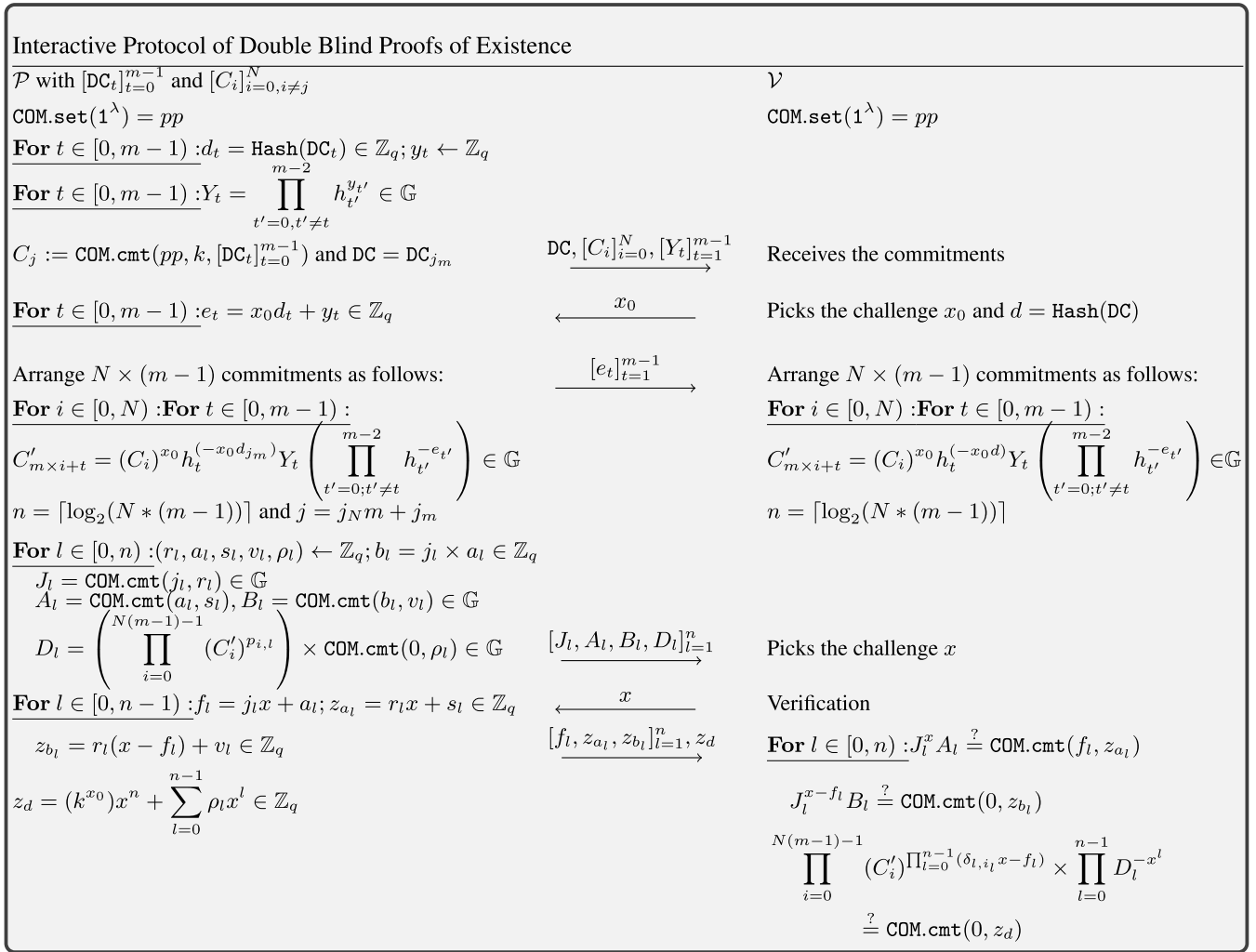


FIGURE 6. Interactive protocol of double blind proofs of existence.

B. SECURITY PROOFS

Theorem 5: The interactive DBPoE protocol is complete and provides the argument of knowledge, including special soundness and computational witness-extended emulation.

Theorem 6: The non-interactive DBPoE protocol is complete, Existential Unforgeable against Chosen Challenge Attacks (EUF), and provides the public coin argument of knowledge of special honest verifiers, including special soundness and computational witness-extended emulation.

1) COMPLETENESS

Here, we show that the arrange commitment set includes a zero value commitment in the $(j_N * m + j_m)$ th index.

$$\begin{aligned}
 & C'_{j_N * m + j_m} \\
 &= (C_{j_N})^{x_0} h_{j_m}^{(-x_0 d_{j_m})} Y_{j_m} \left(\prod_{t'=1; t' \neq j_m}^{m-1} h_{t'}^{-e_{t'}} \right) \\
 &= (C_{j_N})^{x_0} h_{j_m}^{(-x_0 d_{j_m})} \prod_{t'=1, t' \neq j_m}^{m-1} h_{t'}^{y_{t'}} \prod_{t'=1; t' \neq j_m}^{m-1} h_{t'}^{-(d_{t'} x_0 + y_{t'})}
 \end{aligned}$$

$$\begin{aligned}
 &= (g^k \prod_{t'=1}^{m-1} h_{t'}^{d_{t'}})^{x_0} h_{j_m}^{(-x_0 d_{j_m})} \left(\prod_{t'=1, t' \neq j_m}^{m-1} h_{t'}^{-(d_{t'} x_0)} \right) \\
 &= g^{k x_0}
 \end{aligned}$$

Hence, we claim the correctness according to Groth-Kohlweiss Protocol [19].

Lemma 1: DBPoE has public coin argument of knowledge when solving DDH problem is intractable, and Groth-Kohlweiss protocol has public coin argument of knowledge (see Appendix A).

