

RESEARCH ARTICLE

A Self-Sovereign Identity Based on Zero-Knowledge Proof and Blockchain

MOHAMEDEN DIEYE¹, PIERRE VALIORGUE¹, JEAN-PATRICK GELAS², EL-HACEN DIALLO²,
PARISA GHODOUS², FRÉDÉRIQUE BIENNIER³, AND ÉRIC PEYROL¹

¹Univ. Lyon, UCBL, 69622 Villeurbanne, France

²Univ. Lyon, UCBL, CNRS, INSA Lyon, LIRIS, UMR5205, 69622 Villeurbanne, France

³Univ. Lyon, INSA Lyon, CNRS, UCBL, LIRIS, UMR5205, 69621 Villeurbanne, France

Corresponding author: Jean-Patrick Gelas (jean-patrick.gelas@univ-lyon1.fr)


This work was supported in part by INCLUDE (National Digital Demonstrator on Learner Inclusion in Higher Education), and in part by the Agence Nationale de la Recherche (ANR) contributed to the creation of this initiative under Grant ANR-21-DMES-0005.

ABSTRACT Systems for generating and managing digital identities are in the process of being transformed to improve data sharing security and increase decentralization. Addressing both issues, a theoretical solution to create and manage Self-Sovereign Identities (SSI) is proposed using two Zero-Knowledge Proof (ZKP) protocols based on the discrete logarithm difficulty. Automorphism group properties are introduced to link several identities, their identifiers and attributes to produce a proof. The proposed SSI protocol does not encounter the problem of reusing the same secret key as in the case of the initial ZKP Schnorr protocol. The designed protocol ensures minimal disclosure of information to a single trusted third party. In addition, it allows zero disclosure of information to service providers requiring proof of authentication or identification. Such a SSI protocol is compliant with Electronic IDentification And Trust Services (eIDAS) as well as General Data Protection Regulation (GDPR) regulations.

INDEX TERMS Decentralized identity (DID), self-sovereign identity (SSI), zero-knowledge proof (ZKP), higher education.

NOMENCLATURE

Γ	Bijective application $\mathbb{G} \rightarrow \mathbb{G}$.
Λ	Non injective morphism of $\mathbb{G} \rightarrow \mathbb{G}$.
H	Known hash function.
H_i	Automorphism sequence of \mathbb{G} .
H_S	Known hash function given by the State.
LH_i	Composition of H_i ; automorphism of \mathbb{G} .
\mathbb{G}	A finite cyclic group of order q .
\mathbb{Z}	Set of integers.
$\mathbb{Z}_q^{(*,+)}$	Finite field of order q .
\mathbb{Z}_q^*	A finite cyclic group of order $q - 1$.
$Aut(\mathbb{G})$	The set of automorphisms of \mathbb{G} .
c'	$\in \mathbb{Z}_q^*$, challenge.
c	$\in \mathbb{Z}_q^*$, challenge.
g	Group generator.

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

g^x	g raised to the power of x .
g_i	Sequence of group generator.
t	A known element $\in \mathbb{G}$.
x_i	Sequence of product $(a; b)$.
$-a$	Inverse of a , such $q - a = -a$ in $\mathbb{Z}_q^{(*,+)}$.
$-b$	Inverse of b , such $q - b = -b$ in $\mathbb{Z}_q^{(*,+)}$.
2^{-1}	Inverse of 2, such $2 * 2^{-1} = 1$ in \mathbb{Z}_q^* .
σ_i	$\in \mathbb{Z}_q^*$, signature of the prover.
a	A secret element $\in \mathbb{Z}_q^*$.
a_i	A secret sequences $\in \mathbb{Z}_q^*$.
b	A secret element $\in \mathbb{Z}_q^*$.
b_i	A secret sequences $\in \mathbb{Z}_q^*$.
c'	Constraint on m, k and o , $\in \mathbb{Z}_q^*$.
c_i	Sequence of hashed value of $(g, LH_i, m_i u^{-1} + (k_i \cdot o_i)^{u^{-1}})$, simulated challenge.
e	A secret element $\in \mathbb{Z}_q^*$.
f	A secret element $\in \mathbb{Z}_q^*$.
k	$\in \mathbb{G}$, response of the prover.

k_i	$\in \mathbb{G}$, response of the prover.
m	$\in \mathbb{G}$, response of the prover.
m_i	$\in \mathbb{G}$, response of the prover.
n	A secret element $\in \mathbb{Z}_q^*$.
o	$\in \mathbb{G}$, response of the prover.
o_i	$\in \mathbb{G}$, response of the prover.
p_i	A secret sequence element $\in \mathbb{Z}_q^*$.
Q	A known element $\in \mathbb{G}$.
q	Order of the group \mathbb{G} .
r	$\in \mathbb{Z}_q^*$, response of the prover.
s	A secret element $\in \mathbb{Z}_q^*$.
t^{-1}	Inverse of t , such $t \cdot t^{-1} = 1 \in \mathbb{G}$.
u	$\in \mathbb{Z}_q^*$.
w	Integer such as $0 < w < q$, number of proofs (m, k, o) .
x	A secret element $\in \mathbb{Z}_q^*$.
y'	A known element $\in \mathbb{G}$.
Y	A known element $\in \mathbb{G}$.
y	A known element $\in \mathbb{G}$.
Z''	Constraint on m, k and o , $\in \mathbb{Z}_q^*$.
Z'	Constraint on m, k and o , $\in \mathbb{Z}_q^*$.
z'	Sum of a and $b \in \mathbb{Z}_q^*$.
Z	Product of m, k and $o \in \mathbb{Z}_q^*$.
z	Produce of a and $b \in \mathbb{Z}_q^*$.

I. INTRODUCTION

In an increasingly connected world, the management and control of users' digital identities are becoming crucial elements of our numerical lives [1] where we are more and more led to share sensitive information. To address this issue, the Electronic Identification and Trust Services (eIDAS) has established standards. A service provider offering an identification (or authentication) system must comply with these standards to be certified [2]. However, eIDAS compliant service providers for identity such as the ones proposed by GAFAM [3] require the user to transfer the rights of sensitive data in order to benefit from their services.

As an alternative identity service, a self sovereign identity (SSI) would improve user's experience by giving them power and control over their digital identifiers [4], [5]. Such an SSI could help address the metaverses' need for scalable interoperability and security highlighted in [6]. Self sovereign identity systems have been inspired by the developments of digital identity management systems for distributed architectures [4], [5]. By relying on the properties of distributed architectures, such as public distributed blockchains, providing immutability on all information written in the blockchain, the majority of SSI implemented with a blockchain comply with C.Alen's 11 criteria defining self-sovereign identities [3], [7], [8]. However, a trusted third party certifying the identity is still required in order to uniquely identify a person [5].

Following European Union's General Data Protection Regulation principles, data related to a person on the blockchain must remain absolutely confidential. By default data on a blockchain is plain text [9]. Then, data must be encrypted [10]

and the amount of data transferred while exchanging information should be minimized. The two technologies minimizing the amount of data transferred while exchanging information are selective disclosure of identity attributes and verifiable claims [5]. Going beyond these technologies, Zero-Knowledge Proof (ZKP) enables writing only a proof of knowledge on the blockchain instead of the information itself [11].

The challenge is to implement a decentralized SSI using blockchain and ZKP technology allowing access to a service without disclosing sensitive information.

The remainder of the paper is organized as follows. Section II presents a literature review of works addressing this challenge. Section III summarizes the main contributions of this work. In section IV, an original SSI solution is detailed with an emphasis on its compliance with eIDAS, GDPR as well as criteria introduced by C. Allen and complemented by Q. Stokkink and J. Pouwelse with provability [3], [12]. Section V illustrates the proposed solution with an SSI-Protocol example. In section VI, a qualitative comparison and an evaluation based on simulations for the introduced SSI-ZKP_{a-b} is proposed. Finally, section VII concludes the paper.

