IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 16, 2019, accepted July 22, 2019, date of publication July 25, 2019, date of current version August 12, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2931173

In Search of Self-Sovereign Identity Leveraging Blockchain Technology

MD SADEK FERDOUS[1,2], **FARIDA CHOWDHURY**[1], **AND MADINI O. ALASSAFI**[3]
[1]Department of Computer Science and Engineering, Shahjalal University of Science and Technology, Sylhet 3114, Bangladesh
[2]Imperial College Business School, Imperial College London, London SW7 2AZ, U.K.
[3]Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah 21589, Saudi Arabia

Corresponding author: Md Sadek Ferdous (sadek-cse@sust.edu)

ABSTRACT In recent times, with the advent of blockchain technology, there is an optimism surrounding the concept of self-sovereign identity which is regarded to have an influential effect on how we interact with each other over the Internet in future. There are a few works in the literature which examine different aspects of self-sovereign identity. Unfortunately, the existing works are not methodological and comprehensive at all. Moreover, there exist different notions of what the term self-sovereign identity means. To exploit its full potential, it is essential to ensure a common understanding in a formal way. This paper aims to achieve this goal by providing the first-ever formal and rigorous treatment of the concept of self-sovereign identity using a mathematical model. This paper examines the properties that a self-sovereign identity should have and explores the impact of self-sovereign identity over the laws of identity. It also highlights the essential life-cycles of an identity management system and inter-relates how the notion of self-sovereign identity can be applied in these life-cycles. In addition, the paper illustrates several envisioned flows involving a self-sovereign identity leveraging blockchain technology covering different aspects of an identity management system. All in all, this paper presents the first formal and comprehensive step toward an academic investigation of self-sovereign identity.

INDEX TERMS Identity, identity management system, self-sovereign identity, blockchain.

I. INTRODUCTION

In recent years, one of the most widely-used terms in the landscape of Identity Management is *Self-Sovereign Identity*. With the proliferation of online services in the last fifteen years or so, the management of the identities of users and services has taken a central stage and in many ways, has become the foundation upon which different online services are built. Different needs and requirements in different use cases have driven the development of several Identity Management Systems (IMS), most of which are (service) provider-centric, without realising that a muddled jungle of identities is being created. The ultimate consequence of this is that users, having ended up with a large number of identities, often feel lost in this jungle. It becomes increasingly difficult to manage those scattered identities. What is worse is that these IMS only serve the needs of the providers to manage their user-bases and they provide an extremely limited capability for the user to exercise control over their identity data [1]. Without any meaningful control over such data, users are hardly aware how their data is being abused by the providers. The concept of Self-Sovereign Identity has emerged with the promise to usher a new era in the landscape of Identity where the user, and only the user, is to have the full control over their identity data with strong support for a user-controlled data management facility.

A few exciting recent developments in technology have convinced many enthusiasts to believe that we are in the fore-front of a technological breakthrough which can be leveraged to realize a Self-Sovereign Identity Management System. In particular, with the advent of blockchain technology, many believe that such technology can provide the technical foundation upon which the concept of self-sovereign identity can be realized. This has fuelled the excitement where many use-cases for different scenarios are being explored to understand the suitability of such a system. Even though such exploration is essential to advance the state-of-the-art, one inadvertent side-effect is that there exist different notions what the term self-sovereign identity means. It has been defined in a multitude of ways in different places which

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

only add to the confusion. Even though it might be leveraged in different ways in different scenarios, we believe that a common understanding on what a self-sovereign identity means is fundamental to exploit its full potential.

The motivation of this article is to underline this common understanding. Towards this aim, we examine and critically analyze the existing definitions of self-sovereign identity. Based on our analysis and founded upon mathematical properties, we provide the first formal definition of self-sovereign identity in this article. Then, we utilize this definition to concretize the properties it must hold. In addition, we analyze the impact of self-sovereign identity over the well-known *Laws of Identity*[2] and argue the role a blockchain system can play in realising a self-sovereign identity. Furthermore, we explore different aspects of identity management concerning self-sovereign identity and envision their corresponding use-cases utilizing blockchain technology. With these contributions, this article presents the first formal and comprehensive step towards an academic exploration of self-sovereign identity.

The article is organized as follows. In Section II, we provide a brief description of a number of terminologies that are used in the literature of identity management and also utilized in this article. In Section III, we present an existing mathematical model of identity. We explore the evolution of different identity management systems in Section IV. Next, in Section V, we present and analyze the existing definitions of self-sovereign identity, concretize the essential properties of self-sovereign identity using a taxonomy, mathematically formalize the concept of self-sovereign identity and formulate the laws of self-sovereign identity. Subsequently, we present the life-cycle of identity in an IMS in Section VI and discuss different aspects of blockchain in Section VII with a specific focus on the role blockchain can play with respect to self-sovereign identity. Section VIII explores different use-cases involving the life-cycle of self-sovereign Identity. Finally, we conclude in Section IX along with brief discussion and a hint of future work.

II. TERMINOLOGIES

In this section, we provide a short definition of different terms, related to identity, which will be used throughout this article. Many of these terms have been defined in different (sometimes in conflicting) ways in different literature. Hence, in order to avoid any confusion, it is useful to underline the semantic meanings with which these terms will be used in this article.

It is to be noted that we do not wish to formulate any novel definition of any of these terms. We are merely interested to present a definition, from a selection of online resources, which provides the most suitable semantic meaning, in our opinion, for a particular term that fits the scope of this article.

A. ENTITY

According to [3], an entity is a physical or logical object which has a separate distinctive existence either in a physical or logical sense. This existence is often characterized "*through the measurement of its different attributes*" [4]. Within the scope of this article, we restrict our focus on digital entities, which is merely a digital representation of an entity and assumes that a digital entity is either a person or an organization which provides some sort of online services.

B. CONTEXT/APPLICATION DOMAIN

In the scope of this article, we denote a context as an environment under which a (digital) entity exists and operates [5]. It can be regarded as the application domain or namespace in which an entity is represented and identified uniquely. Hence, the term context and application domain (or simply domain) will be used interchangeably throughout this article.

C. TRUSTED THIRD PARTY (TTP)

According to [4], a trusted third party (TTP) "*is an entity trusted by multiple other entities within a specific context*". A TTP is leveraged to make *assertions* (discussed later), about an entity, which are trusted by other entities.

D. ATTRIBUTE

According to [5], "*an attribute is a distinct, measurable named property belonging to an entity in a context whose value can be used to identify the entity (not necessarily uniquely) within the context*". Accordingly, each attribute has a name and value. The name (or simply the attribute) alone cannot identify an entity without its corresponding value. In a context, the value of an attribute is provided either by an entity or a (trusted) third party.

E. IDENTIFIER

According to [5]: *"an identifier is an attribute whose value can be used to uniquely identify an entity within a context"*. There may be many attributes in a context (domain) that can uniquely identify an entity at a certain point in time. However, when more entities are added into the domain, it may happen that the attribute no longer uniquely identifies an entity. To avoid unnecessary complications, each application domain considers one attribute as the identifier and ensures that its value can always uniquely identify an entity.

F. CREDENTIAL

A credential is an attribute that accompanies an identifier and whose value is utilized to attest the authority of an entity over the supplied identifier value via a process called the *authentication process* (see below). The simplest and weakest form of a credential is a password. Digital certificates, biometrics such as fingerprints, voice recognition, retina scan or secure hardware tokens such as the OTP (One Time Password) are examples of more secure form of credentials.

G. CLAIM

A claim is a statement about an entity [2, 6]. In rudimentary sense, such a statement consists of an attribute name-value pair. It can also be a collection of several attribute name-value

pairs. Such a claim can be made by the entity itself or by a TTP.

H. ASSERTION

An assertion, also known as a verifiable claim, is a special type of claim, regarding an entity [7]. To ensure the authenticity and integrity of an assertion, it is often distributed as a cryptographic token which can be easily verified by the recipient of such a token [8].

I. ATTESTATION

Attestation is the process by which a recipient verifies an assertion/credential.

J. PARTIAL IDENTITY & IDENTITY

The partial identity of an entity, dominantly a user, within a domain is the set of all attribute name-value pairs bound to the entity using the identifier within that domain [5]. In essence, the partial identity of an entity fully encodes the identity of an entity in a domain.

On the other hand, the (total) identity of an entity can be defined as the union of all its partial identities in all domains [5]. Note that such a combined view of identity only makes sense from the first-person perspective of an entity.

In the literature, a partial identity is commonly termed as the identity of a user. However, we would to like to underline the subtle difference in their meaning.

K. IDENTITY MANAGEMENT

Identity Management is a process to facilitate the management of (partial) online identities [3]. It consists of technologies and policies for representing and identifying entities with their (partial) digital identities and leveraging such identities in order to access online services in different application domains [9].

L. IDENTITY MANAGEMENT SYSTEM

A system that is used for identity management is known as the Identity Management System (IMS) [5].

M. PROFILE

One of the functionalities of an IMS is to enable users to share their partial identities between different organizations (domains). During this process, for the sake of privacy, users usually do not share their full partial identities between two domains. Instead, a privacy-friendly approach is to share a limited view of a user's data across domains. Such a limited view is defined as the *Profile* of a user.

N. ACTORS

Each IMS involves the following three parties:

- **Identity Provider (IdP):**In the traditional setting, an IdP is responsible for storing the partial identity of a user within its domain and sharing it across different domains.
- **Service Provider (SP):**An SP is responsible for providing online services to a user based on the profile of the user as received from the IdP.
- **User:**A user is the entity whose partial identity is stored in an IdP and who accesses services from an SP.

III. DIGITAL IDENTITY MODEL

If Numerous works can be found in which the term "*Identity*" is defined in different ways. In most cases, the definitions are textual and express different semantic meanings [5]. A definition/model founded on the mathematical properties would be beneficial to purge the semantic inconsistencies in textual definitions. Surprisingly, in the academic literature, there is little work on defining a mathematical model of digital identity for any entity. The only mathematical model that can be found is called the *Digital Identity Model (DIM)* as introduced in [5].

According to this model, the (whole) identity of an entity (mainly focusing on a user) is actually distributed in different partial identities which are valid within different domains (contexts) of different enterprises (organizations). Since the partial identity of a user is only valid within a domain, it is essential to specify the domain whenever a partial identity is mentioned. Each such partial identity consists of a number of attributes and their corresponding values, valid within the domain of a particular organization. Next, we briefly present this mathematical model.

Let us assume that D denotes the set of domains and $d \in D$ defines the domain of a single organization whereas U_d denotes the set of users, A_d denotes the set of attributes and AV_d denotes the set of values for those attributes within d. Then, we can relate users and their attributes in a domain by the following partial function:

Definition 1: Let $atEntToVal_d : A_d \times U_d \rightarrow AV_d$ *be the (partial) function that for an entity and attribute returns the corresponding value of the attribute in domain* d.

The function is partial as not all entities have a value for each attribute. This also makes sense in practical systems as in many such systems, users are required to provide values for a number of attributes (e.g. email, telephone number, etc.). However, there remain some optional attributes (e.g. age, postal addresses, etc.) for which users may not provide any values.

The identifier $i \in A_d$ in domain d is defined as the attribute having the following two conditions:

- $atEntToVal_d\,(i, u)$ *is defined for all* $u \in U_d$
- $atEntToVal_d\,(i, u_1) \neq atEntToVal_d\,(i, u_2)$ *for all distinct* $u_1, u_2 \in U_d$

The above conditions represent the following (required) properties of an identifier in a domain:

- each entity in the domain must have a value for the identifier and
- the value of the identifier uniquely identifies an entity in the domain.