Lemma 2: Let Oracle Π be separated into three oracles \mathcal{Y} for computing Y_t values, \mathcal{H}_0 for x_0 values, and \mathcal{G} for Groth-Kohlweiss protocol. Assume that a p.p.t. adversary makes q_y queries to \mathcal{Y} , q_g queries to \mathcal{G} , q_{x_0} queries to the random oracle \mathcal{H}_0 . The adversary's advantage is at most

$$\begin{aligned}
 \text{Adv}_{\text{DBPoE}}^{\text{EUF}} &\leq \text{Adv}_{\text{DK}, \mathcal{A}}^{\text{quasi-u}} + \text{Adv}_{\mathcal{G}, \mathcal{A}}^{\text{DDH}} \\
 &\quad + O\left(\frac{q_y}{q} + \frac{q_y(q_y + q_{x_0})}{q^2}\right)
 \end{aligned}$$

during time $t' = O(t + \text{poly}(q_y + q_{x_0} + q_g))$ when solving DL problem requires at least time t . The proof is explained in Appendix B.

We claim that Theorem 5 is valid due to the completeness of DBP_{OE}, Lemma 1, and Lemma 2.

V. DBPOE AGAINST THE GRAPH ANALYSIS OF COLLABORATING VERIFIERS

One-of-many proofs are subjected to graph analysis, where the collaborating verifiers try to figure out the exact commitment or at least try to narrow down the space of possible commitments. In this section, we explain the best practices and why our DBPoEs are more resistant to those graph analyses than other one-of-many proofs with examples.

A. BEST PRACTICE 1: SAME COMMITMENT SET FOR THE SAME DID

We start with the best practices of creating DBPoEs for the same DID core. For example, simple analysis would be where a verifier(s) know two DBPoEs of two commitment sets such that (DC, π_1, C_1) and (DC, π_2, C_2) for the same DID core DC. In this case, the verifier(s) can narrow down the possible commitment space to $(C_1 \cap C_2)$ with the hypothesize that the DID core is only committed once due to the financial reasons. Hence, the best practice is to use the same commitment set. In other words, the owner shares the commitment set at the beginning with the verifier and only sends DBPoEs after that. This reduces the communication complexity of sending commitments as well.

B. BEST PRACTICE 2: LARGE COMMITMENT SETS AGAINST THE EXCLUSION ATTACKS

In DBPoE and other one-of-many proofs, the probability of the opened value being in a particular commitment is $1/N$ if there are N number of commitments in the set C . Hence, if the verifier(s) knows N' number of commitments C' out of C , they can narrow down the possible commitment set to $(C \setminus C')$, and the probability increases to $1/(N - N')$. Hence, it is important to include many commitments to the set to increase commitments unknown to the verifier(s). Fortunately, the size complexity of our DBPoE is logarithmic to N ; hence, many commitments can be included efficiently.

DBPoE vs. Other One-of-Many Proofs:

The verifier(s) can narrow down the possible commitment set by turning it into the **maximal-flow problem** [32], [33], [34], [35], [36] when they know many one-of-many proofs and their commitment sets, which we call graph analysis. Let us explain the graph analysis of *other one-of-many Proofs*, e.g., ring signatures [37], [38], [39], [40], [41] or confidential digital asset protocols [42], [43], using a sample analysis. Let there be $[C_1, C_2, C_3]$ and DID cores' commitment sets; $(DC_1, [C_1, C_2])$, $(DC_2, [C_1, C_2])$, and $(DC_3, [C_2, C_3])$. Once we turn the data into a maximal-flow problem (by setting the flow between any vertices into 1), the graph looks like in Figure 7. There are only **two solutions** to the problem as illustrated in Figure 12. From the solutions, we can see

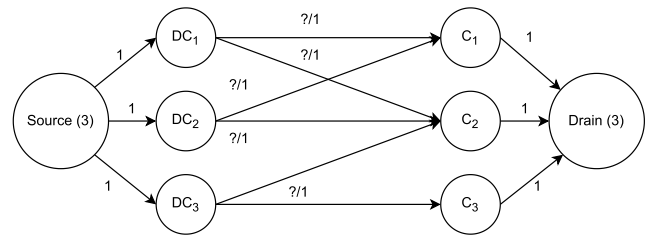


FIGURE 7. Converting other One-of-Many proof data to maximal-flow problem.

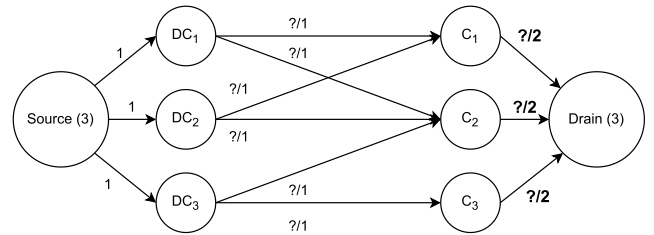


FIGURE 8. Converting our DBPoE data to maximal-flow problem when $m = 3$ (two value generators).

that C_3 is the commitment of DC_3 (the probability is 1 in Table 2). Similarly, the verifiers can considerably narrow down the possible commitments when they have access to large data sets (e.g., collaborating verifiers) even though the commitment set is considerably large in one-of-many proofs.

As discussed before, DBPoEs use multi-generator commitments. Hence, DBPoE are **more** resistant to graph analysis than other one-of-many proofs. For example, if we use the same data for our DBPoE protocol even with $m = 3$ (means two value generators), the flow capacity of a commitment to the drain increases to 2 from 1 as explained in Figure 8. Hence, there are **seven solutions** as shown in Figure 13, and we cannot identify DC_3 's commitment unlike in other one-of-many proofs. Suppose we update m to be $m = 4$ (three value generators). In that case, the probabilities are equal to the initial probabilities, and the graph analysis does not gain any information as shown in Table 5.

TABLE 1. Initial one-of-many data probabilities.

Probabilities	C_1	C_2	C_3
DC_1	1/2	1/2	0
DC_2	1/2	1/2	0
DC_3	0	1/2	1/2

TABLE 2. One-of-many data probabilities after the maximal-flow solutions.

Probabilities	C_1	C_2	C_3
DC_1	1/2	1/2	0
DC_2	1/2	1/2	0
DC_3	0	0	1

TABLE 3. Initial DBPoE data probabilities.