II. STATE OF THE ART

Several SSI solutions have been implemented with blockchain [3], [4], [5]. Such SSI blockchain implementations can be further improved adding ZKP layers [4], [5], bearing in mind that different ZKP protocol implementations will have different properties such as their ZKP size, proof size, time duration for creation and verification [13], [14], [15].

In practice, as stated in the report for the French ministry of interior on Blockchain and digital identification working group [5], most SSI implementation solutions provide three common tools, namely:

- **Cryptographic and hash functions.** The user recovers or creates a unique digital identity, generates his decentralized identifiers and links his certified documents or attributes to it (by a trusted third party, another user or a self-certificate).
- **Decentralized Identifiers (DID).** The user uses his decentralized identifier once connected to access a good or a service. He declares to be "who he is", and proves that he has the attributes required to access service through verifiable assertions relating to attributes of his identity and/or authentic documents.
- **Verifiable assertions.** The verifier checks the certificates and proofs provided before granting access to the service.

A SSI solution can be tailored for specific purposes. Different organizations have made proposals addressing different civil, economic and social problems. For example, the Secure Identity Alliance (SIA) consortium proposed interfaces (API) in Open Standard for the interoperability of operations on civil, identity and administrative registers. The ID2020 Alliance

aimed to fund the technology development of secure digital certificates and establish standards to facilitate interoperability and multi-party collaboration [16]. Currently the French company Archipel offers an identity management solution based on the Blockchain which tends towards an SSI compliant with eIDAS and GDPR.

SSIs can be evaluated with the following C.Allen [3], [12] criteria, which gives an overview of their performance and level of security:

- 1) **Existence:** users must have independence and autonomy in the creation of their digital identity. No collision of identifiers, which must be attached to a unique identity in real life.
- 2) **Control:** the user must have control over the creation and management of their identities.
- 3) **Access:** users should have full access to their own data without any restrictions.
- 4) **Transparency:** the proposed SSI implementation modalities and algorithms must be shared, open-source, well-known, and as independent as possible
- 5) **Persistence:** the identity must have a durable lifespan, and any update related to this identity must be made only by the user.
- 6) **Portability:** the storage and durability of the identity must not rely entirely on a trusted third party, at the risk that its disappearance implies the disappearance of the identity.
- 7) **Interoperability:** The number of niche does not impact the proper functioning of the proposed protocol.
- 8) **Consent:** any manipulation of the identity or claim must be preceded by a consent of the user attributed to this identity.
- 9) **Minimization:** only information relevant and required to serve claims is disclosed and shared.
- 10) **Protection:** all the legitimate rights of users must be preserved and respected.
- 11) **Provable:** claims must be verifiable.

Existing SSI blockchain implementations can be further improved toward better checking of the above mentioned criteria by adding ZKP layers [3], [4], [5], [7], [8]. Zero-Knowledge Proof protocols pave the way toward technologies embedding privacy-first principles for data transmission. The user is given the possibility to remain anonymous while being able to prove he possesses his verifiable credential (anonymous credentials) [17].

Different ZKP protocols are based on different cryptographic assumptions. Different ZKP protocols are also designed or implemented for different use cases with ad-hoc computational problems. ZKP differences induce limited comparisons between them due to heterogeneous design parameters. Among these parameters one could note the size of proofs as well as exchange protocols for generation and verification keys [13], [14], [15], [18].

Two classes of protocols are distinguished, Interactive ZKP (IZKP) and Non-Interactive ZKP (NIZKP), depending on

whether the protocol requires (IZKP), or does not require (NIZKP), the simultaneous connection of the prover and the verifier(s). NIZKP protocols have a strong user experience design advantage. The three non-interactive ZKP protocols which have been reported to remain unbroken at the time of writing are zk-SNARK, BulletProofs and zk-STARK [18], [19].

Different SSI protocols fulfilling each and every one of the above listed criteria have been published. For example, Sora identity [1] has been proposed as a secure digital identity based on the Blockchain. Sora presents itself as a mobile application that allows the creation of verifiable credentials while guaranteeing pseudonymity.

Another SSI protocol example, fulfilling the previously enumerated criteria, is SelfKey [20]. SelfKey is proposed as a blockchain-based self-sovereign digital identity network using ZKP. In SelfKey, users can completely control and reveal their identity to a third party. ZKP is used to minimize shared information while approving third-party access to the revealed data. LifeID [21] is similarly proposed as a self-sovereign digital identity platform that allows users to create independent online identities. LifeID uses ZKP to ensure only necessary information is revealed during identity verification.

Although the above-cited SSI protocols claim using ZKP techniques, they still reveal information to a third party. Using existing ZKP techniques has limitations that could be overcome by designing ZK-native SSI based on a tailored NIZKP protocol.

The challenge addressed in this article is then to design a non-interactive SSI data exchange protocol using blockchain technology respecting the above mentioned criteria and inherently ensuring zero information disclosure (NIZKP) while communicating sensitive information.

III. CONTRIBUTIONS

The main contributions of this work can be summarized as follows:

- An original Self-Sovereign Identity solution embedding Zero Knowledge Proof based on blockchain technologies is proposed.
- A ZKP protocol enabling to prove that a and b such that $a \cdot b = n$ are known without revealing a and b has been proposed.
- Automorphism sequence linking different identities, identifiers and/or attributes as well as different blockchain information has been designed linking the ZKP Schnorr and an original ZKP protocol named ZKP_{a-b} .

IV. METHODS

Would a theoretical implementation for a privacy-first SSI solution based on ZKP and blockchain be able to carry out the objective of giving to the user the ability to manage his digital identity and associated wallet? Depending on what a user needs to authenticate, identify, justify or prove, such an

implementation would enable the user to select and share a proof of his identity attributes without disclosing any other sensitive information.

The proposed SSI solution corresponds to the challenges, demands and objectives raised in [5] *i.e.* selective disclosure of attributes/identifiers, confidentiality and security. In order to implement the envisaged SSI solution, a decentralized protocol scheme, cryptographic techniques based on the structure of algebraic groups and blockchain technology have been assembled.

Two ZKPs have been employed, a ZKP Schnorr (IV-A) and an original one (IV-B) tailored to meet C. Allen criteria when exploited on a blockchain. In order to reduce the risk of usurpation [22], a trusted third party referred to as “The State” has been introduced. The State will allow to have a basic identity created by the user, which can guarantee its uniqueness and veracity. Starting from this State approved digital identity, other digital identities can be built. In the proposed SSI, identities and its attributes can be gathered and linked to produce a single proof.

A. SCHNORR’S ZERO KNOWLEDGE-PROOF

A ZKP Schnorr transformed with the Fiat-Shamir method [23] has been chosen for its straightforward cryptographic architecture, its compatibility with the original ZKP protocol and its robustness induced by the discrete logarithm problem. This protocol has the advantage of being generalized [24] with homomorphisms instead of the automorphism which forms $g \rightarrow g^w$, g being a group generator and w an integer such as $0 < w < q$, q being the order of the group.

Let’s consider the following problem: Alice wishes to prove to Bob that she knows an element x , without disclosing it, using a Schnorr ZKP. To do so, let q be a prime number and \mathbb{G} a group generated by g of order q .