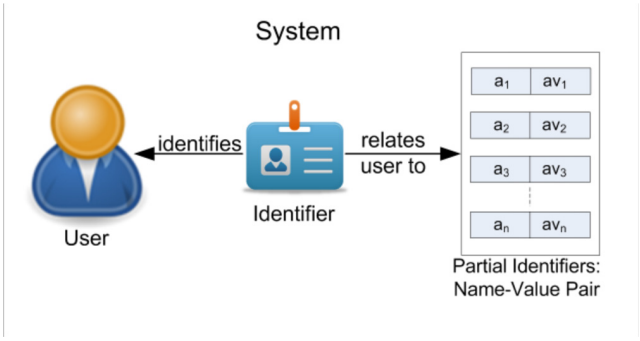

FIGURE 1. Relation between a user, identifier and partial identifiers.

Intuitively, the attributes other than the identifier are regarded as partial identifiers as their values cannot uniquely identify a user in a context.

Next, the partial identity of a user (u) is defined using the following definition:

Definition 2: For a domain d, the partial identity of a user $u \in U_d$, *denoted* $parIdent_d^u$, *is given by the set*:

$$\{(a, v)\} \,|a \in A_d, atEntToVal_d\,(a, u) \text{ is defined and equals } v$$

If it is assumed that there are n valid attribute-value (including the identifier) pairs for a user u, the partial identity of u in d can also be defined as:

$$parIdent_d^u = \{(i_d, v_{i_d}),(a_1, v_1),(a_2, v_2),(a_3, v_3),(a_{n-1}, v_{n-1})\}$$

Here, i_d *and* v_{i_d} represent the identifier and its corresponding value in domain d respectively.

The (total/whole) identity of an entity can be defined as the union of all her partial identities in all domains.

Definition 3: For an entity $u \in U$, *the identity of u is given by the set*:

$$ident^u = \bigcup\{(d, parIdent_d^u)|d \in D \text{ and } u \in U_d\}$$

Graphical representations of this model are illustrated in Figure 1 and Figure 2. Figure 1 presents the inter-relation between a user, identifier and partial identifiers while Figure 2 illustrates the identity of a user according to the Digital Identity model.

Mathematically, a profile is a subset of the partial identity of a user within a domain:

$$profile_d^u \subseteq parIdent_d^u.$$

Hence, we can define the profile of a user $u \in U_d$ in domain d in the following way, where $u \leq n$:

$$profile_d^u = \{(a_1, v_1), (a_2, v_2), (a_3, v_3), \ldots, (a_j, v_j)\}$$

IV. EVOLUTION OF IDENTITY MODEL

The landscape of identity management has gone through an evolutionary path: starting with the simplest model and then evolving in different phases with the introduction of newer models. This evolution is depicted in Figure 3.

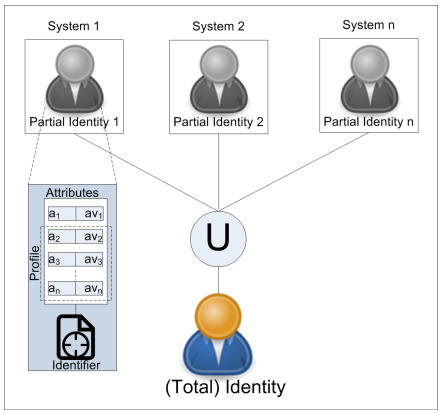

FIGURE 2. The (total) identity of a user.

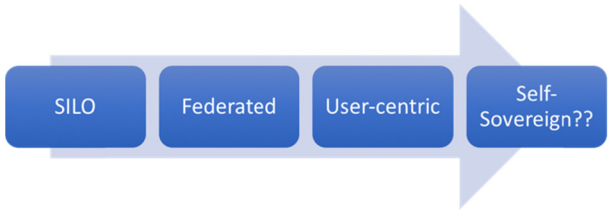

FIGURE 3. Evolution of identity models.

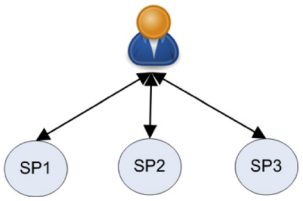

FIGURE 4. The SILO model.

The first three model is briefly presented below whereas the fourth is the subject for discussion of the current article.

A. SILO MODEL

The Isolated User Identity (SILO) Model represents the most common and the simplest identity management model [1]. There are only two parties involved in the scenario: a service provider combined with its own IdP and its users (Figure 4). Each service provider provides the identifier (e.g. username) and the corresponding credential (e.g. password) to the clients who wish to receive its services. Each SP has its own identity domain and identity operations performed in one domain are not valid in other domains. When a user wishes to access a service from different SPs, she needs to visit and authenticate to each service provider separately. All these result in a user ending up with numerous partial identities (along with the identifiers with their corresponding credentials) which become increasingly difficult to manage. Currently, all major and leading online service providers such as Google, Yahoo, Amazon, EBay, etc. follow this model, however, trends are changing towards other models.

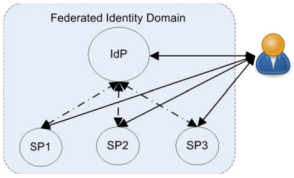

FIGURE 5. The federated model.

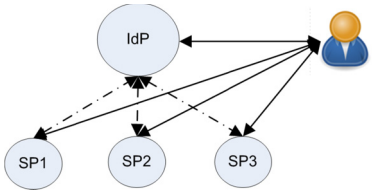

FIGURE 6. The user-centric model.

B. FEDERATED MODEL

In the Federated model, each single identity domain consists of a single IdP and one or more SPs (Figure 5) [1]. The IdP issues identifiers and the related credentials to the user. The SP depends on the IdP for authenticating the user and providing user attributes and their values to the SP. To access any service, users authenticate themselves to the IdP and once authenticated, are redirected to the service provider to access the service. Once a user is authenticated to the IdP, she can access services from all service providers that share the same IdP. The shared identity domain is known as the Federated Identity domain and is created once a notion of trust is established among the IdP and the corresponding SPs. This notion of trust is created by establishing a contract between the corresponding entities. This model is hugely popular where the identity data is only shared between trusted entities, e.g. in governmental services and educational institutions. The most dominant example of this model is any system based on Security Assertion Markup Language (SAML) [10].

C. USER-CENTRIC MODEL

A user-centric model is similar to the federated model [1]. In this model, a number of SPs can share a single IdP, however, there is no need to establish the notion of trust among the entities (Figure 6). Whenever a user tries to access a service by an SP, the user is forwarded to the requested IdP where the user authenticates herself. Then, the IdP releases the identity data of the user using a profile to the SP where an authorization decision is taken based on the profile to grant or reject the user request for accessing the service. With the absence of any notion of trust, every entity in this model trusts each other. For this reason, this model is also known as the Open-trust model. Examples of the systems leveraging this model are systems based on OpenID [11] and OAuth protocols [12]. This is a dominant model in the current setting of web services provided by large social networking service providers such as Facebook, Google and so on.

V. SELF-SOVEREIIGN IDENTITY

In this section, we examine and critically analyze the existing definitions for Self-sovereign Identity (Section 5.1) and concretize the essential properties of self-sovereign identity using a taxonomy (Section 5.2). Then, we present a mathematical model of self-sovereign identity in Section 5.3. Finally, we analyze the impact of self-sovereign identity over the well-known Laws of Identity in Section 5.4.

A. EXISTING DEFINITIONS

In recent years, we have noticed growing interest and activities surrounding self-sovereign identity. Different groups of people are experimenting the numerous ways this concept can be underpinned and deployed technologically. A lack of mutual cohesion among these groups have led towards creating disparate, sometimes even contradictory, notions of self-sovereign identity.

Next, we analyze the existing definitions/notions of self-sovereign identity. We plan to leverage our analysis to understand the underlying common theme in order to use it as a foundation for proposing a formal definition of the concept.

Being a relatively new notion, the amount of scholarly works on this domain in the academic community is quite minimal. A very few examples of some of the academic works in this domain can be found in [13]–[17]. In [13], the authors explored the concept of self-sovereign identity as well as presented its challenges and opportunities in a rather informal way. On the other hand, the authors in [14] investigated different components of a self-sovereign identity using existing non-academic literature. The main focus of other works in [15]–[17] is on the application of self-sovereign identity, more specifically, to explore how a self-sovereign identity system can be built and developed. Within this aim, they proposed developed different systems.

Unfortunately, none of these works explored the fundamental concept of self-sovereign identity. Interestingly, the fundamental conceptual work on this domain has emerged in the technical community, published online via different forums in the form of articles and whitepapers. Therefore, to further our cause, we resort to these handful available resources to understand how this concept has been fundamentally defined.

One of the most influential works in this domain is an online article by Christopher Allen where he debated the need for self-sovereign identity by drawing an evolution of online identity [8]. In his article, the concept of self-sovereign identity is defined as follows:

> "*. . . the user must be central to the* ***administration*** *of identity. That requires not just the* ***interoperability*** *of a user's identity across multiple locations, with the* ***user's consent****, but also true user* ***control*** *of that digital identity, creating user autonomy. To accomplish this, a self-sovereign identity must be* ***transportable****; it can't be locked down to one site or locale. A self-sovereign identity must also allow ordinary users to make* ***claims****, which could include*

> *personally identifying information or facts about personal capability or group membership. It can even contain* ***information*** *about the user that was* ***asserted*** *by other persons or groups.*"

In a way, this definition identifies several crucial properties of a self-sovereign identity, highlighted in bold. The underlying notions of these properties are the autonomous administration of portable identities, user-controlled attribute disclosure based on a user's consent and interoperability of identity. The definition also hints towards the constitution of self-sovereign identity with the statement that it may consist of claims: either self-asserted or asserted by a TTP. However, the details of how to model such an identity is completely missing in the online article. The definition also contains a misstatement: "*A self-sovereign identity must also allow ordinary users to make claims"*. We argue that an identity cannot allow a user to make claims since an identity is just a digital representation of a user, consisting of different attribute name-value pairs. Instead, an IMS is required to facilitate or release a profile consisting of claims.

The Sovrin Foundation[1] is a not-for-profit global consortium aiming towards building and governing a network of self-sovereign identity, known as *Sovrin Identity Network*. In their whitepaper, another definition of self-sovereign identity can be found identifying three crucial properties: *individual control*, *security* and *full portability*. The synopsis of the definition, with highlighted crucial properties, is quoted below [18]:

> "*The individual (or organization) to whom the identity pertains completely* ***owns***, ***controls*** *and* ***manages*** *their identity. In this sense the individual is their* ***own identity provider****—there is* ***no external party*** *who can claim to* ***"provide" the identity*** *for them because it is intrinsically theirs.*
> *. . . You can reveal some or all of it some of the time or all of the time. You can record your* ***consent to share*** *data with others, and easily* ***facilitate*** *that* ***sharing****. It is* ***persistent*** *and* ***not reliant*** *on any single third party.* ***Claims*** *made about you in identity transactions can be* ***self-asserted****, or* ***asserted by a 3rd party*** *whose* ***authenticity*** *can be independently* ***verified*** *by a relying party.*"

Similar to Allen's definition, this definition also captures the notion of self-controlled manageability of identity and explicit consent for sharing identity data, which may consist of claims. In addition, a few new characteristics have been specified here: user-controlled ownership, the capability for a user to act as her IdP, persistency of identity and the non-reliance of identity data onto a single party. However, like Allen's definition, the whitepaper fails to properly model a self-sovereign identity.