Probabilities	C_1	C_2	C_3
DC ₁	1/2	1/2	0
DC ₂	1/2	1/2	0
DC ₃	0	1/2	1/2

TABLE 4. $m = 3$: DBPoE data probabilities after the maximal-flow solutions.

Probabilities	C_1	C_2	C_3
DC ₁	1/2	1/2	0
DC ₂	1/2	1/2	0
DC ₃	0	3/7	4/7

TABLE 5. $m = 4$: DBPoE data probabilities after the maximal-flow solutions.

Probabilities	C_1	C_2	C_3
DC ₁	1/2	1/2	0
DC ₂	1/2	1/2	0
DC ₃	0	1/2	1/2

C. BEST PRACTICE 3: LARGE M AND N AGAINST GRAPH ANALYSIS

We can mitigate graph analysis by increasing the m and N altogether, i.e., the number of possible DID cores in a N commitments is $N \times (m - 1)$. Also, as we saw in the graph analysis example, the edge between the drain to each commitment increases to $(m - 1)$. Hence, increasing m and N mitigates these graph analyses. In other words, verifiers need more information and more processing time for these attacks [37], [38], [39], [40], [41]. Since the size complexity of our DBPoE is $O(\log_2 N + m)$, computing DBPoE for large N and m is efficient and private.

In the next section, we show how DBPoE size, generation time, and verification time change with N and m .

VI. PERFORMANCE EVALUATION

We implement the DBPoE protocol using SECP256k1 Elliptic Curve Library (public on <https://github.com/DData-core/secp256k1-did>). This section evaluates DBPoE sizes, generation times, and verification times for different N and m values. Note that all times are measured on i7-1065G7 CPU at 1.30GHz as single thread programs. Our DBPoE’s asymptotic time complexity is $O(Nm)$ for DBPoE generation and verification. Figure 9 and Figure 10 show DBPoE generation time and verification times for different N and m values, respectively. The asymptotic size complexity is $O(\log_2(N) + m)$, and the sizes are shown in Figure 11 for different N and m values. Due to these short sizes, e.g., a DBPoE of 1024 commitments ($m = 4$) is only 3KB, DBPoE can be efficiently transferred to verifiers.

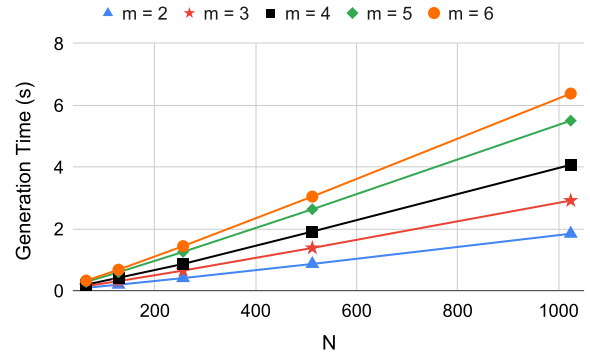


FIGURE 9. Generation time vs. N and m .

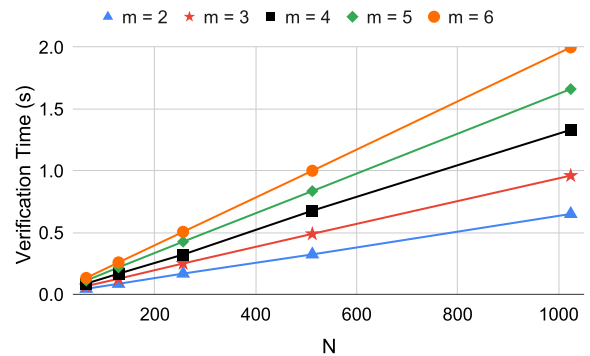


FIGURE 10. Verification time vs. N and m .

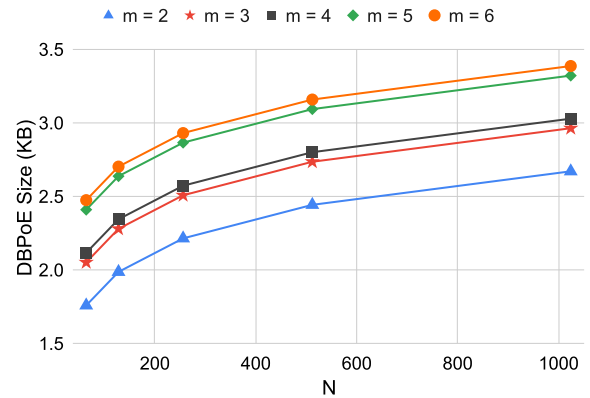


FIGURE 11. DBPoE Size vs. N and m .

VII. DISCUSSION AND RELATED WORK

The need for decentralized identities or user-controlled identities [6], [44], [45], [46] became essential with misuses of identities like selling them to third-parties and the incidents of de-anonymization of anonymous data set [1], [2]. As it appears, the first problem of decentralized identity management is standardization. The World Wide Web Consortium (W3C), with the collaboration of an international community group, released the first version of DID v.1 [31]. The released draft contains methods of primitive identity management and the standards for advanced features like

selective disclosable identities based on BBS+ signatures [47]. In selective disclosable identities [48], the users can decide what to disclose from a DID Document and to whom; hence, disclosed data and their proofs can be seen as an advanced form of DID cores. Therefore, identity owners can use *DBPoE for selective disclosable identities* by updating DID cores with relevant proofs, which gives more privacy with double-blinding.

Note: There are many advanced zero-knowledge protocols [43], [49] that are not standardized by [31] but can be used to improve the privacy of decentralized identities other than the protocols based on BBS+ signatures [47]. DBPoE is also generally compatible with those protocols since the DID core can be modified according to their proofs.

TABLE 6. DIM Solutions and their DBPoE Compatibility. Here, “✓*” means that the native blockchain is only suitable for offline-DBPoEs.