Alice wishes to prove Bob that she knows an element x such that $y = g^x$. Let’s also define the following steps of a Schnorr ZKP protocol:

- Creation step of the interactive Schnorr environment where Alice generates the proof to be sent to Bob:
 - Commitment: Alice randomly draws an integer s from the set \mathbb{Z}_q^* and keeps it secret. Alice will then calculate $t = g^s$ and send it to Bob.
 - Challenge: Bob randomly draws an integer $c \in \mathbb{Z}_q^*$ and sends it to Alice.
 - Answer: Alice calculates $r = s - c \cdot x$ and sends it to Bob.
- Verification step where Bob validates the received answer r from Alice:

Using $r = s - c \cdot x$ and $y = g^x$, Bob verifies that Alice knows and possesses the secret x if $g^r \cdot y^c$ corresponds to t as in equation (1).

$$g^{s-c \cdot x} \cdot g^{c \cdot x} = g^s = t \tag{1}$$

Adding the Fiat-Shamir transformations to the Schnorr protocol, the signature scheme becomes non-interactive using a hash function. The creation and verification steps of the Schnorr ZKP protocol are then modified as follows:

- Creation step where Alice still wants to prove that she knows x such that $y = g^x$ with $y \in G$ a group generated by g of order q (g and q being public):
 - Keys generation: Alice will randomly draw an integer $s \in \mathbb{Z}_q^*$ and calculate $t = g^s$. s is Alice’s private key and t is Alice’s public key.
 - Challenge and answer: Alice computes $c = H(g, y, t)$ where H is a public hash function and calculates $r = s - c \cdot x$. Alice sends to Bob the tuple (c, r) which corresponds to Alice’s signature.
- Verification step: Bob or anyone having access to the public variables (g, q, t, H, c, r) can verify that t is equal to $g^r \cdot y^c$, proving that Alice knows and possesses the secret s . Bob will authenticate that the secret belongs to Alice when receiving Alice’s signature (c, r) . The Fiat-Shamir transformation therefore removes the need of interaction between the provider of a proof and the verifier.

Both the interactive and non-interactive protocols satisfy all three ZKP fundamental properties:

- Consistency: if anyone using the proposed protocol and verifying the relation t is equal to $g^r \cdot y^c$, he can be certain that another claiming to know x (like Alice) is true.
- Robustness: First, it is not possible for Alice to generate a proof for Bob without possessing her private key (required to produce s). Second, it is not possible for Alice to generate a proof for Bob without knowing the secret x .
- Zero-knowledge disclosure: Alice’s secret x is protected as it is not possible for Bob to calculate or access Alice’s private key from the public data.

This protocol will be used in (IV-C) alongside an original protocol described in the following section.

B. ZKP_(a-b): CONTEXT AND PROTOCOL DEFINITION

1) 1- CONTEXT

The purpose of the proposed ZKP_(a-b) protocol is for someone to demonstrate that he knows and possesses two secrets, the numbers a and b . Let z be the product of a and b :

$$a \cdot b = z \tag{2}$$

If a and b are not prime numbers, several pairs satisfying this challenge can be found. As a trivial example, for $a \cdot b = 12$, three pairs of possible positive solutions are (2; 6), (3; 4), (1; 12). Let’s add a constraint on a and b that guarantees uniqueness for $(a;b)$, and let it be:

$$a + b = z' \tag{3}$$

Let's introduce the integers e and f as solutions of equations (2,3):

$$\begin{cases} a \cdot b = e \cdot f = z & (4) \\ a + b = e + f = z' & (5) \end{cases}$$

As deductions from the constraint in equations (4) and (5):

$$\begin{aligned} a + b = e + f \text{ thus } (a + b)^2 &= (e + f)^2 \\ &= a^2 + b^2 + 2 \cdot a \cdot b \\ &= e^2 + f^2 + 2 \cdot e \cdot f \end{aligned} \quad (6)$$

$$a^2 + b^2 = e^2 + f^2 \quad (7)$$

Considering equation (4):

$$\begin{aligned} a^2 - f^2 + b^2 - e^2 &= (a + f)(a - f) - (b + e)(e - b) \\ &= (a - f)[a + f - b - e] \\ &= 0 \end{aligned} \quad (8)$$

Two solution are obtained:

$$\begin{cases} a = f \text{ or} & (9) \\ a + f = b + e & (10) \end{cases}$$

Again combining equation (5) with equation (10):

$$\begin{aligned} a + f - (a + b) &= b + e - (e + f) \\ f - b &= b - f \\ b &= f \end{aligned} \quad (11)$$

Consequently, a unique couple is obtained from equations (2,3) since equations (9,11) give:

$$\begin{cases} a = f \text{ and } b = e \\ \text{or} \\ a = e \text{ and } b = f \end{cases} \quad (12)$$

Since a and b are the secrets that someone wants to prove he knows, let's introduce m, k and o as images of a and b in order to be able to exchange information without disclosing a and b . In that spirit, let m be an image of a linear combination of a and b , let k be an image of a and let o be an image of b . Similarly as done with equation (2) for a and b , let Z be the product of the images m, k and o :

$$m \cdot k \cdot o = Z, Z \in \mathbb{Z}_q^* \quad (13)$$

Let's add the following constraints for the triplet $(m; k; o)$:

$$\begin{cases} m + k \cdot o = Z', Z' \in \mathbb{Z}_q^* & (14) \\ k + o = Z'', Z'' \in \mathbb{Z}_q^* & (15) \end{cases}$$

Z, Z' and Z'' will be defined as known variables in the protocol. Let's define the images m, k and o as follows:

$$m = g^{(a+b)^2} \quad (16)$$

$$k = g^{-a^2} \quad (17)$$

$$o = g^{-b^2} \quad (18)$$

Using equations (13, 14, 15), a triplet $(m; k; o)$ is the image of a couples of two secrets, the couple $(a; b) \in \mathbb{Z}_q^{(*,+)} \times \mathbb{Z}_q^{(*,+)}$

and its inverse. Let's demonstrate this affirmation starting by defining the bijective application Γ as well as the function Λ :

$$\Gamma: \mathbb{G} \rightarrow \mathbb{G}, x \rightarrow y = g^x \quad (19)$$

$$\Lambda: \mathbb{G} \rightarrow \mathbb{G}, x \rightarrow y = x^2 = (-x)^2 \quad (20)$$

Let's calculate $g^{a \cdot b}$ for later use in the protocol. First, using the remarkable identity eq. (21), $g^{2 \cdot a \cdot b}$ is obtained in eq. (22).

$$\begin{aligned} (a + b)^2 &= a^2 + b^2 + 2ab \\ \iff 2 \cdot a \cdot b &= (a + b)^2 - a^2 - b^2 \end{aligned} \quad (21)$$

$$g^{2 \cdot a \cdot b} = g^{(a+b)^2 - a^2 - b^2} \quad (22)$$

Secondly, choosing $u = 2^{-1} \in \mathbb{Z}_q^*$ and using equations (16,17,18) in equation (22), $(m \cdot k \cdot o)$ and $g^{a \cdot b}$ are related by the following equation:

$$(m \cdot k \cdot o)^u = g^{a \cdot b} \quad (23)$$

Finally, applying the constraints from eq.(13,14,15,19,20) on the triplet $(m; o; k)$, the unique solutions of eq. (23) are the couples $(a; b) \in \mathbb{Z}_q^{(*,+)} \times \mathbb{Z}_q^{(*,+)}$ and $(-a; -b) = (q-a; q-b) \in \mathbb{Z}_q^{(*,+)} \times \mathbb{Z}_q^{(*,+)}$.