Another definition of self-sovereign identity is quoted from [19]:

> "*Self-sovereign identity means not having to* ***ask permission to create, provide, or terminate*** *the use of identifying information for correlation across contexts.*
> *A self-sovereign identity system allows us to* ***selectively*** *present our own means of identification for correlating our interactions in formal and informal situations around the world, online and off. A good self-sovereign identity system will. . . put the individual in* ***control*** *of most uses of identity. . . enable individuals to exercise greater* ***control*** *. . .* "

The underlying notions in this definition are the autonomous management of identity, selective disclosure of identity and controllability.

A final definition of self-sovereign identity is taken from [7]:

> *"A form of identity that attempts to balance* ***transparency***, ***fairness***, *and* ***support*** *of the commons with protection for the individual."*

This definition is rather vague and abstract, even so it underlines three properties, highlighted in bold.

The last two definitions also do not consider how to model such an identity. In the subsequent sections, we explore how we can develop a model of self-sovereign identity in a formal way which captures the common properties highlighted in different definitions.

B. PROPERTIES OF SELF-SOVERIGN IDENTITY

In this section, we analyze different properties of self-sovereign identity as highlighted in different definitions. The properties are presented using a taxonomy to highlight their classifications. The main motivation of this analysis is to better understand the properties in terms of their semantic meaning. This will help us to develop the model in a rigorous fashion.

A number of required properties for a self-sovereign identity have been presented [8], [19]. The author in [8] has introduced ten essential properties, namely Existence, Control, Access, Transparency, Persistence, Portability, Interoperability, Consent, Minimalization and Protection. A taxonomy of these properties has been presented in [18] where they have been classified in three groups: Controllability, Security and Portability. Another taxonomy of self-sovereign identity is presented in [19] consisting of three groups: Control, Acceptance and Zero Cost. Several properties in this taxonomy are similar to what have been proposed in [8]. However, it also presents some additional properties such as Choosability (Opt-In and Opt-Out), Standard and Cost.

However, we feel that the none of the taxonomies is comprehensive as each of them missing some properties presented in other. Also, it seems that some properties categorized under a group are out of place. For example, in [18], the Existence and Persistence properties under the Controllability group

[1] https://sovrin.org/

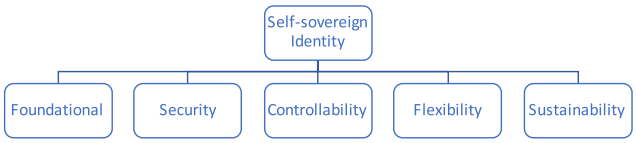

FIGURE 7. Taxonomy of self-sovereign identity.

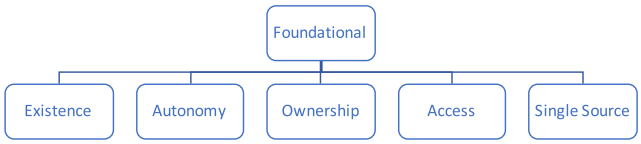

FIGURE 8. Taxonomy of the foundational property.

FIGURE 9. Taxonomy of the security property.

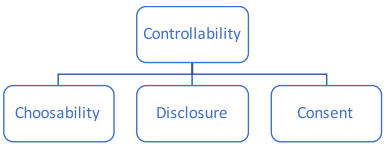

FIGURE 10. Taxonomy of the controllability property.

do not fit well. To rectify this situation, we have created an extended taxonomy, based on the two taxonomies presented in [18], [19], by introducing three additional groups called *Foundational*, *Flexibility* and *Sustainability* and then rearranging some properties in different groups. The taxonomy is presented in Figure 7. Different properties under each group are presented below.

1) FOUNDATIONAL PROPERTY

The properties (Figure 8) fundamental to the notion of self-sovereignty are placed under this group. These represent the core group of properties without which a self-sovereign identity cannot exist. The properties are depicted next.

- **Existence**. A self-sovereign identity must enable a user to encode her characteristics digitally to assert her existence in the digital domain. This essentially is a quasi-representation of one's self in digital form.
- **Autonomy**. A self-sovereign identity must support full autonomy on the management and administration of identity information. A user must be fully independent on creating such an identity, as many as required, without relying on any party and be able to update/remove it when she wishes to do so.
- **Ownership**. A user must be the ultimate owner of a self-sovereign identity. This applies to any information encoding the identity, including the claims, whether it is self-asserted or provided by a third party.
- **Access**. A user must have unrestricted access to her identity information. She must be able to retrieve every single piece of information, including claims and assertions, which constitute her identity.
- **Single source.**A user should be the single source of truth regarding her identity. She should be the ultimate guardian to create self-asserted and/or accumulate third party asserted claims leveraging her identity and distribute them when required. This will ensure that third party cannot collude to exchange her identity data without her knowledge. For ease of use, however, she can delegate this task to an autonomous agent which is under her control.

2) SECURITY PROPERTY

Under this group, we place those properties which are used to ensure the security of self-sovereign identity. The properties presented under this group are as crucial as the foundational properties to guarantee that the concept of security is tightly-coupled with any self-sovereign identity. The properties are presented in Figure 9 and discussed next.

- **Protection**. A self-sovereign identity should be well protected with latest cryptographic mechanisms satisfying the CIA (Confidentiality, Integrity and Authenticity) and non-repudiation properties. Each interaction involving an identity must be authorized and the corresponding entities must be properly authenticated. Any identity information must be stored in a secure manner and transmitted via a secure channel. Any system handling such identity must support a fine-grained access control mechanism to ensure the required level of controllability of the user over their identity.
- **Availability**. A self-sovereign identity must be readily available and accessible from different platforms when required by its owner. It must be robust enough to be recoverable even with the loss of a particular storage medium where such data is stored.
- **Persistence**. A self-sovereign identity must be persistent, at least as long as it is required by its owner. For third party claims, they must be persistent until the asserted authority ceases to exist.

3) CONTROLLABILITY PROPERTY

Under this group, we place those properties which can be used to control any identity data. The properties belonging to this group are illustrated in Figure 10 and described next.

- **Choosability**. A user must have the ultimate control to decide when she wishes to release an identity data and to which entity for whatever purpose.
- **Disclosure**. When an identity data is released to a third party, a user must have the ability to selectively disclose

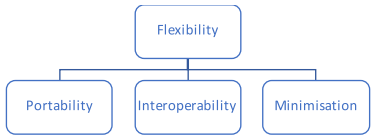

FIGURE 11. Taxonomy of the flexibility property.

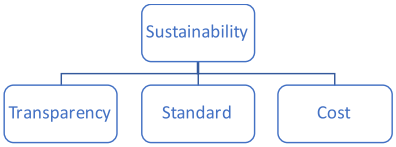

FIGURE 12. Taxonomy of the sustainability property.

particular attributes so that the user can exercise ultimate control over her data.

- **Consent**. Every single piece of identity data must be released to a third party only after the corresponding user has consented to do so.

4) FLEXIBILITY PROPERTY

In order to ensure that a self-sovereign identity can operate with different systems, it needs to be as much flexible as possible. Under this category, we have placed those properties that are related to the flexibility of self-sovereign identity. The properties under this category are presented in Figure 11 and discussed next.

- **Portability**. A self-sovereign identity must be portable. This will ensure that a user's partial identity can be transferred to a medium or platform when the previous medium or platform disappears form the landscape due to a variety of reasons. A portable identity will also ensure the persistence of identity for a longer period of time.
- **Interoperability**. Due to the heterogeneous nature of the Internet and online services, a self-sovereign identity must be designed in such a way that it can achieve the maximum level of interoperability. It must also ensure that it is backward compatible with legacy identity systems for a period of time to ensure a smoother interaction between those systems and a self-sovereign system.
- **Minimisation**. The disclosure of identity must be minimized as much as possible. Such identity must be flexible enough to ensure a user can achieve her desired goal by leveraging the minimum amount of identity data.

5) SUSTAINABILITY PROPERTY

The properties that are crucial in maintaining the sustainability of self-sovereign identity is listed under this category. The properties are presented in Figure 12 and described next.

- **Transparency**. A self-sovereign identity and its system must be transparent enough for every involved entity. A user should be well aware about all her partial identities and their corresponding interactions. The system and the corresponding algorithm must allow an easy retrieval of such interaction to ensure transparency. Another way to achieve it is to ensure that the system is fully open source, allowing anyone to examine its internal mechanism and algorithms. This will enable finding bugs within the system as well as ensure to be sustainable with a wider participation of members from the open source communities.
- **Standard**. A self-sovereign identity must be based on open standards to ensure maximum portability, interoperability and adoption as well as sustainability.
- **Cost**. The cost to create, manage and adopt a self-sovereign identity must be as minimum as possible. Otherwise it will create an unnecessary hindrance towards its wide-scale adoption.

C. FORMAL DEFINITION

In the traditional identity management setting, a user's partial identity is merely defined from the perspective of the provider and hence is only valid within the domain of that provider. Users do not own their partial identities and they do not have any provision to create a partial identity without relying on a provider. Once a provider ceases to exist, so does all the partial identities of all users in its domain. These go against the notion of all the foundational properties (such as Existence, Autonomy, Ownership and Access) of a self-sovereign identity.

This notion is captured in the Digital Identity Model (DIM) as presented in Section 3, where the symbol d denotes the domain in which a partial identity is valid. Each partial identity of a user is locked to a provider using the corresponding identifier (i_d) in d. The total identity of a user in DIM is defined with a collection of different partial identities in different provider-centric domains. It is evident from this notion that the coupling of a user identity on a specific domain governed by a provider dictates its reliance on the corresponding provider. Hence, modeling an identity coupled with a domain which is not governed by a specific provider would be the first step to disrupt the current notion of the provider-centric identity.

Our first step towards this goal is to present the concept of *decentralized domain* which is defined in the following way.

Definition 4: A decentralized domain is an application domain which is not controlled and governed by a single entity. In such a domain, any entity, including a user or a provider, can participate and engage in activities autonomously without relying on a specific entity (provider).

We use the notation D^{dec} to denote the set of decentralized domains and d^{dec} to denote a single decentralized domain such that $d^{dec} \in D^{dec}$.

Intuitively, a decentralized domain can continue to exist even after an entity, e.g. a provider, ceases to exist.

Next, we define a partial identity of a user u within a decentralized domain in the similar fashion as DIM, following the same condition as defined previously:

$$parIdent^{u}_{d^{dec}} = \{(i_{d^{dec}}, v_{i_{d^{dec}}}), (a_1, v_1), (a_2, v_2), (a_3, v_3), \ldots, (a_n, v_n), \ldots\}$$

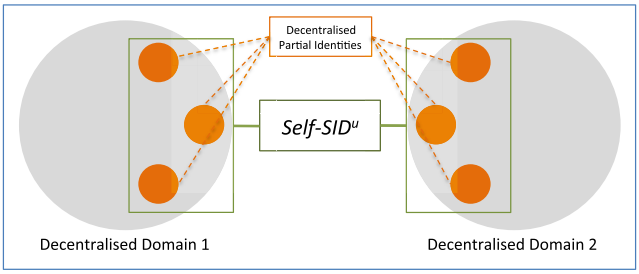

FIGURE 13. Graphical representation of idealized notion of self-sovereign identity.