DIM	Double-Blind	DBPoE?
Uport (Ethereum) [10]	✗	✓*
Sovrin (Indy) [50]	✗	✓
ShoCard (Bitcoin) [51]	✗	✓*
IDChainZ [52]	✗	✓
EverID (Ethereum) [53]	✗	✓*
LifeID [54]	✗	✓
SelfKey (Ethereum) [55]	✗	✓*
Civic + Ethereum [56]	✗	✓
TheKey (NEO) [57]	✗	✓
Bitnation (NEO) [58]	✗	✓

Blockchain-based decentralized identity platforms like [10], [50], [51], [52], [53], [54], [55], [56], [57], [58], and [59] were introduced to bring decentralization and immutability of blockchains to identity management. While some of them rely on Public blockchains like Ethereum [60], others build special identity blockchains like Hyperledger Indy [50]. However, they still need to address the privacy issues we address in this paper, that is, how to protect privacy from identity holders and identity verifiers, especially when they collaborate to profile users. Hence, these current implementations can benefit from our efficient DBPoEs to improve user privacy at a little cost. In Table 6, we state current DIM solutions and whether they are compatible with DBPoE or not. Here, “✓*” means that the native blockchain is only suitable for storing multi-generator commitments due to the cost of storing data, and the DBPoE should be shared offline. For example, according to the current prices, storing a 33-byte commitment roughly costs 0.01 USD on Ethereum. Hence, we believe that the affordability of our DBPoE solutions can improve Decentralized Identity Management for day-to-day use.

One-of-many proofs like ring signatures [37], [38], [39], [40], [41] or confidential digital asset protocols [42], [43] are common protocols that provide double-blinding properties for different applications, e.g., [61] and [62]. In general, they are allowed to hide something within a set. However, unlike

these protocols, the DBPoE protocol is built to resist the graph analysis of malicious collaborating verifiers who share data, similar to what happens to decentralized identities. DBPoE protocol overcomes these attacks by using multi-generator commitments, as explained in Section V. We state the related work of DBPoE protocols and their complexities in Table 7. As shown in the table, the DBPoE protocol can commit many DIDs in a single commitment yet be more resistant to graph analysis than any other one-of-many proofs. Our DBPoE-based DID solutions are more affordable and private for regular use.

TABLE 7. Other one-of-many proofs vs. DBPoE protocol.

Protocol	Size	DB	$m > 2$
Digital Assets [42]	$O(N)$	✓	✗
Bulletproofs [43]	$O(\log(N))$	✓	✗
Groth-Kohlweiss [19]	$O(\log(N))$	✓	✗
Log-Ring [38], [39]	$O(\log(N))$	✓	✗
Dual-Ring [40]	$O(\log(N))$	✓	✗
Repudiable Ring [41]	$O(\log(N))$	✓	✗
Our DBPoE	$O(\log(N) + m)$	✓	✓

VIII. CONCLUSION

This work proposes the notion of Double Blind Proofs of Existence (DBPoE) for decentralized identity solutions to improve the privacy of decentralized identities against identity holders and identity verifiers since blockchain peers or applications that share data with others can profile users, learn private information, and deanonymize the pseudonyms of blockchain users. Due to the complexity of logarithmic size, current decentralized identity solution platforms can benefit from this double-blind PoE protocol to give their users more affordable privacy. Our main objective is to improve decentralized identities for regular use. Technical improvements like DBPoE with logarithmic size complexity can motivate researchers and application developers to use Decentralized Identity solutions for regular use, e.g., to replace federated identities or use as interoperable metaverse identities that connect identity holders in a trustless manner.

**APPENDIX A
PROOFS OF PUBLIC COIN ARGUMENT OF KNOWLEDGE**

We prove Lemma 1 in this section. We separate our DBPoE protocol into two sections: arranging the new commitment set and using Groth-Kohlweiss protocol that shows that there is a “0” value commitment. Since we have proven the completeness in Section IV, we directly move to the soundness of DBPoE.

A. (N + 2)-SPECIAL SOUNDNESS

For the first section, witness is the simulated $[y_t, d_t = \text{Hash}(\text{DC}_t)]_{t=0}^{m-1}$. To prove the special soundness of the

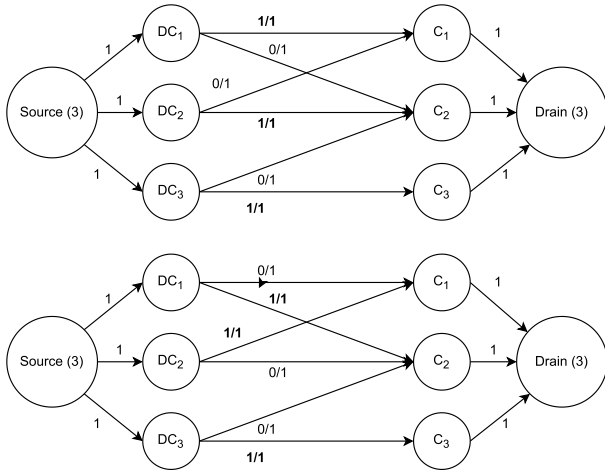


FIGURE 12. Graph analysis of other One-of-Many proof data from maximal-flow problem.

first section, let the initial message be $a = ([DC]_{t=0}^{m-1}, [C_i]_{i=1}^N, [Y_t]_{t=1}^{m-1})$ and the adversary has access to

- $(x_0, [e_t]_{t=0}^{m-1})$ and $(x'_0, [e'_t]_{t=0}^{m-1})$ such that $e_t = x_0 \text{Hash}(DC_t) + y_t$ and $e'_t = x'_0 \text{Hash}(DC_t) + y_t$.

The adversary can compute the simulated hashes of DID cores since for each t , $d_t \leftarrow (e_t - e'_t)/(x_0 - x'_0) \in \mathbb{Z}_q$ or the adversary has to break DL problem of the generators to find another d_t set that returns Y_t .

For the second section, we refer to [19] (pages 9-10), which shows that $(n + 1)$ queries are sufficient to extract a valid witness. Altogether, we claim that DBP_{OE} provides $(n + 2)$ -Special Soundness from Definition 10.