2) 2-ZKP_(a,b) PROTOCOL

Based on the above background, let us now define the ZKP_(a,b) protocol itself and start setting up the following variables:

$$g^{a \cdot b} = \Gamma(a \cdot b) \quad (24)$$

$$t = g^s, \text{ with } s \text{ a secretly and randomly drawn integer} \quad (25)$$

$$m = \Gamma(s + u \cdot \Lambda(a + b)) \quad (26)$$

$$k = \Gamma(-\Lambda(a) \cdot u) \quad (27)$$

$$o = \Gamma(-\Lambda(b) \cdot u) \quad (28)$$

$$c' = H(g, y, (t^{-1} \cdot m)^{u^{-1}} + (k \cdot o)^{u^{-1}}), \quad (29)$$

$$H \text{ a hash function and } y \text{ a given number.} \quad (30)$$

$$u = c'/2, \text{ if } c' \text{ is even or } u = (c' + 1)/2 \text{ otherwise} \quad (31)$$

$$c = 2u, c \text{ being the challenge} \quad (31)$$

In order for one person, let her be Alice, to demonstrate to another person, let him be Bob, that she knows and possesses two secrets (the numbers a and b) using the protocol ZKP_(a,b), the below successive steps should be followed:

- Proof generation
 - Objective: Alice wants to prove that she knows a and b both in the set \mathbb{Z}_q^* such that $a \cdot b = z$.
 - Generation of keys: Alice will randomly draw an integer s and calculate $t = g^s$. Let s the secret and Alice's private key. Alice publishes t , her public key.
 - Generation of the challenge and of the answer to the proof to be verified: Alice computes $c' = H(g, y, (t^{-1} \cdot m)^{u^{-1}} + (k \cdot o)^{u^{-1}})$, H being a public hash function. Alice also calculates the

set $r = (m, k, o)$. The couple (c, r) will represent Alice's signature. Alice sends to Bob (c, r) . Let's note that Alice cannot only send the variable r since Bob will not be able to reconstruct c' from public variables and r .

- Alice publishes $\sigma_r = k + o$, the constraint from equation. (15).

• Response verification

- Bob calculates $\sigma_v = k + o$ from the received set r and compares σ_v to the published Z'' .
- Bob will then verifies with r that the product $m \cdot k \cdot o$ corresponds to $t \cdot g^{c \cdot a \cdot b}$ which Bob will calculates from public variables t, c, g and the z (the product of a and b).

The $ZKP_{(a,b)}$ protocol verifies the robustness, consistence and zero knowledge properties of a ZKP protocol:

- consistency: if anyone using the proposed protocol and verifying the relation $t \cdot g^{a \cdot b \cdot c}$ is equal to $m \cdot k \cdot o$ and $k + o$ is equal to σ_r , he can be certain that an another person claiming to know (a, b) is sincere.
- robustness: First, it is not possible for Alice to generate a proof for Bob without knowing two elements from the list $[a^2, b^2, (a + b)^2]$. Second, it is not possible for Alice to calculate c without knowing u which is required to calculate c' .
- zero-knowledge disclosure: Alice's secret x is protected as it is not possible for Bob to calculate or access Alice's private key from public data.

The $ZKP_{(a,b)}$ protocol ensures that it is impossible for Bob, knowing $r = (m, k, o)$ and c , to retrieve Alice's secrets a and b with currently available algorithms since:

- Bob cannot find the secret resolving $y = g^{x^2}$, indeed
 - finding x^2 has the complexity of the discrete logarithm problem [25].
 - finding x the modular root of x^2 has non-polynomial complexity. [26]
- Bob cannot find the secret a and b resolving $y = g^{a \cdot b}$, indeed
 - finding $a \cdot b$ has the complexity of the discrete logarithm problem.
 - finding a and b knowing $z = a \cdot b$ is difficult [27].

C. PROOF OF KNOWLEDGE AND AUTOMORPHISMS

1) 1- CONTEXT

Automorphisms properties can be exploited to create a chain of (proofs, verifications) containing several assertions.

To this aim, let's first define a group G generated by g of order q , q being a prime number:

$$\forall y \in G, \exists i, 0 \leq i \leq q - 1 \text{ such that } y = g^i \quad (32)$$

Let $Aut(G)$ be the set of automorphisms of G . From group theory, $Aut(G)$ will:

- be isomorphic to Z_q^* .
- have, if $\zeta \in Aut(G)$, $\zeta(g)$ also a generator of G .

- have exactly $q - 1$ generators, G is of order q .
 $Aut(G)$ having $q - 1$ generators, $q - 1$ assertions can be linked together using H_i and g_i defined as follows:

$$\mathbb{G} \rightarrow \mathbb{G} \quad g \rightarrow H_i(g) = g_i = g^{p_i} \quad \text{with } 0 \leq i \leq q - 1 \text{ and } 0 \leq p_i \leq q - 1 \quad (33)$$

Consequently, a successive composition of w ($1 \leq i \leq w$) automorphisms H_i can be written as:

$$LH_w(g) = H_w \circ H_{w-1} \circ \dots \circ H_1(g) = g^{p_1 \cdot p_2 \dots p_w} \quad (34)$$

Assuming $y = g^n$, equation (34) becomes:

$$LH_w(y) = g^{n \cdot p_1 \cdot p_2 \dots p_w} \quad (35)$$

With $0 \leq i \leq w$, $a = p_1 \cdot p_2 \cdot \dots \cdot p_w \pmod q$ and $b = n$, equations (33,35) become:

$$LH_w(y) = g^{a \cdot b}, \exists i \in [1, q] \text{ such that, } g^b = g_i. \quad (36)$$

To put it in a nutshell, automorphisms offer the possibility to use equation (36) similarly as in:

- a ZKP Schnorr knowing one secret a :

$$(36)LH_w = LH_w(g) = g_i^a \quad (37)$$

- $ZKP_{a,b}$ knowing two secrets a and b :

$$(36)LH_w^n = g^{a \cdot b} = LH_w(y) \quad (38)$$

2) 2- MULTI-STEP PROOF OF KNOWLEDGE

Combining a ZKP Schnorr, the proposed $ZKP_{a,b}$ and making use of automorphisms, let's build a multi-step proof of knowledge enabling to prove a set of w secrets p_i and another secret n (which can be delivered for example by a trusted third party).

Let's define the following sequence x_i based on the secrets p_i and initialised with n :

$$x_i = x_{i-1} \cdot p_i, \quad x_0 = n, \quad LH_i(y) = g^{x_i} \quad (39)$$

Using the $ZKP_{a,b}$ protocol when proving each secrets p_i and n , a and b will be identified as:

$$a_i = x_{i-1} \text{ and } b_i = p_i. \quad (40)$$

However, it is not possible for a_i and b_i to remain secrets while using the $ZKP_{a_i \cdot b_i}$ protocol successively to prove each secrets p_i and n . Indeed, let's consider the following secrets verification for n, p_1 and p_2 :

$$\begin{aligned} \text{Step (1) - } ZKP_{a_1 \cdot b_1} : x_1 &= a_1 \cdot b_1 = n \cdot p_1 \\ \text{Step (2) - } ZKP_{a_2 \cdot b_2} : x_2 &= a_2 \cdot b_2 = n \cdot p_1 \cdot p_2 \\ &\implies p_2 = x_2 / x_1 \end{aligned} \quad (41)$$

From equation (41) for successive steps $ZKP_{a_i \cdot b_i}$, a verifier will be able to guess p_2 (using x_2 which is public), invalidating the Zero Knowledge Protocol.

In order to resolve this issue and keep p_i as secrets during verification, let's redefine the second step in equation (41):

$$\text{Step (1) - } ZKP_{a_1 \cdot b_1} : x_1 = n \cdot p_1$$

$$\begin{aligned} \text{Step (2) - ZKP}_{a_3, b_3} : x_3 &= x_2 \cdot p_3 \\ \implies p_3 &= x_3/x_2 \end{aligned} \quad (42)$$

Using the procedure in equation (42) where x_2 is kept as a secret, a verifier cannot find the secret p_3 .