There is one significant difference between a partial identity in a traditional setting and in a decentralized domain that is worth highlighting. The number of attributes in a traditional domain is generally defined by the provider. A user can only provide values (or values are assigned to them) to the defined attributes and cannot create any new attribute. However, with the absence of any control by a provider, a user can define as many attributes as the user wishes for a partial identity in a decentralized domain, as indicated by the "..." symbol above.

Any user can create as many partial identities as required in a decentralized domain, similar to the traditional domain. For a user u, different partial identities in d^{dec} can be denoted in the following way:

$$parIdent^{u_1}_{d^{dec}}, parIdent^{u_2}_{d^{dec}}, parIdent^{u_3}_{d^{dec}}, \ldots.$$

Similarly, there can be many decentralized domains and a user can have multiple partial identities in multiple decentralized domains. For a user u, different partial identities in multiple domains (d^{dec1}, d^{dec2}, $d^{dec3} \in D^{dec}$) can be denoted in the following way:

$$parIdent^{u}_{d^{dec1}}, parIdent^{u}_{d^{dec2}} and parIdent^{u}_{d^{dec3}}$$

Based on this concept, we can define a self-sovereign identity, denoted as $Self - SID$ in the following manner for a user u:

$$Self - SID^u = \bigcup \{(d, parIdent^{u_i}_d) | d \in D^{dec}, u_i \in U \text{ and } i \in Z^+\}$$

In other words, a self-sovereign identity of a user is the union of her different partial identities in each of different decentralized domains. A graphical representation of this identity is illustrated in Figure 13.

Now, we regard this notion of self-sovereign identity as the idealized notion where all her partial identities reside in decentralized domains. However, in reality, a user will have other partial identities in other provider-centric domains, e.g. in different trusted providers such as government, financial institutions and so on. Therefore, a practical notion a self-sovereign identity can be defined in the following way:

$$Self - SID^u_{prac} = Self - SID^u \bigcup ident^u$$

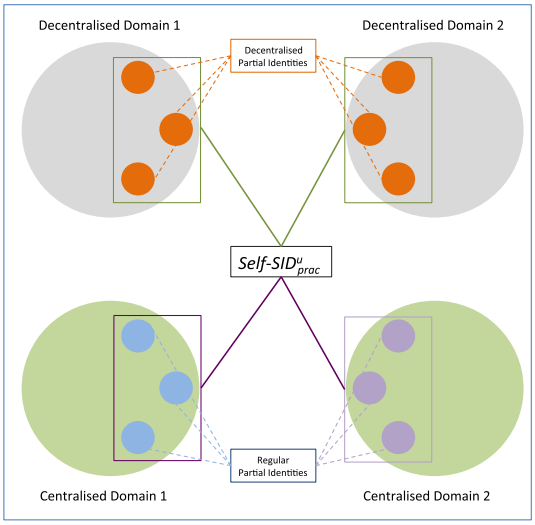

FIGURE 14. Graphical representation of practical notion of self-sovereign identity.

That is, the practical model (Figure 14) of a self-sovereign identity consists of partial identities in decentralized domains as well as partial identities from other provider centric domains.

Having partial identities in such provider-centric domains means that a user cannot exercise a few fundamental principles, such as autonomy, ownership, access and single source, of self-sovereign identity. Therefore, to enable self-sovereignty over those partial identities, we propose two alternatives:

- The trusted (as well as untrusted) providers transfer all partial identities of their user-bases from their provider-centric domains into decentralized domains so that the highlighted principles can be exercised.

The interaction tie between the SP and these providers is severed. Instead, the user acts as the mediator between the providers and the SP via their self-sovereign partial identities.

The first option would be the most deserved one as it would require making minimum changes for any SP to continue to function. In addition, a user can enjoy all the benefits of a self-sovereign identity. However, it is not certain if the providers would be motivated enough to take this option for a variety of reasons: cost involved in making these changes, different security and privacy concerns from their perspectives and so on.

The second option, however, can be deployed with minimum changes in the providers and the SP. For this to happen, only the interaction flows between different entities need to be modified so that the interaction between the SP and providers can only be mediated by a user leveraging a self-sovereign identity management system which in turn utilizes self-sovereign partial identities of the user. Re-establishing the tie between the providers and the SP via the self-sovereign partial identities of users will ensure that users can exercise all the fundamental principles of a self-sovereign identity, even though some identity data of the users are still

stored in the providers. This is why this option might be the most practical solution.

Next, we explore how we can model the second interaction.

As stated in DIM, it follows that the concept of profile is utilized for releasing identity data while engaging in different interactions. Here, we re-define the concept of profile for self-sovereign identity in the following way.

Definition 5: A profile in a self-sovereign identity is the union of different assertions (verifiable claims).

An assertion is defined in the following way:

Definition 6: An assertion is a signed claim bound to an entity. It can be signed by the entity itself, creating the notion of a self-asserted claim. Alternatively, it can be signed by the provider, acting as the trusted third party, creating a provider-asserted claim.

We use the notation $assrtn_{p_d}^{u_{d^{dec}}}$ to denote an assertion consisting of a claim $c^{u_{d^{dec}}}$ regarding a user u in $d^{dec} \in D^{dec}$. The assertion is formally defined as a signed claim in the following way, where K_{p_d} represents the public key of the provider p of domain d and $K_{p_d}^{-1}$ represents the corresponding private key:

$$assrtn_{p_d}^{u_{d^{dec}}} = \{c^{u_{d^{dec}}}\}_{K_{p_d}^{-1}}$$

A self-asserted assertion would then be denoted as $assertn_{u_d}^{u_d}$. However, for brevity, we would simple write: $assertion^{u_d}$.

Based on this, we mathematically define a profile in the following way:

$$profile_{d^{dec}}^{u} = \left\{ \bigcup assrtn_{p_d}^{u_{d^{dec}}} | d \in Dandd^{dec} \in D^{dec} \right\} \bigcup assrtn^{u_{d^{dec}}}$$

That is, a profile consists of several provider-asserted claims and a self-asserted claim prepared by the user for an SP for a particular session. Since a claim inside an assertion can consist of several attribute name-value pairs, a profile can encode as many as required.

Next, we analyze how the proposed model mostly satisfies the foundational as well as security properties of self-sovereign identity.

- **Existence**. The model intuitively encodes the digital representation of a user in the form of attribute name-value pairs. This representation can be used to define a self-sovereign profile which ultimately is released, via a protocol, to an SP to imply her existence in a domain.
- **Autonomy**. The model leverages decentralized domains where no singly party can assert its control. This enables a user to create/manage as many self-sovereign partial identities as required without relying on a specific provider, thereby satisfying the autonomy property.
- **Ownership**. Once a partial identity is created in a decentralized domain, only the creator of the identity can claim ownership. In addition, the model enforces that assertions from the providers are released by the owner via the profile which is created and released only by the owner.
- **Access**. As long as the corresponding decentralized domain exists, a user can access her identity data without any hindrance or interference by any other entity.
- **Single source.**If any provider-asserted claim is only released via a profile leveraged by a self-sovereign identity as advocated previously, the user can act as the single source of truth regarding her identity.
- **Protection**. This property will mainly depend on how a decentralized domain is deployed and what protection mechanisms it supports. A decentralized domain which has better protection support over other domain will have a better probability for wide-scale adoption.
- **Availability**. Any identity data in a decentralized domain will be readily available and accessible as long as such domain exists.
- **Persistence**. Similarly, data in a decentralized domain will be persistent as long as the domain exists.

It is to be noted that the remaining properties of a self-sovereign identity depends on the system which is used to manage such an identity. We would like to highlight the difference between identity and an identity management system: an identity is a representation of a user in a domain whereas an identity management system is the tool by which a user manages her identity. This point was mostly overlooked while presenting definitions in the existing literature where the properties belonging to the remaining categories are also regarded as the property of a self-sovereign identity.

D. LAWS OF (SELF-SOVEREIGN) IDENTITY

In the quest to introduce an Identity meta-system, Kim Cameron from Microsoft introduced the Laws of Identity in 2005 [2]. The laws consisted of seven principles that encoded several guidelines how the identity of a user should be handled and released as well as how different entities can be identified using different types of identifiers. The laws are briefly presented next.

- **Law 1: User Control and Consent.** "*Technical identity systems must only reveal information identifying a user with the user's consent.*"
- **Law 2: Minimal Disclosure for a Constrained Use.***"The solution that discloses the least amount of identifying information and best limits its use is the most stable long-term solution.*"
- **Law 3: Justifiable Parties.** "*Digital identity systems must be designed so the disclosure of identifying information is limited to parties having a necessary and justifiable place in a given identity relationship.*"
- **Law 4: Directed Identity.** "*A universal identity system must support both "omni-directional" identifiers for use by public entities and "unidirectional" identifiers for use by private entities, thus facilitating discovery while preventing unnecessary release of correlation handles.*"

- **Law 5: Pluralism of Operators and Technologies.**"*A universal identity system must channel and enable the inter-working of multiple identity technologies run by multiple identity providers.*"
- **Law 6: Human Integration**. "*The universal identity metasystem must define the human user to be a component of the distributed system integrated through unambiguous human-machine communication mechanisms offering protection against identity attacks.*"
- **Law 7: Consistent Experience across Contexts:**"*The unifying identity metasystem must guarantee its users a simple, consistent experience while enabling separation of contexts through multiple operators and technologies.*"

These seven laws were introduced to formalize, for the first time, different aspects of a digital identity. However, in reality, only one of them (Law 4: Directed Identity) divulges in anything directly related to an identity. All others mostly provide guidelines for a user-centric identity management system. As stated before, we would like to draw a line of distinction between an identity and identity management system regarding their properties. In light of this, we argue that the Laws of Identity are mostly applicable to an IMS, rather than to the concept of identity itself. This essentially means that the laws exclusively related to the concept of identity are still at large.

Aligning to the focus of this article, we restrict ourselves from venturing the path of defining laws for identity. Instead, what we attempt next is to outline the principles of self-sovereign identity - by transforming the foundational and security properties into laws.

- **Law of Existence**. *A self-sovereign identity must encode a quasi-representation of one's self in digital form to assert her existence in a decentralized digital domain.*
- **Law of Autonomy**. *The administration and management of a self-sovereign identity must be fully autonomous, free from the reliance on any party.*
- **Law of Ownership**. *A user must be the ultimate owner of its self-sovereign identity.*
- **Law of Access**. *A user must have unrestricted access to her self-sovereign identity. Only a user (or her delegated agent) can control the access to any information regarding her self-sovereign identity.*
- **Law of Single Source.***A user should be the single source of truth regarding her self-sovereign identity.*
- **Law of Protection**. *A self-sovereign identity should be well protected with latest cryptographic mechanisms satisfying the CIA (Confidentiality, Integrity and Authenticity) and non-repudiation properties.*
- **Law of Availability**. *A self-sovereign identity must be always available and accessible from any platform when required by its owner.*
- **Law of Persistence**. *A self-sovereign identity must be persistent, at least as long as it is required by its owner.*

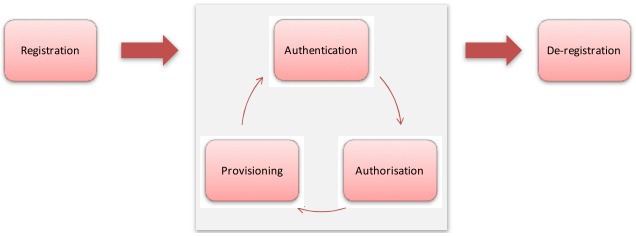

FIGURE 15. Life-cycle of an identity management system.