B. SPECIAL HONEST VERIFIER ZERO-KNOWLEDGE

We start with the first section where the adversary gets challenge x_0 in advance for known $(DC, [C_i]_{i=1}^N)$. The adversary first picks $j_m \xleftarrow{\$} [1, m - 1]$ and $[e_t]_{t=1}^{m-1}$ and $[DC_t]_{t=1, t \neq j_m}^{m-1}$ uniformly at random and then computes $[Y_t]_{t=1}^{m-1}$ such that

$$Y_t = \prod_{t'=1, t' \neq t}^{m-1} h_{t'}^{e_{t'} - x_0 \text{Hash}(DC_{t'})} \in \mathbb{G} \quad (3)$$

due to DDH problem. Now, the adversary has witnesses $([DC_t]_{t=1}^{m-1})$ for the first section of the protocol. Hence, the adversary can move to proof of Groth-Kohlweiss protocol (see page 10 in [19]) for the new commitment set $[C'_i]_{i=1}^{N(m-1)}$ such that

$$C'_{m(i-1)+t} = (C_i)^{x_0} h_i^{(-x_0 d_{j_m})} Y_t \left(\prod_{t'=1; t' \neq t}^{m-1} h_{t'}^{-e_{t'}} \right) \in \mathbb{G}.$$

Therefore, we can claim DBP_{OE} protocol has Special Honest Verifier Zero-Knowledge (from Definition 11).

Finally, we claim that Lemma 1 is correct due to DBP_{OE} being complete, $(2(m - 1) + (n + 1))$ -special sound, and special honest verifier zero-Knowledge.

**APPENDIX B
PROOFS OF UNFORGEABILITY**

We separate DBP_{OE} computation of Oracle Π in Game^{EUF}_{DBP_{OE}} of Definition IV into three Oracles; \mathcal{Y} , \mathcal{H}_0 , and \mathcal{G} . For any input $[y_t]_{t=0}^{m-1}$, Oracle \mathcal{Y} computes;

$$\mathcal{Y} : Y_t = \prod_{t'=1, t' \neq t}^{m-1} h^{y_{t'}} \in \mathbb{G}$$

and saves queries and their outputs in \mathbf{Q}_y such that $\mathbf{Q}_y = \mathbf{Q}_y \cup ([y_t]_{t=0}^{m-1}, [Y_t]_{t=0}^{m-1})$. Oracle \mathcal{H}_0 computes the hash challenge x_0 for any input $([C_i]_{i=0}^N, [Y_t]_{t=0}^{m-1}, w, c, d)$ such that

$$\mathcal{H}_0 : x_0 \leftarrow \text{Hash}([C_i]_{i=0}^N, [Y_t]_{t=0}^{m-1}, w, c, d) \in \mathbb{Z}_q.$$

Oracle \mathcal{H}_0 also keeps queries and outputs such that $\mathbf{Q}_h = \mathbf{Q}_h \cup (x_0, ([C_i]_{i=0}^N, [Y_t]_{t=0}^{m-1}, w, c, d))$. Oracle \mathcal{G} computes the non-interactive Groth-Kohlweiss proofs of the rearranged commitments against any $(x_0, [C'_i]_{i=0}^{N(m-1)})$ and saves them in $\mathbf{Q}_g \cup ((x_0, [C'_i]_{i=0}^{N(m-1)}), \pi)$. In other words, for each query, Oracle Π runs calls \mathcal{Y} , \mathcal{H}_0 , and \mathcal{G} once.

Let there be a p.p.t. adversary who makes q_y number of queries to Oracle \mathcal{Y} , q_g number of queries to Oracle \mathcal{G} , and q_{x_0} queries to the random oracle \mathcal{H}_0 .

A. CASE 1

In the initial case, \mathcal{A} makes at most q_y queries to \mathcal{Y} . The game terminates with 1 if output $[Y_t]_{t=0}^{m-1}$ was queried for a different input previously such that $[y_t]_{t=0}^{m-1} \neq [y'_t]_{t=0}^{m-1}$ and

$$Y_t = \prod_{t'=1, t' \neq t}^{m-1} h^{y_{t'}} = \prod_{t'=1, t' \neq t}^{m-1} h^{y'_{t'}} \in \mathbb{G}.$$

Here, the probability of \mathcal{A} winning the game is $Pr_1 \leq q_y/q$ since \mathcal{A} can be simulated to solve DL problem between the generators.

B. CASE 2

In this case, we assume that \mathcal{A} makes at most q_{x_0} queries to \mathcal{H}_0 with inputs it received from \mathcal{Y} . The game returns 1 if output $x'_0 = x_0$ was queried previously for a different input or \mathcal{Y} terminates with 1. At this stage, each input is no longer uniformly random because \mathcal{A} can modify inputs. However, there will be at least one almost uniformly random string in $[C_i]_{i=0}^N, [Y_t]_{t=0}^{m-1}$ due to DDH problem. The probability of \mathcal{A} winning the game is $Pr_2 \leq \text{Adv}_{\mathbb{G}, \mathcal{A}}^{\text{DDH}} + q_y(q_y + q_{x_0})/q^2$.

C. CASE 3

Now, we let the adversary to make q_g queries to \mathcal{G} . Each of these queries triggers a query to \mathcal{Y} and \mathcal{H}_0 . The game returns 1 if the output was queried for different inputs or any of the oracles returns 1. The probability of \mathcal{G} terminating is $Pr_3 \leq \text{Adv}_{\text{DK}, \mathcal{A}}^{\text{quasi-u}}$ since \mathcal{A} can be simulated to break the quasi-uniqueness of non-interactive Groth-Kohlweiss proofs.