Considering the following ZKP verification steps:

$$\begin{aligned} \text{Step (1) - ZKP}_{a_1, b_1} : a_1 &= x_0; b_1 = p_1 \\ \text{Step (2) - ZKP}_{a_2, b_2} : a_2 &= x_1; b_2 = p_2 \quad \text{with } x_1 = x_0 \cdot p_1 \\ \text{Step (3) - ZKP}_{a_3, b_3} : a_3 &= x_2; b_3 = p_3 \quad \text{with } x_2 = x_1 \cdot p_2 \end{aligned} \quad (43)$$

From equations (41, 42), verifying Step ($i - 1$) and Step ($i + 1$) to prove knowing the secrets ($x_{i-2}; p_{i-1}$) and ($x_i; p_{i+1}$), is enough to avoid a ZKP_{a_i, b_i} verification at Step (i) to prove the secret p_i . Indeed, looking at equations (43), after ZKP_{a_1, b_1} and ZKP_{a_3, b_3} verification for Steps (1) and (3), ZKP_{a_2, b_2} verification for Step (2) will be already completed since:

$$\begin{aligned} \text{Step (1)} &\implies (x_0; p_1) \text{ proved: } a_2 = x_1 = x_0 \cdot p_1 \\ \text{Step (3)} &\implies x_2 \text{ proved: } b_2 = p_2 = x_2/x_1 \end{aligned} \quad (44)$$

As a consequence, after verifying $\text{ZKP}_{a_{i-1}, b_{i-1}}$ for p_{i-1} , ZKP_{a_i, b_i} verification for p_i can be skipped by verifying $\text{ZKP}_{a_{i+1}, b_{i+1}}$ for p_{i+1} which will also provide a proof for p_i .

Assuming $x_i = x_{i-1} \cdot p_i$, $x_0 = n$ and $LH_i(y) = g^{x_i}$, with $a_i = x_{i-1}$ and $b_i = p_i$ for each step, the verification step for ZKP_{a-b} becomes:

$$m_i \cdot k_i \cdot o_i = t \cdot LH_i(y)^c \quad (45)$$

Let's note that a procedure could disclose LH_i for the verification step instead of $z = a \cdot b$. Such a disclosure would avoid the successive ZKP_{a-b} problem shown in equation (41). An example for the proposed SSI using a ZKP Schnorr and a ZKP_{a-b} is presented in the next section.

V. ILLUSTRATIVE EXAMPLE OF THE SSI-PROTOCOL

A. SCENARIO

Alice wishes to present proof of existence of a set of documents without revealing their content to Bob. As an illustration for a SSI-Protocol, a first document of the set of documents would be a proof of identity. One could think of numerous applications for proofs of existence of documents to be linked to an identity such as for example a proof of social security or a proof of solvency with a bank. In the application described in [28] motivating this research, a student would link a proofs of existence for his student identity, working permit or social security number to proofs he would have aggregated in his wallet for his university degrees or verifiable credentials certifying his skills.

For such applications linking a proof of existence of an identity to proofs of existence of other documents, two steps are required. In the first step presented in the next section, a protocol generating a Self-Sovereign Identity using a ZKP Schnorr is proposed. In the following section, the introduced ZKP_{a-b} protocol is exploited to link proofs of existence of other documents to the generated Self-Sovereign Identity.

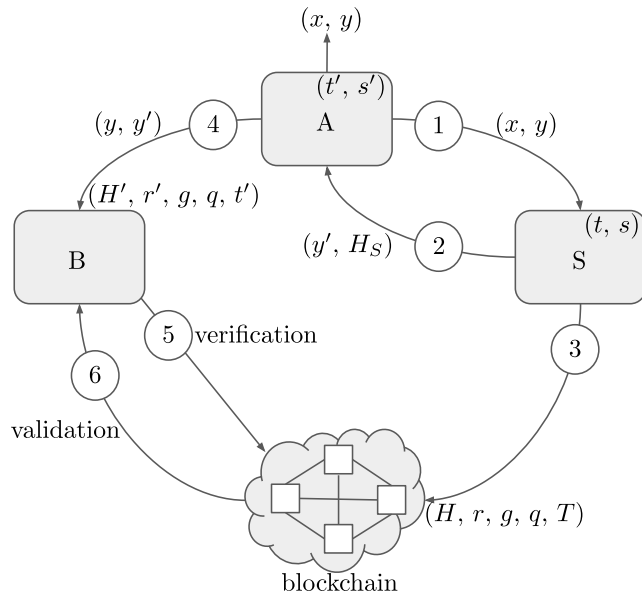


FIGURE 1. SSI generation protocol using a ZKP Schnorr: generation (1,2), registration (3) and verification (4,5,6) of a Self-Sovereign Identity.

B. INITIALISATION STEP

In order for Alice (A) to generate a Self-Sovereign Identity, register it on a blockchain through the State (S) and for Bob (B) to verify Alice's identity, the following steps (illustrated by figure (1)) are proposed:

- generation of a unique and robust couple (x, y) representing Alice's identity:
 - A chooses an x that she will keep as a secret in the group \mathbb{Z}_q^* and calculates $y = g^x$ using g a generator of the group \mathbb{G} of order q , q being a prime number. It is assumed here that g and q have been already defined, that they are public variables and therefore known by all protocol stakeholders.
 - A sends the couple (x, y) to S. It is assumed that any communication in the proposed protocol is encrypted.
 - S checks if the received x has not already been taken and that it is difficult to find x knowing y and g . S should ensure that both sizes of x and y must respect the minimum size to resist attacks and the maximum size to be supported by the verification algorithms.
 - S then sends back the couple (y', H_S) to A as confirmation, H_S being the hash function used to calculate $y' = g^{H_S(x)}$.
- registration of information related to Alice's generated by the state identity on the blockchain:
 - using a ZKP Schnorr, S generates H and r which are information related to Alice's generated identity as well as t , the state public key, and T which combines information from Alice's public key, the state public key and y' (an image of $H_S(x)$).
 - through a smart contract and signing with t , S registers (H, r, g, q, T) on a blockchain.

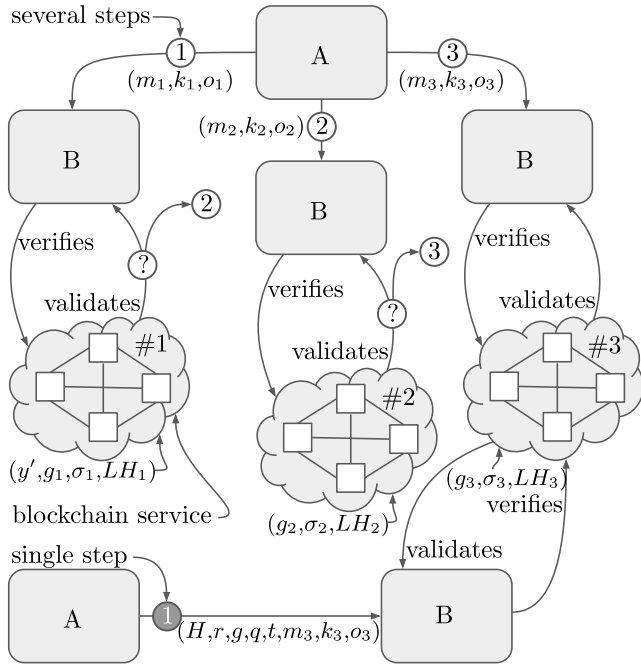


FIGURE 2. Different blockchain services from a single blockchain or multiple blockchains, Derived Step: Top (several steps); Bottom (single step).