VI. LIFE-CYCLE OF (SELF-SOVEREIGN) IDENTITY

An Identity Management System consists of five steps, collectively known as the life-cycle of an IMS (Figure 15): Registration, Authentication, Authorisation, Provisioning & De-registration. Among these the registration and de-registrations processes are one-off processes whereas the other three steps are repeated as many times are required during the service provisioning stage. Each of these steps is presented below.

A. REGISTRATION

Registration is the initial step in which a user registers herself at the decentralized domain by creating/providing unique values for the identifier and its corresponding credential. We denote the registration process using *REG* and define it mathematically in the following manner, based on the registration process defined in [5]:

Definition 7: Let $REG_{d^{dec}} : \left(\{i_{d^{dec}}\} \times \{av^i\}\right) \times \left(\{c_{d^{dec}}\} \times \{av^c\}\right) \rightarrow \{parIdent^u_{d^{dec}}\}$ *be the function that upon providing values for the identifier and the corresponding credential creates a new partial identity of a user u in the decentralized domain* d^{dec}.

In this definition:

- $i_{d^{dec}}$ denotes the identifier of d^{dec}
- av^i denotes the created/provided value for $i_{d^{dec}}$ such that $av^i \in AV_{d^{dec}}$
- $c_{d^{dec}}$ denotes the corresponding credential and av^c represents the provided/created value of the credential where $av^c \in AV_{d^{dec}}$
- $parIdent_{d^{dec}}$ represents the set of partial identities in d^{dec}

Registering a new partial identity for a user extends the set of users and attribute values in d^{dec} in the following way:

$$U'_{d^{dec}} = U_{d^{dec}} \bigcup \{u\}$$
$$AV'_{d^{dec}} = AV_{d^{dec}} \bigcup \{av^i\} \bigcup \{av^c\}$$

B. DE-REGISTRATION

The final step is the de-registration process which allows a user to de-register from a decentralized domain by removing the association between the user and the identifier in the corresponding domain. It is the reverse process of registration in which the set of users and attributes valued are updated in the following way:

$$AV'_{d^{dec}} = AV_{d^{dec}} \setminus \{\{av^i\} \bigcup \{av^c\}$$
$$U'_{d^{dec}} = U_{d^{dec}} \{u\}$$

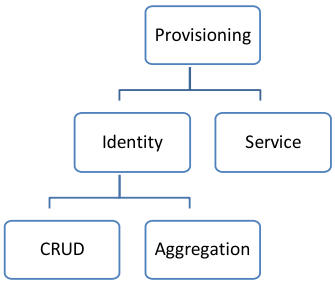

FIGURE 16. Taxonomy of provisioning.

Here, $U'_{d^{dec}}$ and $AV'_{d^{dec}}$ represent the updated sets once the user u is de-registered.

C. AUTHENTICATION

Before any service can be accessed, the user needs to be identified and authenticated. Identification is the process of finding an association between an identifier value and the user whereas authentication is the process of proving the association by providing the corresponding credential value. In many systems, the process of identification and authentication are combined together in a single step.

D. AUTHORISATION

Authorisation is the process to decide if an entity can perform a certain action on a specific resource in a specific domain based on the identifier value along with other attribute values. The authorization usually takes place before the provisioning phase (described below). It usually takes place in an SP where it checks if a user can access the requested service/resources by having certain attribute values. However, it might take place in an IdP belonging to a decentralized domain.

E. PROVISIONNING

Once a user is identified, authenticated and authorized, she moves to the provisioning phase. We differentiate between two different types of provisioning (Figure 16): identity provisioning and service provisioning. Each of these is described below.

1) IDENTITY PROVISIONNING

Identity provisioning represents the step where a user can create and update her attributes stored in an IdP. We consider two different types of identity provisioning as presented below.

- CRUD provisioning: CRUD provisioning represents the create, read, update and delete operations involving identity attributes.
- Aggregation provisioning: Aggregation provisioning represents the aggregation operation involving identity attributes by which a profile is created for a particular session.

2) SERVICE PROVISIONNING

On the other hand, service provisioning represents the phase of accessing an online service, provided by an SP. For this, a user releases a profile to the SP using an identity protocol. The SP extracts attributes embedded in the assertions within the profile and takes an authorization decision to grant/deny access to the requested service.

VII. BLOCKCHAIN

Bitcoin [20], introduced in 2009, has emerged as the world's first widely used digital currency and has been used in a wide range of applications. It is underpinned by a novel mechanism called Distributed Ledger Technology (DLT), also known as blockchain technology, providing its solid technical foundation. Even though the terms blockchain and DLT are used inter-changeably in the literature, there is a subtle difference between them which is worth highlighting. A blockchain is just an example of a particular type of ledger where data can be stored in a specific format. There are other types of ledger with different data formats. When a ledger (including a blockchain) is distributed across a network, it can be regarded as a Distributed Ledger or simply a ledger.

In recent years, blockchain has received widespread attention among the industry, the Government and academia. It is regarded as one of the fundamental technologies to revolutionize the landscapes of several application domains. At the center of DLT is the ledger itself. A distributed ledger or a blockchain is a ledger consisting of consecutive blocks chained together following a strict set of rules. The ledger is distributed and stored by the nodes of a P2P network where each block is created at a predefined interval in a decentralized fashion by means of a consensus algorithm. The consensus algorithm guarantees several data integrity related properties (discussed below) in the ledger.

Evolving from the Bitcoin ledger, a new breed of ledger has emerged which facilitates the deployment and execution of computer programs, known as *smart-contracts*, on top of the respective ledger. Such smart-contracts enable the creation of so-called *de-centralized applications*(DApps), which are autonomous programs operating without relying on any system entity. Being part of the ledger makes smart-contracts and their executions *immutable*and *irreversible*, a sought-after property having a wide-range of applications in different domains.

A. BLOCKCHAIN PROPERTIES

A blockchain exhibits several properties that make it a suitable candidate for several application domains. The properties are summarized below.

- **Distributed consensus on the ledger state**: One of the crucial properties of any distributed ledger is its capability to achieve a distributed consensus on the state of the ledger (in other words the order of the blocks in the chain) without being reliant on any Trusted Third Party (TTP). This opens up the door of opportunities to build and utilize a system where every possible state and interaction are verifiable by any authorized entities.
- **Immutability and irreversibility of ledger state**: Achieving a distributed consensus with the participation of a large number of nodes ensures that the ledger

state becomes practically immutable and irreversible after a certain period of time. This also applies to smart-contracts which enable the deployment and execution of immutable computer programs.

- **Data (transaction) persistence**: Data in a distributed ledger is stored in a distributed fashion ensuring its persistency as long as there are participating nodes in the P2P network.
- **Data provenance**: The data storage process in any distributed ledger is facilitated by means of a mechanism called the transaction. Every transaction needs to be digitally signed using public key cryptography which ensures the authenticity of the source of data. Combining this with the immutability and irreversibility of a distributed ledger provides a strong non-repudiation instrument for any data in the ledger.
- **Distributed data control**: A distributed ledger ensures that data stored in the ledger or retrieved from the ledger can be carried out in a distributed manner that exhibits no single point of failure.
- **Accountability and transparency:** Since the state of the ledger, along with every single interaction among participating entities, can be verified by any authorized entity, it promotes accountability and transparency.

B. TYPES OF BLOCKCHAIN

Depending on the applications domains, different blockchain deployment strategies can be pursued. Based on these strategies there are two predominate types of ledger: *Public* and *Private*. These are discussed below:

- **Public blockchain**, also known as the *unpermissioned* blockchain, allows anyone to create and validate blocks as well as to modify the ledger state by storing and updating data by means of transactions among participating entities. This means that the ledger state and its transactions along with the data stored is transparent and accessible to everyone. This raises privacy concerns for particular scenarios where the privacy of such data needs to be preserved.
- **Private blockchain**, also known as the *permissioned* blockchain, can be restricted unlike its public counterpart in the sense that only authorized and trusted entities can participate in the activities within the ledger. By allowing only authorized entities to participate in activities within the ledger, a private ledger can ensure the privacy of ledger data, which might be desirable in some use-cases.

C. IDENTITIES IN BLOCKCHAIN SYSTEMS

Different types of blockchain systems utilize different types of identities. We explore how identities are represented in different types.

In public blockchain systems, a new identifier is created using a public key from a corresponding public-private key pair. For this, at first, a private-public key pair is generated and then, a new identifier is created from a cryptographic hash of the corresponding public key. Different systems use different types of hashes. For example, Bitcoin creates a 25 bytes long address (representing the identifier) by double hashing the corresponding public key using the SHA-256 and RipeMD-160 respectively and then adding a protocol byte and a checksum of 4 bytes. On the other hand, Ethereum (a smart-contract empowered public blockchain platform[2]) creates a 21 byte address (identifier) from a public key by hashing it using the Keccak-256 (colloquially also known as SHA3-256) algorithm to generate a 32 byte hash. From these 32 bytes, last 20 bytes are taken and then prefixed with '0x' (1 byte) to create a 21 byte address. For both Bitcoin and Ethereum, these addresses (identifiers) represent an identity. A user can create as many identities as required by generating addresses following the steps discussed above. Each such address is protected using a password which is used to encrypt the private key. When a user wishes to interact using a particular address (identity), she needs to provide the password to decrypt the corresponding private key which is used to sign a transaction.

Similarly, different techniques are deployed to represent an identity in different private blockchain systems. For example, in Hyperledger Fabric (a smart-contract empowered private blockchain platform[3]), identities are represented using a certificate consisting of a set of attributes (including an identifier as well as a public key) which is accompanied by a corresponding password (credential) and private key. Users have no option to create their own identities as they are managed by the admin of the Hyperledger network. For other systems, how identities will be represented will depend on their respective method.

D. BLOCKCHAIN AND SELF-SOVEREIGN IDENTITY

Interestingly, blockchain exhibits several properties which coincides with some desirable properties of a self-sovereign identity. For example, blockchain essentially provides a decentralized domain which is not controlled by any single entity. Data stored in any blockchain is readily available (availability property) to any authorized entity (access property). An owner of a particular data (an identity data such as Personally Identifiable Information or PII) has full control over it and dictates how such data can be shared with other users within the blockchain domain, thereby satisfying the disclosure property. Furthermore, such a platform a smart-contract supported blockchain can allow users to deploy an immutable-autonomous program via a smart-contract which could be leveraged to create a user-controlled IdP coupled with a fine-grained access control mechanism to control access and share of such data. In addition to these, a blockchain system can support a few additional advantages in terms of data immutability, provenance, distributed control, accountability and transparency in such a

[2]https://www.ethereum.org/

[3]https://www.hyperledger.org/projects/fabric

way that none of the traditional system can provide. Because of this, we envision that a blockchain system can provide a solid foundation upon which we can deploy an identity management system that supports self-sovereign identity at its core.

E. BLOCKCHAIN-BASED IDENTITIES

Because of its support for self-sovereign identity, blockchain platforms have already been exploited to develop self-sovereign identity applications. These applications have been deployed at the top (application) layer where a blockchain platform resides underneath. A few examples of such applications are uPort,[4] Jolocom,[5] Sovrin[6] and Blockcerts.[7] Next, we explore the functionalities of these applications and investigate if satisfy different properties of a self-sovereign identity.

uPort is a decentralized identity system built on top of Ethereum platform supporting the notion of self-sovereign identity [21]. It consists of a mobile App and several Ethereum smart contracts including a public registry of uPort identity. A user utilizes the respective mobile App to create, update and share identity information with other users. In the backend, such data is controlled by different smart contracts. The bulk of identity data is stored on IPFS[8] (Interplanetary File System, a distributed file system) whereas the mobile App is used to store the corresponding private key of a uPort identity. The public registry is used to create a correlation between a uPort identity and its corresponding IPFS data.