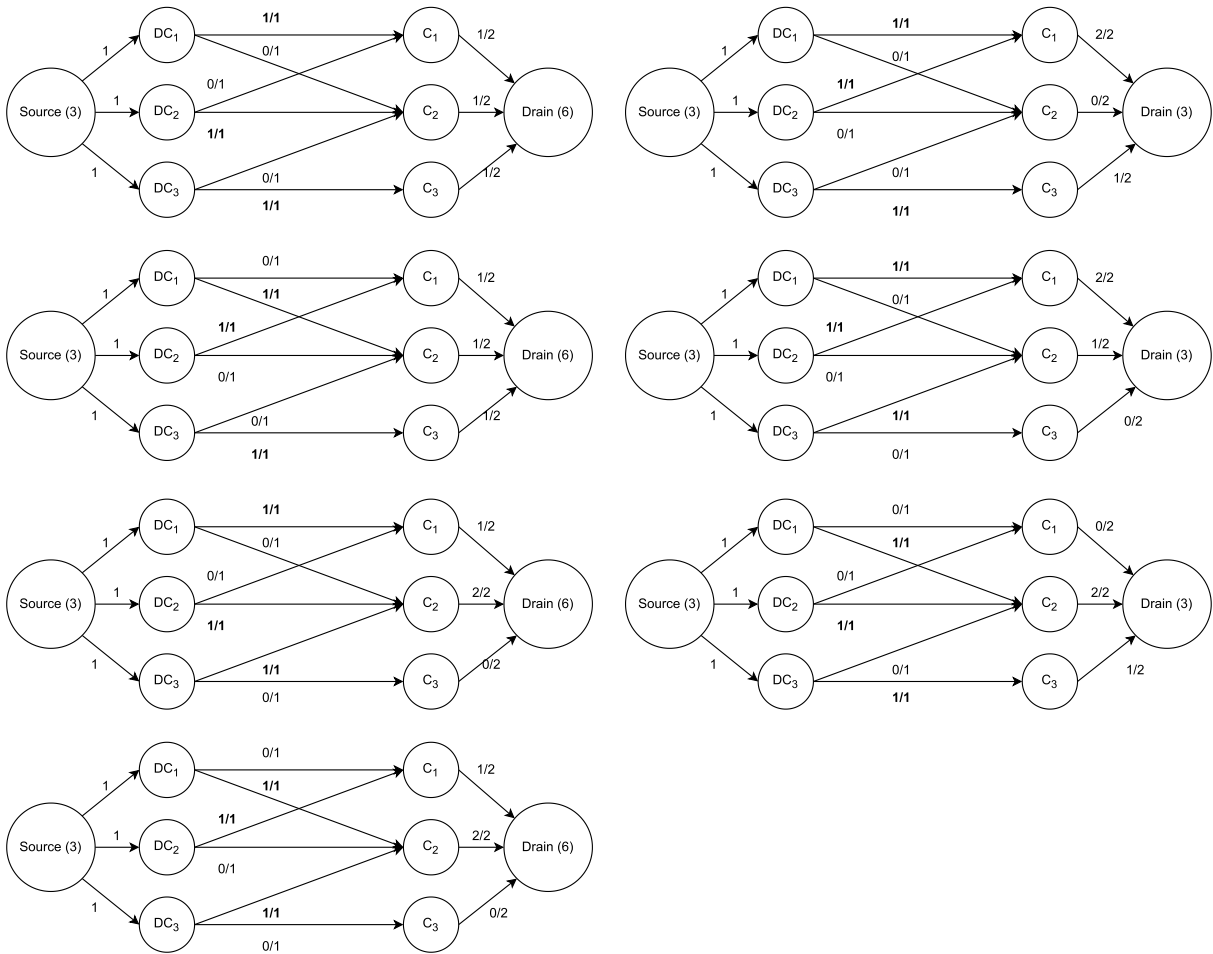


FIGURE 13. Graph analysis of our DBPoE data from maximal-flow problem.

Hence, we see that $Adv_{DBPoE}^{EUF} \leq Adv_{DK, \mathcal{A}}^{quasi-u} + Adv_{G, \mathcal{A}}^{DDH} + O(q_y/q + q_y(q_y + q_{x_0})/q^2)$, and conclude that Lemma 2 is valid.

ACKNOWLEDGMENT

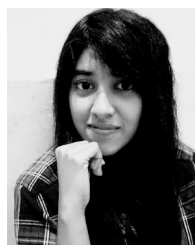
This work was done when the author was a Research Fellow with the Rabdan Academy.

REFERENCES

[1] C. Culnane, B. I. P. Rubinstein, and V. Teague, “Health data in an open world,” 2017, *arXiv:1712.05627*.
 [2] C. A. Kushida, D. A. Nichols, R. Jadmicek, R. Müller, J. K. Walsh, and K. Griffin, “Strategies for de-identification and anonymization of electronic health record data for use in multicenter research studies,” *Med. Care*, vol. 50, pp. S82–S101, Jul. 2012.
 [3] A. Mühle, A. Grüner, T. Gayvoronskaya, and C. Meinel, “A survey on essential components of a self-sovereign identity,” *Comput. Sci. Rev.*, vol. 30, pp. 80–86, Nov. 2018.
 [4] O. Jacobovitz, “Blockchain for identity management,” Lynne William Frankel Center, Dept. Comput. Sci., Ben-Gurion Univ., Beer Sheva, Israel, Tech. Rep., 2016. [Online]. Available: <https://www.cs.bgu.ac.il/~frankel/TechnicalReports/2016/16-02.pdf>
 [5] U. Der, S. Jähnichen, and J. Sürmeli, “Self-sovereign identity—Opportunities and challenges for the digital revolution,” 2017, *arXiv:1712.01767*.