- verification of Alice’s claim by Bob
 - A proves to B that she knows x by providing B with the couple $(y; y')$, related to the existence of Alice’s identity, as well as the set (H', r', g, q, t') related to the proof of knowledge by A of x using a Schnorr’s ZKP.
 - Bob, following the verification step of the Schnorr protocol, goes on the blockchain and checks whether T is such that $T \cdot y'^{-1} = t \cdot t' (t \cdot t'$ being the multiplication of the public keys of S and A).

Bob has then checked whether Alice’s unique couple (x, y) is indeed a couple approved and assigned by S to A without knowing x .

C. ZKP_{a,b} PROTOCOL: LINKING PROOFS OF EXISTENCE OF DOCUMENTS

In order to link a proof of existence of the previously generated Self-Sovereign Identity using a ZKP Schnorr to proofs of existence of other documents registered in separated blocks and on different blockchains services, the proposed ZKP_{a,b} protocol can be employed.

Let’s assume that Alice wants to link κ proofs on separated blocks belonging to different blockchains (or chains) using the ZKP_{a,b} protocol and that a first step for generating a Self-Sovereign Identity with a ZKP Schnorr has been completed. Alice initialises the ZKP_{a,b} protocol randomly generating $(p_1, p_2, \dots, p_\kappa)$ secrets. Alice then calculates the following variables:

$$g_i = g^{p_i}$$

$$\begin{aligned} x_0 &= n = H_S(x) \\ x_i &= x_{i-1} \cdot p_i \\ y' &= g^n \\ LH_i &= LH_i(y') = g^{x_i} \end{aligned}$$

Finalising the ZKP_{a,b} initialisation step described in the previous section, Alice will save g_i and $LH_i(y')$ on the blockchain_{#i} (or on the chain_{#i}). As illustrated by figure 2, Alice can now generate proofs for p_i either in several steps or in one single step.

1) *Several steps proof for p_i*

For each step i , Alice sets up a ZKP_{a,b} with $(a = x_{i-1}, b = p_i)$ and saves σ_i in a block_i.

Let’s note that the challenge for Alice is $c_i = 2u$; this challenge enables the ZKP_{a,b} protocol to be non-interactive.

Alice then sends to Bob:

$$\begin{cases} [(r_1, c_1); (r_2, c_2); \dots; (r_\kappa, c_\kappa)] \\ r_i = (m_i, k_i, o_i) \\ c'_i = H(g, LH_i, (t^{-1} \cdot m_i)^{u-1} + (k_i \cdot o_i)^{u-1}) \end{cases}$$

Bob can now calculate u :

$$\begin{cases} u = c'_i/2, \text{ if } c'_i \text{ is even, OR} \\ u = (c'_i + 1)/2 \text{ otherwise.} \end{cases}$$

For verification, Bob checks whether both following conditions are met:

$$\begin{cases} k_i + o_i = \sigma_i \\ m_i \cdot k_i \cdot o_i = t' \cdot LH_i^{c_i} \end{cases}$$

Using the ZKP_{a,b} properties mentioned in IV-C2, Bob can perform verification every two steps.

2) *Single step proof for p_i*

From a ZKP Schnorr where she proved she knows n , Alice obtained (H, r', g, q, t') . From a ZKP_{a,b} where she proves she knows $a = n \cdot m_{\kappa-1} \dots m_1$ and $b = m_\kappa$, Alice obtained $(m_\kappa, k_\kappa, o_\kappa)$.

Alice then calculates and registers on a blockchain the values of $y' = g^n$, σ_κ and LH_κ .

Let’s note that the challenge for Alice is $c = c_\kappa = H(g, y', (t^{-1} \cdot m_i)^{u-1} + (k_i \cdot o_i)^{u-1})$; this challenge enables the ZKP_{a,b} protocol to be non-interactive.

Alice then sends to Bob:

$$\begin{cases} (m_\kappa, k_\kappa, o_\kappa) \\ (H, r', g, q, t') \end{cases}$$

For verification, Bob checks whether both following conditions are met:

$$\begin{cases} k + o = \sigma_\kappa \\ m_\kappa \cdot k_\kappa \cdot o_\kappa = g^{r'} \cdot (y' \cdot LH_\kappa)^{c_\kappa}, t' = g^{r'} \cdot y'^c \end{cases}$$

Let’s note that for the application described in [28] motivating this research, the objective is to contribute to a match-making process enabling recruitment based on certified skills

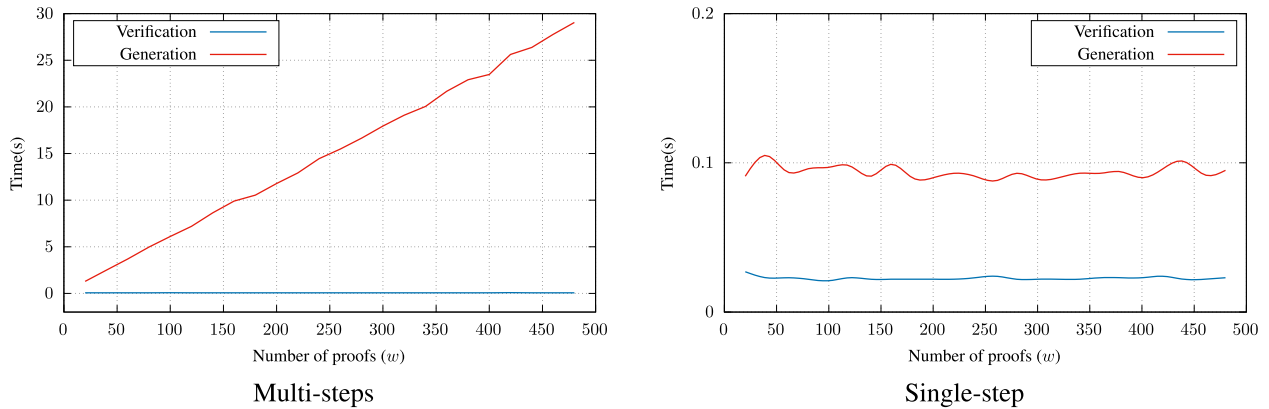


FIGURE 3. Impact of the number of proofs on generation and verification time.

regardless of other personal information in order to be as inclusive as possible. Using $ZKP_{a,b}$ to combine proofs for student identity, working permit or social security number with credentials for skills or diploma would help fulfilling such an objective, provided that skills or diploma data which would have to be disclosed for matchmaking purposes will be stripped from personal data.

VI. EVALUATION

A. SSI-ZKP_{a,b} COMPARISON

A protocol for an SSI embedding the proposed $ZKP_{a,b}$ being an original contribution, a technical analysis emphasizing its design advantages is given below. The proposed SSI design's performance will be compared using the same methodology and criterias (C. Allen's & provability) as well as against the same list of existing SSI as in [3] and [12].

- **Portability:** Information and services about identity should be transportable. Our solution implements this criteria using ZKP since information and services related to identity are condensed into a proof key and a verification key. The two keys can have sizes between 600 bits and 1024 bits depending on the level of security desired. Each user can keep his identity without needing a trusted third party validating the viability of his identity.
- **Interoperability:** Several aspects may reduce interoperability. However, scalability is the only relevant criteria for the proposed protocol. The two proposed ZKP in this article are scalable as the generation and/or verification of proofs is executed in a decentralized way. As a consequence, scalability depends only on the number of proofs that a user will generate. The time necessary for the generation and/or verification steps is polynomial. The proposed SSI ZKP is a generic protocol and it is not limited to particular use cases.
- **Minimalization:** In the proposed SSI using ZKP protocols, shared information is limited to the ones related to the verification of the prover's response. The State is the only trusted (and legitimate) third party introduced in the primary step while certifying the sovereign identity. For any other identity/identifier/attribute, no

information other than that of the proof/verification key is shared.