Jolocom [22] is another self-sovereign identity system whose functionality is surprisingly similar to uPort. Like uPort, it is developed on top of Ethereum and consists of several Ethereum smart contracts including a registry smart contract. Users utilize a mobile App, similar to uPort, to interact, create, manage and share their identities. The only differentiating factor of a Jolocom identity from any uPort is the way identity data is structured and represented which is not explored any further.

Sovrin foundation[6] is private non-profit entity whose goal is to facilitate and promote the notion of self-sovereign identity. Within this goal, it has developed Sovrin Identity system [23] which utilizes its own blockchain called *Sovrin ledger* that leverages a novel consensus algorithm called Plenum. A user can utilize a mobile App or a web site which acts as a Sovrin client to interact with the ledger in order to create, update, manage and share their identity data. Sovrin also supports the notion of *Agents* which can act as a Trusted Third Party to vouch or certify for identity data of a user. With the support of verifiable claims, Sovrin allows users to claim not only about themselves but also about another user/organization. It also enables users to exercise control in a way so that they can choose exact the data they want to share with someone else. In a whole, Sovrin promises a lot, however, it is still in the development cycle with very limited release. Thus, it is yet to be seen if it can fulfil all its promises.

Blockcerts is a tool to store and verify the cryptographic hash of any digital certificate using blockchain [24]. A user, upon receiving a certificate (e.g. a degree certificate from a university), regarding herself, from an entity, can store its hash in the blockchain. Once a verifier (e.g. an employer) receives the certificate from the user for a particular application (e.g. job application), it can utilize the respective verification tool to verify the integrity of the certificate by comparing its stored hash in the blockchain. Currently, Blockcerts is agnostic about certificate type and can utilize any of Bitcoin or Ethereum to store the respective hash. However, unlike uPort, Jolocom and Sovrin, Blockcerts is not a full-fledged self-sovereign identity system.

Next, we analyze if these systems satisfy different properties of self-sovereign identity by consulting with their corresponding whitepapers and technical documents. The result of our analysis is presented in Table I.We have used the '√' symbol to indicate if a certain property is satisfied by the corresponding systems and the '?' symbol to indicate that we have not found any information regarding this. Finally, the '−' symbol is used to imply that the respective is not applicable for the particular system.

TABLE 1. Comparison of the Selected Systems

		uPort	Jolo	Sovrin	Blockcerts
Foundatinal	Existence	√	√	√	-
	Autonomy	?	?	√	-
	Ownership	√	√	√	-
	Access	√	√	√	-
	Single Source	?	?	?	-
Security	Protection	√	√	√	-
	Availability	√	√	√	√
	Persistence	√	√	√	√
Controllabili	Choosability	√	?	√	-
	Disclosure	√	?	√	?
	Consent	?	?	√	?
Flexibility	Portability	?	?	?	?
	Interoperability	?	?	?	?
	Minimisation	?	?	?	?
Sustainbility	Transparency	√	√	√	√
	Standard	√	√	√	√
	Cost	√	√	√	√

[4]https://www.uport.me
[5]https://jolocom.io
[6]https://sovrin.org
[7]https://www.blockcerts.org
[8]https://ipfs.io

As evident in the table, most of the systems (except Blockcerts) satisfy several properties. However, it is still not clear if these systems satisfy or will satisfy other properties in future. Also, it is understandable that Blockcerts, not being a full-fledged self-sovereign identity systems, do not satisfy many properties or even such properties do not apply to it. Next, we provide further explanations to justify our analysis for some properties.

uPort currently allows to create only one identity. Therefore, a user cannot create as identities as required. Therefore, we conclude that it does not support full autonomy. It is not clear if uPort, Jolocom and Sovrin will support the notion of aggregation. Without this, they cannot act as the single source for the respective user. That is why we conclude that these systems do no satisfy this property. Similarly, it is not explicitly specified if an explicit consent is required to disclose attributes in these systems, resulting

Even though Sovrin claims to support portability, however it must be noted that Sovrin has proposed a standard to represent an identity. Their claim is justified if their proposal is standardizes across all systems. Otherwise, Sovrin will lose its portability feature as soon as their corresponding ledger ceases to exist. It is not clear, as of now, whether the presented systems are compatible with existing identity management standards/systems such as SAML and/or OpenID or even with other self-sovereign identity systems.

None of the systems seems to have considered the issue of data minimization. Because of its usage of blockchains, each of these systems are transparent in nature. In addition, they either use an open standard or have proposed an open standard for self-sovereign identity. An important factor is cost, since all systems incur cost to create transactions or store data in their respective blockchain platforms. Depending on the incurred cost, it might create additional barrier for any wide-scale adoption.

VIII. USE-CASES

In this section, we explore several use cases that illustrate how a blockchain empowered identity management system supporting self-sovereign identity can be utilized in different life-cycle activities of identity management. At first, we assume the following additional functionalities of an IdP and SP in the setting of self-sovereign Identity.

- **Identity Provider (IdP):**We envision that an IdP provides an interface to any user via an IMS that allows the user to engage in activities involving the life-cycle of an IMS as presented below. In this regard, we envision two different types of IdPs. The first type is deployed and managed by different governmental and business organizations, either for providing governmental services to the respective citizens or for providing business services to their customers respectively. Such an IdP can be deployed in a traditional centralized domain or in a decentralized domain within the deployed blockchain system. An IdP deployed in a decentralized domain represents the decentralized analogue of any traditional IdP in the current setting as users must use the corresponding IMS to create, access and manage any identity within this IdP. For this, users have less control over their identity data stored in such IdPs. To counteract this problem, the notion of **Personal IdP** has been put forward [25]. A Personal IdP represents a user-controlled IdP which stores identity data on the user's behalf so that she can engage in the life-cycle activities of an IMS. We envision that this is the IdP that will be used to capture the notions of a self-sovereign identity. Such an IdP will leverage an IMS by which a user can participate in the identity related life-cycle activities and can satisfy all the properties of a self-sovereign identity. Even though such an IdP does not represent a business organization, however, a business use case can be sketched that facilitates the development and release of such an IdP. Then, a user can leverage such an IdP in exchange of a small fee.
- **Service Provider (SP):**An SP in the setting of self-sovereign identity has similar functionalities of a traditional SP with just one restriction: an SP can receive a profile only from a user-controlled IdP as mediated by the corresponding user. This is to ensure that the use can exercise the control required to satisfy several properties of a self-sovereign identity.

In addition, the use cases are based on the following assumptions:

- A smart contract blockchain system has already been deployed providing the capability of a decentralized domain.
- Both types of IdPs provide dedicated interfaces for interactions via an App or via a web browser. A user can interact with any type of IdPs using a multitude of devices such as smart phones, tablets, PCs, tablets or other smart devices.

To build up our use case story, we consider one user Alice (denoted as *Alice*), one governmental organization (denoted as *Gov*) of a country, one financial institution (denoted as *Bank*) and a service provider (denoted as *SP*). We also consider that *Gov* has been deployed in a centralized domain whereas *Bank* has been deployed in a decentralized domain.

A. REGISTRATION

The registration procedure will depend on two factors: type of IdP & type of domain. Each of these factors is explored separately.

1) REGISTRATION AT PUBLIC IDPS

At first, we explore the registration procedure at Public IdPs considering both centralized as well as decentralized domains.

- **Within a centralized domain**: This represents the traditional setting of Identity Management and as such the corresponding registration procedure is similar to what

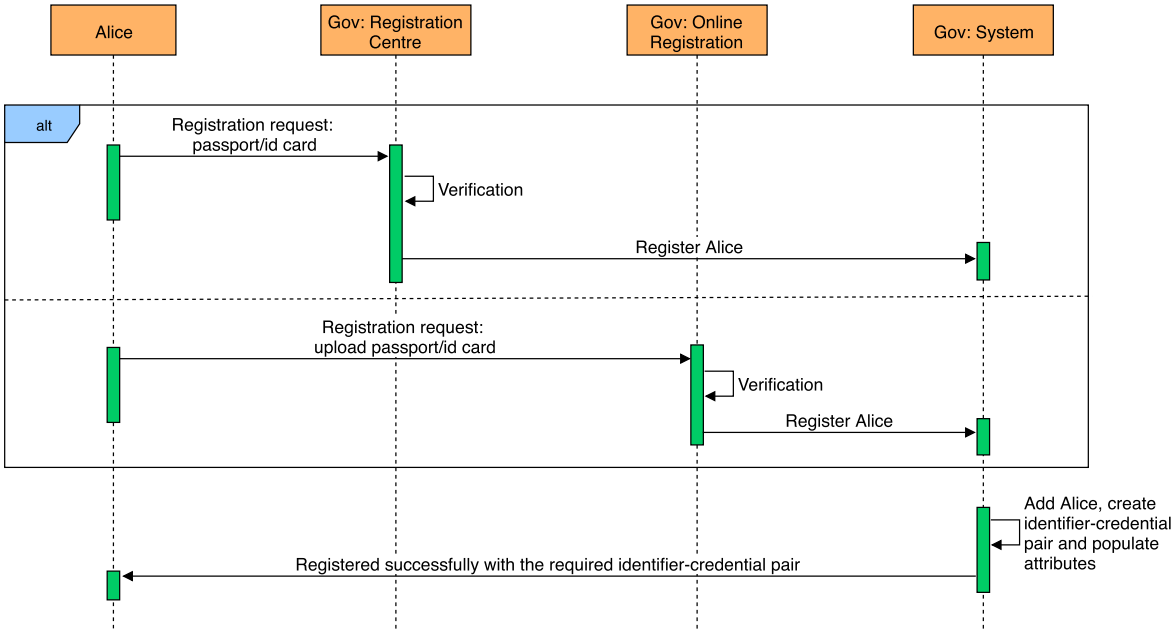

FIGURE 17. Registration at public IdPs in centralized domains.

is carried out in most current scenarios. This use case is illustrated in Figure 17 where we assume that Alice wants to register at *Gov*. The registration process might involve a verification process which can be done offline and online. An offline verification process might require Alice to provide a physical document (e.g. passport, identity card and so on) as a claim regarding the citizenship/residency of Alice for that particular country to a registration center. On the other hand, an online verification might involve uploading the scanned cope of the required physical document to an online service which then can be verified either via an advanced image analysis mechanism or via human inspections. Upon verifying the claim, Alice is either provided with a newly created identifier (e.g. username) value and its corresponding credential (e.g. password) or she is given the opportunity to choose her own values for the identifier and its corresponding credential. At the end of the verification and the registration process, values for several crucial attributes are pre-populated with the option to change these values when required, on the condition of another successful verification.

The verification process is itself quite rigorous given the sensitivity and importance of different attributes provided by a governmental service. The registration process might be similar when carried out by a non-government IdP with a more relaxed verification process (e.g. verification via SMS or email).

- **Within a decentralized domain**: The registration procedure in a decentralized domain depends entirely on the underlying blockchain system. If a public IdP (*Bank* as per our use-case) leverages a public blockchain system, it might follow the registration procedure as presented in Figure 18. The public IdP might provide a registration interface accessible via a mobile app or a web browser to generate the required crypto key-pair and the identifier (representing an identity) along with the associated credential. The private key can be stored in encrypted format (using a credential) either in the user's device along with corresponding public key and the identifier. The App stores the association between the identifier and the public in its internal storage which is later used in other use-cases. It is difficult to generalize the Registration procedure when a private blockchain system is used and hence is skipped here.