[6] G. Mega, A. Montresor, and G. P. Picco, “Efficient dissemination in decentralized social networks,” in *Proc. IEEE Int. Conf. Peer-Peer Comput.*, Aug. 2011, pp. 338–347.
 [7] S.-W. Seong, J. Seo, M. Nasielski, D. Sengupta, S. Hangal, S. K. Teh, R. Chu, B. Dodson, and M. S. Lam, “PrPI: A decentralized social networking infrastructure,” in *Proc. 1st ACM Workshop Mobile Cloud Comput. Services, Social Netw. Beyond*, Jun. 2010, pp. 1–8.
 [8] C. Fromknecht, D. Velicanu, and S. Yakoubov, “A decentralized public key infrastructure with identity retention,” *IACR Cryptol. ePrint Arch.*, 2014, p. 803. [Online]. Available: <https://eprint.iacr.org/2014/803.pdf>
 [9] Microsoft Identity Platform Team. (Oct. 12, 2022). *Introduction to Microsoft Entra Verified ID*. [Online]. Available: <https://devblogs.microsoft.com/microsoft365dev/learn-how-to-set-up-decentralized-identities-on-azure/>
 [10] uPort. (Oct. 10, 2022). *uPort: Introducing the Next Generation of Decentralized Identity*. [Online]. Available: <https://www.uport.me/>
 [11] Coinbase. (Oct. 12, 2022). *Using Coinbase APIs*. [Online]. Available: <https://help.coinbase.com/en/cloud/api/coinbase/using-apis>
 [12] E. Androulaki, G. O. Karame, M. Roeschlin, T. Scherer, and S. Capkun, “Evaluating user privacy in bitcoin,” in *Proc. Int. Conf. Financial Cryptogr. Data Secur.* Springer, 2013, pp. 34–51.
 [13] M. Spagnuolo, F. Maggi, and S. Zanero, “Bitiodine: Extracting intelligence from the bitcoin network,” in *Proc. Int. Conf. Financial Cryptogr. Data Secur.* Springer, 2014, pp. 457–468.
 [14] M. Fleder, M. S. Kester, and S. Pillai, “Bitcoin transaction graph analysis,” 2015, *arXiv:1502.01657*.
 [15] F. Reid and M. Harrigan, “An analysis of anonymity in the bitcoin system,” in *Security and Privacy in Social Networks*. Springer, 2013, pp. 197–223.