- **Protection:** The user has full control over his Zero Knowledge Proofs generation using his private key thereby protecting his full ownership rights over his identities.
- **Provable:** The two ZKP are verifiable within a finite and reasonable time (cf. figure 3).

The proposed SSI-ZKP_{a,b} checks all 11 criteria (C. Allen's and provability) and differs from other SSIs checking the 11 criteria by using the two proposed ZKP, thereby addressing the proposed challenge of this article.

B. ZKP_{a,b} SIMULATIONS AND COMPARISON

An feasibility study is supplied based on $ZKP_{a,b}$'s protocol simulation results obtained from a python implementation. The implementation source code is published alongside this article and referenced here as $ZKP_{a,b}$ python implementation repository [29]. Simulations have been run for a cyclic group $\mathbb{Z}_q^{(*,+)}$, q being a prime number chosen such that $q = 2 \cdot q' + 1$ with q' a prime number. In order to comply with current security requirements, simulations have been carried out with a size for q of 1024 bits.

Considering that studies similar to [30] for ZKP Schnorr protocol evaluation are available in the literature, the feasibility study has been delimited to the $ZKP_{a,b}$ protocol. More specifically, the feasibility study has been delimited to (m, k, o) generation and verification both for the multi-step (a) and the single-step (b) proofs. As summarized in Figure 3, an evaluation of the SSI-ZKP_{a,b} is provided for the protocol's time complexity as a function of the number of proofs to be generated and verified. In Figure 3 (a) concerned with multi-step proofs, time complexity is shown to increase linearly with the number of proofs to be generated and to be constant with the number of proofs to be verified. In Figure 3 (b) concerned with single-step proofs, time complexity is shown to be constant with both the number of proofs to be generated and validated.

As highlighted in [18], different ZK-proof systems are based on different cryptographic assumptions and designed

TABLE 1. Qualitative comparison of the proposed ZKP_{a,b} protocol against other ZK-proof algorithms as proposed in [18].

	Prover scalability (quasilinear time)	Verifier scalability (polyalgorithmic time)	Transparency (public randomness)	Post-quantum security
hPKC *	YES	Only repeated computation	NO	NO
DLP **	YES	NO	YES	NO
IP ***	YES	NO	YES	NO
MPC ****	YES	NO	YES	YES
IVC+hPKC *****	YES	YES	NO	NO
zkp-STARK	YES	YES	YES	YES
ZKP _(a,b)	YES	YES	YES	NO

* hPKC : homomorphic Public-Key Cryptography ; ** DLP : Discrete Logarithm Problem ; *** IP : Interactive Proofs based ; **** ; MPC : secure Multi-Party Computation ; ***** ; IVC : Incrementally Verifiable Computation [18].

for different computational systems. Acknowledging such limitations, a qualitative comparison for the proposed ZKP_{a,b} protocol against other ZK-proof algorithms is provided in Table 1. Simulations of ZKP_{a,b}'s scalability for the prover, (m, k, o) generation, as well as for the verifier, (m, k, o) verification, have been reported in Figure 3 while investigating time complexity as a function of the number of proofs to be emitted and verified. Concerning ZKP_{a,b}'s transparency, this criteria has been implemented within the proposed exchange protocol of private keys and use of hash functions.

VII. CONCLUSION

An original Self-Sovereign Identity solution with Zero Knowledge Proof based on blockchain technologies is proposed. This solution is compliant with eIDAS, GDPR as well as criteria introduced by C. Allen and complemented by Q. Stokkink and J. Pouwelse with provability. A ZKP protocol enabling to prove that a and b such that $a \cdot b = n$ are known without revealing a and b has been proposed. Automorphism sequence linking different identities, identifiers and/or attributes as well as different blockchain information has been designed linking the ZKP Schnorr and an original ZKP protocol named ZKP_{a,b}. The properties of group automorphisms have been exploited to create a link between these two ZKP protocols in order to link multiple SSI in a single SSI. In this proposal the creation/acquisition of identity and the management of identifiers/attributes are entirely managed by the user and at most one trusted third party. Once implemented, such a method will minimize the amount of information shared and facilitate the regulation of identification/authentication by allowing only relevant actors to access relevant information.

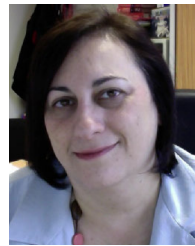
REFERENCES

- [1] M. Takemiya and B. Vanieiev, "Sora identity: Secure, digital identity on the blockchain," in *Proc. IEEE 42nd Annu. Comput. Softw. Appl. Conf. (COMPSAC)*, Jul. 2018, pp. 582–587.
- [2] J. Andraško, "Mutual recognition of electronic identification means under the eIDAS regulation and its application issues," *AD ALTA, J. Interdiscipl. Res.*, vol. 7, no. 2, pp. 1–5, 2017.
- [3] D. van Bokkem, R. Hageman, G. Koning, L. Nguyen, and N. Zarin, "Self-sovereign identity solutions: The necessity of blockchain technology," 2019, *arXiv:1904.12816*.
- [4] U. Der, S. Jähnichen, and J. Stürmeli, "Self-sovereign identity – opportunities and challenges for the digital revolution," 2017, *arXiv:1712.01767*.
- [5] S. Coutor, C. Hennebert, and M. Fajer. (Oct. 2020). *Blockchain et Identification Numérique, Restitution Des Ateliers du Groupe de Travail 'Blockchain et identité' (BCID)*. Paris, France: French Ministère de l'intérieur, Oct. 2020. [Online]. Available: <https://www.vie-publique.fr/rapport/280103-blockchain-et-identification-numerique>
- [6] Y. Wang, Z. Su, N. Zhang, R. Xing, D. Liu, T. H. Luan, and X. Shen, "A survey on metaverse: Fundamentals, security, and privacy," *IEEE Commun. Surveys Tuts.*, vol. 25, no. 1, pp. 319–352, 1st Quart., 2023.
- [7] Q. Stokkink and J. Pouwelse, "Deployment of a blockchain-based self-sovereign identity," in *Proc. IEEE Int. Conf. Internet Things (iThings) IEEE Green Comput. Commun. (GreenCom) IEEE Cyber, Phys. Social Comput. (CPSCom) IEEE Smart Data (SmartData)*, Jul. 2018, pp. 1336–1342.
- [8] K. Cameron, *The Laws of Identity*, vol. 12. Redmond, WA, USA: Microsoft Corporation, 2005, pp. 8–11.
- [9] R. Shrestha and S. Kim, "Integration of IoT with blockchain and homomorphic encryption: Challenging issues and opportunities," in *Advances in Computers*. Amsterdam, The Netherlands: Elsevier, 2019, vol. 115, pp. 293–331.
- [10] L. Jarry-Lacombe and V. Langard, "Systems and computer-based methods of document certification and publication," U.S. Patent US20 200 099 511 A1, Mar. 14, 2020. [Online]. Available: <https://patents.google.com/patent/US20200099511A1/en>
- [11] N. Kulabukhova, "Self-sovereign identity as trusted root in knowledge based systems," in *Proc. Int. Conf. Comput. Sci. Appl.* Cham, Switzerland: Springer, 2020, pp. 14–24.
- [12] C. Allen. (2016). *The Path to Self-Sovereign Identity*. Accessed: Feb. 28, 2023. [Online]. Available: <http://www.lifewithalacrity.com/previous/>
- [13] D. Benarroch, M. Campanelli, D. Fiore, K. Gurkan, and D. Kolonelos, "Zero-knowledge proofs for set membership: Efficient, succinct, modular," in *Proc. Int. Conf. Financial Cryptogr. Data Secur.* Cham, Switzerland: Springer, 2021, pp. 393–414.
- [14] D. Benarroch, K. Gurkan, R. Kahat, A. Nicolas, and E. Tromer, "Community proposal: Zkinterface, a standard tool for zero-knowledge interoperability," QEDIT, Tel Aviv-Yafo, Israel, Tech. Rep., 2020.
- [15] D. Benarroch, A. Nicolas, J. Thaler, and E. Tromer, "Community proposal: A benchmarking framework for (zero-knowledge) proof systems," QEDIT, Tel Aviv-Yafo, Israel, Tech. Rep., 2020.
- [16] The World Bank, "A digital stack for transforming service delivery: Id, payments, and data sharing," Washington, DC, USA, Tech. Rep. 170268, Feb. 2022. [Online]. Available: <http://documents.worldbank.org/curated/en/099755004072288910/pdf/P1715920ed6b5990d60b83e037f756213782.pdf>
- [17] E. C. Crites and A. Lysyanskaya, "Delegatable anonymous credentials from mercurial signatures," in *Proc. Cryptographers Track RSA Conf.* San Francisco, CA, USA: Springer, Mar. 2019, pp. 535–555.
- [18] E. Ben-Sasson, I. Bentov, Y. Horesh, and M. Riabzev, "Scalable, transparent, and post-quantum secure computational integrity," *IACR Cryptol. ePrint Arch.*, vol. 46, p. 2018, 2018.
- [19] D. Capko, S. Vukmirovic, and N. Nedic, "State of the art of zero-knowledge proofs in blockchain," in *Proc. 30th Telecommun. Forum (TELFOR)*, Nov. 2022, pp. 1–4.
- [20] The Selfkey Foundation. *Selfkey*. Accessed: Nov. 2022. [Online]. Available: <https://selfkey.org/wp-content/uploads/2022/08/selfkey-whitepaper-en.pdf>