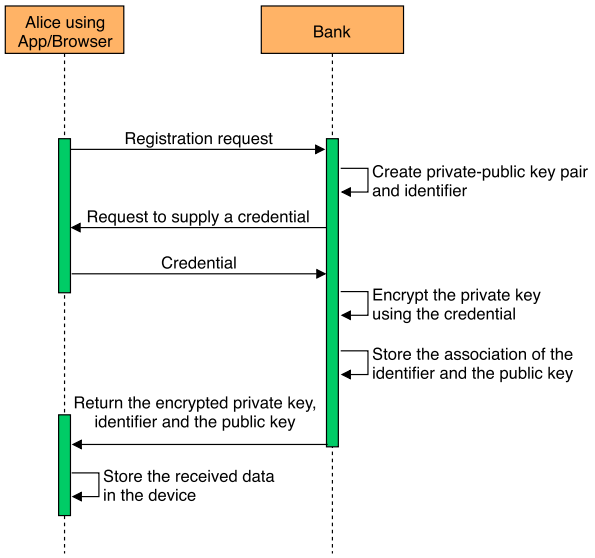

FIGURE 18. Registration at Public IdPs in decentralized domains.

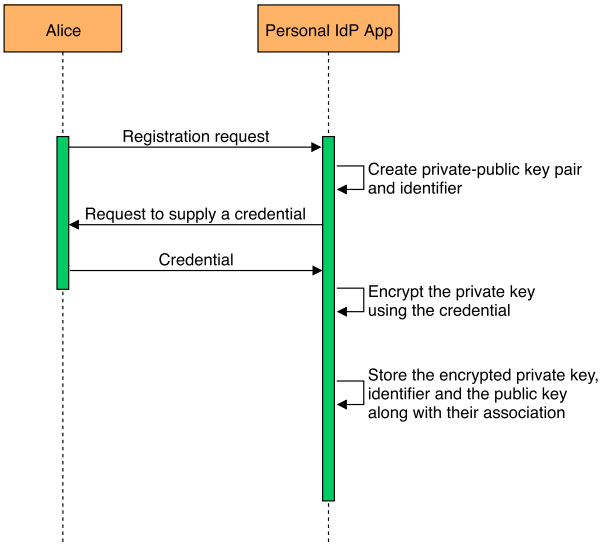

FIGURE 19. Registration at a Personal IdP.

2) REGISTRATION AT PERSONAL IDPS

The registration procedure by a personal IdP is facilitated by a blockchain system and follows the same steps already presented above and is illustrated in Figure 19. It is assumed that the Personal IdP provides an app for smart devices or a legacy application by which a user can initiate the registration process in which a new key-pair is generated which is accompanied by a user supplied credential (password). The public key of this pair is utilized to generate an identifier to represent a newly created respective identity for the user. The private key is stored in the corresponding device in encrypted format, using the credential, whereas the public key can be stored in plain text.

B. DE-REGISTRATION

Similar to the registration procedure, the de-registration procedure will also depend on two factors: type of IdP & type of domain. Each of these factors is explored separately.

1) DE-REGISTRATION AT PUBLIC IDPS

- **Within a centralized domain**: The de-registration process is quite straightforward which can be initiated either by the user or by the respective public IdP. As stated earlier, each partial identity is associated with an identifier, a de-registration process is simply destroying the association between the partial identity and its corresponding identifier. Therefore, once the de-registration procedure is initiated, the public IdP simply deletes any record associating the identifier and the partial identity from its database as well as any other data belonging to that partial identity. The process is illustrated in Figure 20.

2) DE-REGISTRATION AT PERSONAL IDPS

The de-registration process is similar to what has been described for the decentralized domain. For this, the user would simply need to destroy the corresponding key pair and any data associated with respective partial identity.

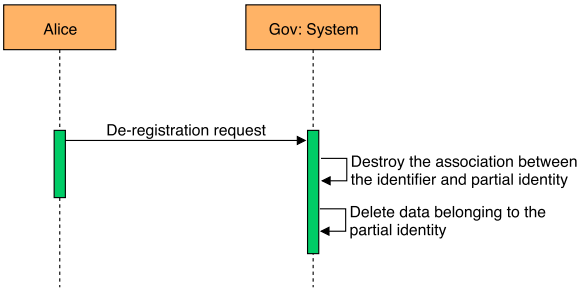

FIGURE 20. De-registration at public IdPs in centralized domains.

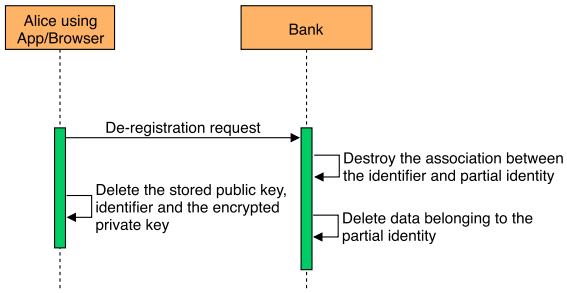

FIGURE 21. De-registration at public IdPs in decentralized domains.

- **Within a decentralized domain**: The de-registration procedure at a decentralized domain will require the destruction of the respective crypto key-pair, the association between the identifier and the partial identity and any data associated with the corresponding partial identity as illustrated in Figure 21.

C. AUTHENTICATION/AUTHORISATION

The authentication process will depend on the domain in which the IdP is deployed. We explore this process separately for each domain.

- **Within a centralized domain**: Any public IdP deployed within a centralized can leverage any traditional mechanism to authenticate a user. The most widely-used mechanism in such setting is to authenticate a user by using a username/email as the identifier and a password as a credential. In recent years, more secure technologies such as OTP (one-time-password), two-factor authentication using either a mobile app or mobile text and biometrics such as fingerprint are increasingly being deployed.
- **Within a decentralized domain**: There could be many ways authentication be done within a decentralized domain. Here, we highlight one potential flow with the assumption that Alice utilizes the corresponding App to participate in this flow. We also assume that a digital signature based protocol is devised to allow a user to authenticate herself by simply digitally signing a data with her corresponding private key. A probable protocol flow is illustrated in Figure 22 and is presented

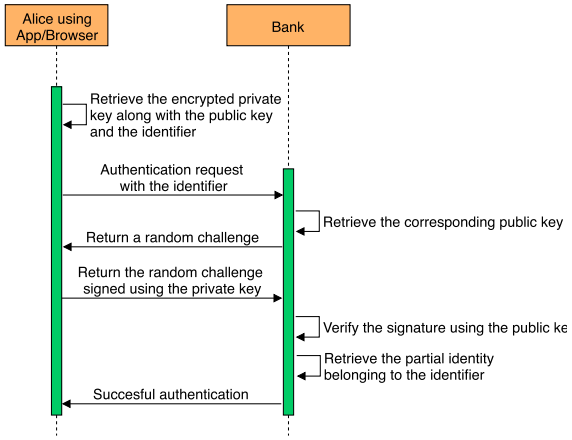

FIGURE 22. Authentication in decentralized domains.

next. Alice utilizes the App to retrieve the encrypted private key, the public key and the identifier. Then, the encrypted private key is decrypted using the credential. Then, Alice submits an authentication request to *Bank* with the identifier. *Bank* retrieves the corresponding public key associated with the identifier and generates a random challenge which is returned to the App. App creates a digital signature with the random challenge and the private key and returns it back to *Bank*. A successful verification of the digital signature signifies that Alice is authenticated using the particular public key and the identifier. Bank, then, can retrieve the partial identity data for Alice using the authenticated identifier. It is to be noted that even though this particular flow has been illustrated using a mobile App, it could be deployable using a browser as well.

The authorization process can take place either in the IdP or in the SP. Within an IdP, an authorization process is activated once a user is authenticated and then the user proceeds to the provisioning phase. For each provisioning request, the user is verified to check if the user is authorized to perform the requested provisioning action. There are many existing authorization mechanisms, such as Access Control List (ACL) [26], Role Based Access Control (RBAC) [27] or Attribute Based Access Control (ABAC) [28], that an IdP can deploy both within the centralized and decentralized domains.

Since we mainly focus on the activities within the IdP, we skip any discussion regarding authorization at the SP.

D. IDENTITY PROVISIONING

We explore two different identity provisioning separately below.

1) CRUD PROVISIONING

Like before, CRUD provisioning can be presented for two different domains with the assumption that a user has already been authenticated and authorized to perform any CRUD action. Within a centralized domain, the respective IdP provides a UI which allows the user to perform a CRUD operation with respect to new attributes as well as existing attributes. On the other hand, within a decentralized domain, the respective IdP provides a UI via a web interface or an App to allow user to perform a CRUD operation with respect to new as well as existing attributes.

2) AGGREGATION PROVISIONING

Traditional IdPs within the centralized domain generally do not support any attribute aggregation mechanism. However, there have been works that have explored how attribute aggregation can be achieved within the federated domain [29], [30]. Nevertheless, we do not explore them any further than the assumption that there are some centralized public IdPs which provide interfaces (e.g. via an API) to generate and supply assertions to other IdPs in a secure way. Next, we explore aggregation provisioning within a decentralized domain only.

- **Within a decentralized domain**: Aggregation within a decentralized domain can be facilitated both by a public IdP and a personal IdP. In any case, the IdP must provide a UI for the user to initiate and complete the aggregation process which includes creating assertions within its domain as well as interacting with other IdPs to request and receive assertions from them. Even though the aggregation can be carried out both in a public and personal IdP, we recommend it to be carried out in a personal IdP to safeguard against any privacy issues. This will ensure that the user has the full control over the aggregated attributes and a public IdP will not have the provision to build a profile of a user in the background without the user's knowledge and consent. A potential protocol flow for this use-case is illustrated in Figure 23 and presented next. Alice utilizes the personal IdP App. At the aggregation interface of the personal IdP, she can initiate interactions with each corresponding IdP to retrieve the required assertion.

For each interaction, the user might need to go through the authentication process if the user is not already authenticated. In addition, the user also needs to engage with the assertion creation and release process in each interaction with the respective IdP. Once the App receives each assertion, it must be verified. Once verified, the App can store the assertion in its storage.

E. SERVICE PROVISIONING

The service provisioning use-case illustrates how a user can access an online service using a self-sovereign Identity Management system. Since the ultimate goal of any Identity Management system is to facilitate service provisioning using the system, we sketch its flow in a detailed fashion in Figure 24 and discuss next.