- [16] J. Herrera-Joancomartí, "Research and challenges on bitcoin anonymity," in *Data Privacy Management, Autonomous Spontaneous Security, and Security Assurance*. Springer, 2014, pp. 3–16.
- [17] M. C. K. Khalilov and A. Levi, "A survey on anonymity and privacy in bitcoin-like digital cash systems," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 3, pp. 2543–2585, 3rd Quart., 2018.
- [18] L. Morris, "Anonymity analysis of cryptocurrencies," Rochester Inst. Technol., Rochester, NY, USA, Tech. Rep., 2015.
- [19] J. Groth and M. Kohlweiss, "One-out-of-many proofs: Or how to leak a secret and spend a coin," in *Proc. 34th Annu. Int. Conf. Theory Appl. Cryptograph. Techn.*, Sofia, Bulgaria, Springer, Apr. 2015, pp. 253–280.
- [20] J. Jensen, "Federated identity management challenges," in *Proc. 7th Int. Conf. Availability, Rel. Secur.*, Aug. 2012, pp. 230–235.
- [21] E. Ghazizadeh, M. Zamani, J.-L. A. Manan, and A. Pashang, "A survey on security issues of federated identity in the cloud computing," in *Proc. 4th IEEE Int. Conf. Cloud Comput. Technol. Sci.*, Dec. 2012, pp. 532–565.
- [22] A. Bakre, N. Patil, and S. Gupta, "Implementing decentralized digital identity using blockchain," *Int. J. Eng. Technol. Sci. Res.*, vol. 4, no. 10, pp. 379–385, 2017.
- [23] M. M. El Gayyar, H. F. El Yamany, K. Grolinger, M. A. M. Capretz, and S. Mir, "Blockchain-based federated identity and auditing," *Int. J. Blockchains Cryptocurrencies*, vol. 1, no. 2, pp. 179–205, 2020.
- [24] Y. Liu, D. He, M. S. Obaidat, N. Kumar, M. K. Khan, and K.-K. R. Choo, "Blockchain-based identity management systems: A review," *J. Neww. Comput. Appl.*, vol. 166, Sep. 2020, Art. no. 102731.
- [25] J. Herbert and A. Litchfield, "A novel method for decentralised peer-to-peer software license validation using cryptocurrency blockchain technology," in *Proc. 38th Australas. Comput. Sci. Conf. (ACSC)*, vol. 27, 2015, p. 30.
- [26] A. Theodouli, K. Moschou, K. Votis, D. Tzouvaras, J. Lauinger, and S. Steinhorst, "Towards a blockchain-based identity and trust management framework for the IoT ecosystem," in *Proc. Global Internet Things Summit (GIoTS)*, Jun. 2020, pp. 1–6.
- [27] H. Xu, Z. Li, Z. Li, X. Zhang, Y. Sun, and L. Zhang, "Metaverse native communication: A blockchain and spectrum prospective," 2022, *arXiv:2203.08355*.
- [28] M. A. I. Mozumder, M. M. Sheeraz, A. Athar, S. Aich, and H.-C. Kim, "Overview: Technology roadmap of the future trend of metaverse based on IoT, blockchain, AI technique, and medical domain metaverse activity," in *Proc. 24th Int. Conf. Adv. Commun. Technol. (ICACT)*, Feb. 2022, pp. 256–261.
- [29] T. P. Pedersen, "Non-interactive and information-theoretic secure verifiable secret sharing," in *Proc. Annu. Int. Cryptol. Conf.* Springer, 1991, pp. 129–140.
- [30] I. Damgård, "Efficient concurrent zero-knowledge in the auxiliary string model," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.*, Bruges, Belgium, Springer, May 2000, pp. 418–430.
- [31] D. Reed, M. Sporny, D. Longley, C. Allen, R. Grant, and M. Sabadello. (Dec. 9, 2019). *Decentralized Identifiers—W3C Working Draft*. [Online]. Available: <https://www.w3.org/TR/did-core/>
- [32] Z. Galil, "Efficient algorithms for finding maximum matching in graphs," *ACM Comput. Surv.*, vol. 18, no. 1, pp. 23–38, Mar. 1986.
- [33] A. V. Goldberg and R. E. Tarjan, "A new approach to the maximum-flow problem," *J. ACM*, vol. 35, no. 4, pp. 921–940, Oct. 1988.
- [34] M. Mucha and P. Sankowski, "Maximum matchings via Gaussian elimination," in *Proc. 45th Annu. IEEE Symp. Found. Comput. Sci.*, Oct. 2004, pp. 248–255.
- [35] A. V. Goldberg and R. E. Tarjan, "Efficient maximum flow algorithms," *Commun. ACM*, vol. 57, no. 8, pp. 82–89, Aug. 2014.
- [36] O. Cruz-Mejía and A. N. Letchford, "A survey on exact algorithms for the maximum flow and minimum-cost flow problems," *Networks*, vol. 82, no. 2, pp. 167–176, Sep. 2023.
- [37] J. Groth, "On the size of pairing-based non-interactive arguments," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.* Springer, 2016, pp. 305–326.
- [38] M. Backes, N. Döttling, L. Hanzlik, K. Klucznik, and J. Schneider, "Ring signatures: Logarithmic-size, no setup-from standard assumptions," in *Proc. 38th Annu. Int. Conf. Theory Appl. Cryptograph. Techn.*, Darmstadt, Germany, Springer, May 2019, pp. 281–311.
- [39] B. Libert, T. Peters, and C. Qian, "Logarithmic-size ring signatures with tight security from the DDH assumption," in *Proc. 23rd Eur. Symp. Res. Comput. Secur. (ESORICS)*, Barcelona, Spain, Springer, Sep. 2018, pp. 288–308.
- [40] T. H. Yuen, M. F. Esgin, J. K. Liu, M. H. Au, and Z. Ding, "DualRing: Generic construction of ring signatures with efficient instantiations," in *Proc. 41st Annu. Int. Cryptol. Conf.* Springer, Aug. 2021, pp. 251–281.
- [41] H. Lin and M. Wang, "Repudiable ring signature: Stronger security and logarithmic-size," *Comput. Standards Interfaces*, vol. 80, Mar. 2022, Art. no. 103562.
- [42] A. Poelstra, A. Back, M. Friedenbach, G. Maxwell, and P. Wuille, "Confidential assets," in *Proc. Int. Conf. Financial Cryptogr. Data Secur.* Springer, 2018, pp. 43–63.
- [43] B. Bünz, J. Bootle, D. Boneh, A. Poelstra, P. Wuille, and G. Maxwell, "Bulletproofs: Efficient range proofs for confidential transactions," IEEE SP citation_publication_date=May 2018, Tech. Rep., 2017.
- [44] C. Allen, "The path to self-sovereign identity," Life With Alacrity, Tech. Rep., 2016. [Online]. Available: <http://www.lifewithalacrity.com/2016/04/the-path-to-self-sovereign-identity.html>
- [45] D. S. Baars, "Towards self-sovereign identity using blockchain technology," M.S. thesis, Univ. of Twente, Enschede, The Netherlands, Tech. Rep., 2016.
- [46] A. Tobin and D. Reed, "The inevitable rise of self-sovereign identity," *Sovrin Found.*, vol. 29, p. 18, Sep. 2016.
- [47] D. Boneh, X. Boyen, and H. Shacham, "Short group signatures," in *Proc. Annu. Int. Cryptol. Conf.* Springer, 2004, pp. 41–55.
- [48] D. Chaum, "Security without identification: Transaction systems to make big brother obsolete," *Commun. ACM*, vol. 28, no. 10, pp. 1030–1044, Oct. 1985.
- [49] J. Alupotha and X. Boyen, "Practical UC-secure zero-knowledge smart contracts," IACR Cryptol. ePrint Arch., Tech. Rep., May 2022.
- [50] T. Kuhrt. (Oct. 10, 2022). *HyperLedger Indy*. [Online]. Available: <https://wiki.hyperledger.org/display/indy>
- [51] ShoCard. (Jan. 18, 2023). *ShoCard*. [Online]. Available: <https://shocard.com/>
- [52] chainZy. (Jan. 18, 2023). *chainZy*. [Online]. Available: <https://www.chainzy.com/>
- [53] EverID. (Jan. 18, 2023). *EverID*. [Online]. Available: <https://everest.org/>
- [54] Lifeid. (Jan. 18, 2023). *Lifeid*. [Online]. Available: <https://lifeid.io/>
- [55] selfKey. (Jan. 18, 2023). *selfKey*. [Online]. Available: <https://selfkey.org/>
- [56] Civic. (Jan. 18, 2023). *Civic*. [Online]. Available: <https://civic.com/>
- [57] THEKey. (Jan. 18, 2023). *THEKey*. [Online]. Available: <https://www.pickacrypto.com/coins/thekey-tyk/>
- [58] K. Gilani, E. Bertin, J. Hatin, and N. Crespi, "A survey on blockchain-based identity management and decentralized privacy for personal data," in *Proc. 2nd Conf. Blockchain Res. Appl. Innov. Netw. Services (BRAINS)*, Sep. 2020, pp. 97–101.
- [59] ID2020. (Oct. 10, 2022). *Accelerating Good Digital ID*. [Online]. Available: <https://id2020.org/projects>
- [60] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum Project Yellow Paper*, vol. 151, pp. 1–32, Apr. 2014.
- [61] Y. Fu, J. Shao, Q. Huang, Q. Zhou, H. Feng, X. Jia, R. Wang, and W. Feng, "Non-transferable blockchain-based identity authentication," *Peer-Peer Netw. Appl.*, vol. 16, no. 3, pp. 1354–1364, May 2023.
- [62] M. Shuaib, N. H. Hassan, S. Usman, S. Alam, S. Bhatia, P. Agarwal, and S. M. Idrees, "Land registry framework based on self-sovereign identity (SSI) for environmental sustainability," *Sustainability*, vol. 14, no. 9, p. 5400, Apr. 2022.



JAYAMINE ALUPOTHA received the Ph.D. degree in computer science from the Queensland University of Technology, Australia. She is currently a Postdoctoral Researcher with the University of Bern, Switzerland. Her research interests include privacy-preserving cryptography, mainly blockchain, and post-quantum cryptography.

...