- [21] lifeID. *Ban Open-Source, Blockchain-Based Platform for Self-Sovereign*. Accessed: Nov. 2022. [Online]. Available: <https://lifeid.io/whitepaper.pdf>
- [22] J. Nick, T. Ruffing, Y. Seurin, and P. Wuille, "MuSig-DN: Schnorr multi-signatures with verifiably deterministic nonces," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2020, pp. 1717–1731.
- [23] N. Bitansky, D. Dachman-Soled, S. Garg, A. Jain, Y. T. Kalai, A. López-Alt, and D. Wichs, "Why 'Fiat-Shamir for proofs' lacks a proof," in *Proc. Theory Cryptogr. Conf.* Cham, Switzerland: Springer, 2013, pp. 182–201.
- [24] J. Camenisch, A. Kiayias, and M. Yung, "On the portability of generalized Schnorr proofs," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.* Cham, Switzerland: Springer, 2009, pp. 425–442.
- [25] K. S. McCurley, "The discrete logarithm problem," in *Proc. Symp. Appl. Math.*, vol. 42, 1990, pp. 49–74.
- [26] E. Jeřábek, "Integer factoring and modular square roots," *J. Comput. Syst. Sci.*, vol. 82, no. 2, pp. 380–394, Mar. 2016.
- [27] E. Costa and D. Harvey, "Faster deterministic integer factorization," *Math. Comput.*, vol. 83, no. 285, pp. 339–345, May 2013.
- [28] W. Azan, P. Valiorgue, E. Peyrol, P. H. B. Hadid, Y. Li, and L. F. M. Miranda, "Proposal for an integrative performance framework based on distributed ledger technology dedicated to higher education students entering the labor market," in *Proc. IEEE 6th Int. Conf. Logistics Oper. Manage. (GOL)*, Jun. 2022, pp. 1–6.
- [29] (2023). *Link to the SSI-ZKP_{a,b} Protocol Implementation*. [Online]. Available: <https://forge.univ-lyon1.fr/mohameden.dieye/ssi-zkp.git>
- [30] F. Hao, "Schnorr non-interactive zero-knowledge proof," Newcastle Univ., Newcastle upon Tyne, U.K., Tech. Rep. RFC8235, 2017. [Online]. Available: <https://www.rfc-editor.org/rfc/rfc8235>



EL-HACEN DIALLO received the degree in computer science engineering from the Polytechnic School of Nouakchott, in 2018, and the Ph.D. degree from the University of Paris-Saclay, in April 2022. He is currently a Postdoctoral Researcher with University Claud Bernard Lyon 1 and a member of the Laboratory of Computer Graphics, Images and Information Systems (LIRIS). His research interests include distributed systems, cybersecurity, vehicular networks, and blockchain technology.



PARISA GHODOUS is currently a Full Professor with the Computer Science Department, University Claude Bernard Lyon I. She is also a member of the Laboratory of Computer Graphics, Images and Information Systems (LIRIS UMR 5205). Her research and education interests include blockchain, cloud computing, interoperability, web semantic, service science, collaborative modeling, product data exchange, and modeling and standards. She is in the editorial boards of *CeRA*, *ICAE*, and *IJAM* journals and in the committees of many relevant international associations, such as concurrent engineering and interoperability. She was involved in more than 20 European projects, four as a Coordinator (VET4APPS, BLISS, MACHINA, and CHAISE).



Pierre Valiorgue, Jean-Patrick Gelas, Parisa Ghodous, and Frédérique Biennier.

MOHAMEDEN DIEYE received the master's degree in statistics-econometrics, IT security, and data science from Claude Bernard University Lyon 1 (UCBL). He is currently a young Researcher, who carries out his work within a research team of the Liris Laboratory, UCBL. His current research interests include theoretical (cryptography, ZKP, and SSI) and practical (DID, interoperability, the IoT, and blockchain) aspects. He works in close collaboration with



FRÉDÉRIQUE BIENNIER received the M.Sc. degree in computer science, in 1988, and the Ph.D. degree in computer science and automata, in 1990. She is currently a Professor with INSA Lyon, Computer Science Department and UMR CNRS LIRIS, Service Oriented Computing Research Team. She has published several papers linked to these fields in many journals (*IJCIM*, *JSA*, and *IEEE TRANSACTIONS ON SERVICES COMPUTING*) and conferences and has conducted 15 Ph.D. thesis and supervise currently two Ph.D. thesis in the field of service ecosystems, governance, and security. She has participated and/or coordinated several projects at the regional or national scale (the most recent is the INvestissement d'Avenir PICS 2016 aiming at providing a user-centric personal information control systems). Her research interests include service computing in the context of interoperable information systems, security and privacy, and the multi-cloud IoT services deployment.



PIERRE VALIORGUE is currently a Research Engineer with Université Claude Bernard Lyon 1. His research interests include decentralized autonomous organization, blockchain, and skills management.



ÉRIC PEYROL is currently an Associate Professor with Université Claude Bernard Lyon 1. From 2016 to 2021, he was the Vice President of Continuing and Lifelong Training, where he initiated and implemented the skill-based approach with Université Lyon 1. He is also the Vice President of Entrepreneurship and Partnerships. As part of this role, he coordinates and takes part in innovative state sponsored training projects (in the fields of artificial intelligence and energy), where he implements his expertise on the development of student's employability. He also takes part in the development of inclusive solutions for students entering the labor market.



blockchain technologies, energy efficiency, and cloud computing.

JEAN-PATRICK GELAS received the D.E.A. degree in computer science from Ecole Normale Supérieure de Lyon (ENS Lyon), France, in 2000, and the Ph.D. degree from Université Claude Bernard Lyon 1 (UCBL), France, in December 2003. He is currently an Assistant Professor, teaching computer science (systems, computer networks, embedded systems, and blockchain technologies) with UCBL. His research interests include large scale distributed systems, such as