- Alice goes to an SP in order to access one of its services.
- The SP lists the attributes that Alice needs to release to access the requested services. The SP also provides a list

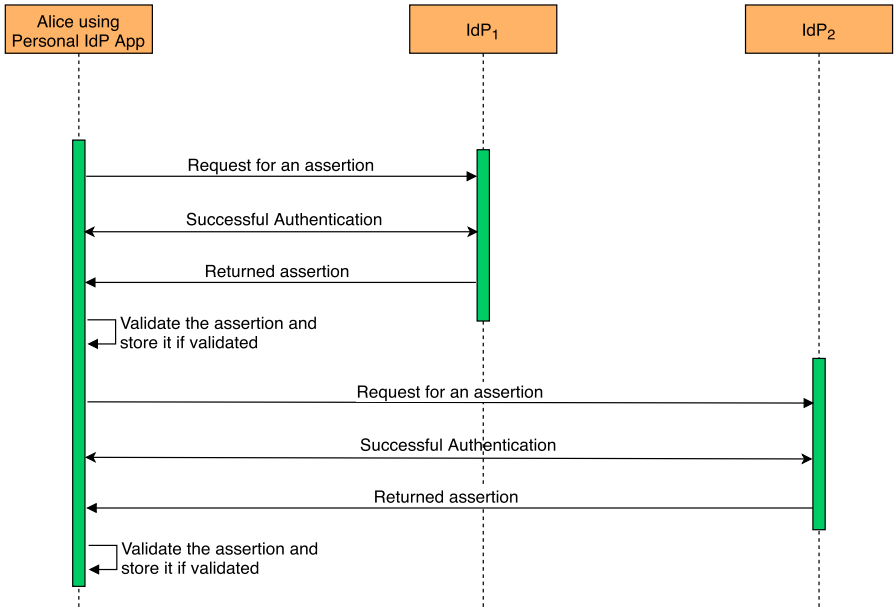

FIGURE 23. Aggregation flow using two IdPs.

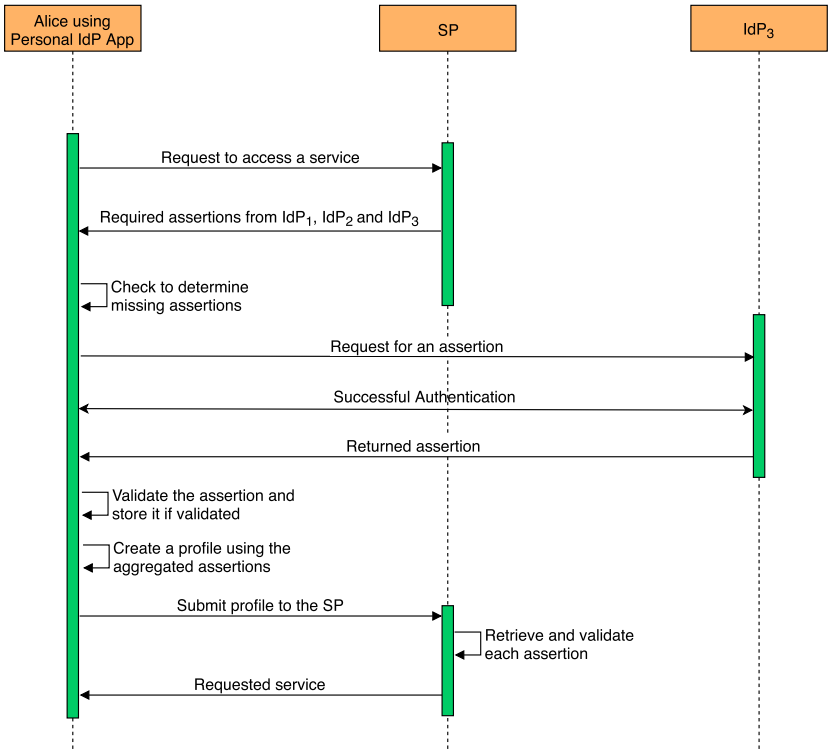

FIGURE 24. Service provisioning using a Personal IdP.

of IdPs from which assertions containing the attributes should be released.

- This will require Alice to build a profile consisting of different assertions, probably collated from different IdPs. To initiate the aggregation process, she provides a link of her personal IdP and is forwarded to the aggregation interface of the personal IdP.
- She engages in the aggregation process using the flow described above (Figure 23).
- Once all the required assertions are collated at the personal IdP, the user can then create a profile and return it to the SP.

The SP, upon receiving the profile, verifies each assertion and finally extracts attributes from all assertions. The attributes are then matched against the requirement for the requested

service. If the requirement is fulfilled, the user is allowed to access the requested service.

IX. DISCUSSION AND CONCLUSION

The concept of self-sovereign identity is an exciting prospect. It has the potential to liberate any user, for the first time ever, from the parochial control of an organization regarding the management of her identities. It will give them the ultimate control over their identity data – an elusive notion under the landscape of the current identity management systems. Because of this reason, there is an optimism surrounding self-sovereign identity which has resulted in different online blogs and technical articles in recent years.

Unfortunately, the existing works are not methodological and they do not explore the topic in a more formal way. In those works, the core concept of self-sovereign identity has mostly been explored from the property perspectives in textual formats. It is as if an identity having a particular set of properties could be defined as a self-sovereign identity. In this article, we have contended against this notion and argued that a self-sovereign identity is not necessarily required to exhibit all these properties. Rather, many of these properties belong to an identity management system. Those works seem to be oblivious that an identity and an identity management system, albeit related, are separate concepts. This confusion has enabled assigning many properties to the concept of self-sovereign identity, even though they are more related to the underlying system. We have highlighted this subtle difference for the first time in this article. In addition, we have created a taxonomy of properties to classify them accordingly so as to provide a layer of separation regarding which properties are for a self-sovereign identity and which are for its corresponding identity management system.

All in all, there is a gap to concretize and formalize the notion of self-sovereign identity. The primary focus of this article is to fill in this gap by seeking an answer to this question in a more fundamental way: what is a self-sovereign identity?

In our quest to reply this question, we have explored a self-sovereign identity in a more fundamental way: by formalizing the concept using mathematical notions and properties. This mathematical notion is conceptualized in way that it captures the essential properties from the taxonomy for a self-sovereign identity. Being based on mathematical foundations, the notion is expressed much more rigorously in comparison to textual definitions. We have also formulated the laws of self-sovereign identity utilizing the essential properties from the taxonomy. This formulation has helped us to highlight the required properties in a more formal way for a self-sovereign identity. Finally, we have divulged into the life-cycle of an IMS and presented our envisioned flows to leverage our concept to be exploited in different aspects of this life-cycle.

One may ask about the practicality of such a system: particularly, if a self-sovereign identity management system can be realized in practice. We are quite optimistic in this regard, specifically in the light of several exciting technological innovations in recent years. We envision a smart-contract supported blockchain system to take the central stage in the realization of this concept. Such a blockchain system exhibits a majority of the foundational and security properties. Specifically, they can provide a solid foundation to deploy a decentralized domain upon which a Self-SID system can be instantiated. This has been briefly explored in the article as well.

In order to explore the usefulness and the applicability of a self-sovereign identity, it is essential to sketch out a few real-life use-cases involving a self-sovereign identity. This will also help to identify a few essential application domains to which the concept of self-sovereign identity will bring significant advantages. In fact, there have been a few attempts in the form of different blockchain-based self-sovereign identity systems. However, as per our analysis, none of them satisfies all the properties of a self-sovereign identity system. In our next step, we aim to fill in this gap by venturing into this direction. Our ultimate goal is to create a self-sovereign identity management system that satisfies the identified properties and can be leveraged to deploy the identified use-cases within the promising application domains. This article, we believe, will be the foundational step towards that aim and will pave down the way for further research in this domain.

REFERENCES

[1] M. S. Ferdous, "User-controlled identity management systems using mobile devices," Ph.D. dissertation, School Comput. Sci., Univ. Glasgow, Glasgow, Scotland, 2015.

[2] K. Cameron. Microsoft Corporation. (Nov. 5, 2005). *The Laws of Identity*. Accessed: Mar. 20, 2019. [Online]. Available: http://www.identityblog.com/stories/2005/05/13/TheLawsOfIdentity.pdf

[3] M. S. Ferdous, A. Jøsang, K. Singh, and R. Borgaonkar, "Security usability of petname systems," in *Proc. 14th Nordic Conf. Secure IT Syst.* Berlin, Germany: Springer, 2009, pp. 44–59.

[4] Modinis Project. (Nov. 23, 2005). *Common Terminological Framework for Interoperable Electronic Identity Management. European Commission*. Accessed: Mar. 20, 2019. [Online]. Available: https://www.cosic.esat.kuleuven.be/modinis-idm/twiki/bin/view.cgi/Main/GlossaryDoc

[5] M. S. Ferdous, G. Norman, and R. Poet, "Mathematical modelling of identity, identity management and other related topics," in *Proc. 7th Int. Conf. Secur. Inf. Netw.*, 2014, pp. 9–16.

[6] A. Lewis. (May 17, 2017). *A Gentle Introduction to Self-Sovereign Identity*. Accessed: Mar. 23, 2019. [Online]. Available: https://bitsonblocks.net/2017/05/17/a-gentle-introduction-to-self-sovereign-identity/

[7] M. Sporny and D. Longley. (May 7, 2016). *A Self-Sovereign Identity Architecture*. Accessed: Mar. 23, 2019. [Online]. Available: https://github.com/WebOfTrustInfo/ID2020DesignWorkshop/blob/master/topics-and-advance-readings/a-self-sovereign-identity-architecture.pdf

[8] C. Allen. (Apr. 25, 2016). *The Path to Self-Sovereign Identity*. Accessed: Mar. 23, 2019. [Online]. Available: http://www.lifewithalacrity.com/2016/04/the-path-to-self-sovereign-identity.html

[9] A. Jøsang and S. Pope, "User centric identity management," in *Proc. Asia Pacific Inf. Technol. Secur. Conf. (AusCERT)*, 2005, p. 77.

[10] S. Cantor, I. J. Kemp, N. R. Philpott, and E. Maler. (Mar. 15, 2005). *OASIS Standard*. Accessed: Mar. 27, 2019. [Online]. Available: https://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf

[11] (Dec. 2, 2007). *OpenID Authentication 2.0—Final*. Accessed: Mar. 27, 2019. [Online]. Available: https://openid.net/specs/openid-authentication-2_0.html

[12] D. Hardt. *The OAuth 2.0 Authorization Framework*. Accessed: Mar. 27, 2019. [Online]. Available: https://tools.ietf.org/html/rfc6749
[13] U. Der, S. Jähnichen and J. Sürmeli, "Self-sovereign identity—Opportunities and challenges for the digital revolution," 2017, *arXiv:1712.01767*. [Online]. Available: https://arxiv.org/abs/1712.01767
[14] 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.
[15] D. Baars, "Towards self-sovereign identity using blockchain technology," M.S. thesis, Fac. Elect. Eng., Math. Comput. Sci., Univ. Twente, Enschede, The Netherlands, 2016.
[16] Q. Stokkink and J. Pouwelse, "Deployment of a blockchain-based self-sovereign identity," in *Proc. IEEE Int. Conf. Internet Things (iThings)*, Jul./Aug. 2018, pp. 1336–1342.
[17] P. Coelho, A. Zúquete, and H. Gomes, "Federation of attribute providers for user self-sovereign identity," *J. Inf. Syst. Eng. Manage.*, vol. 3, no. 4, p. 32, 2018.
[18] A. Tobin and D. Reed, "The inevitable rise of self-sovereign identity," Sovrin Found., Salt Lake City, UT, USA, Tech. Rep., 2016.
[19] J. Andrieu, "A technology free definition of self-sovereign identity," in *Proc. 3rd Rebooting Web Trust Design Workshop*, 2016, p. 4.
[20] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.
[21] C. Lundkvist, R. Heck, J. Torstensson, Z. Mitton, and M. Sena. (Oct. 20, 2016). *UPORT: A Platform For Self-Sovereign Identity*. Accessed: Jul. 16, 2019. [Online]. Available: http://blockchainlab.com/pdf/uPort_whitepaper_DRAFT20161020.pdf
[22] C. Fei, J. Lohkamp, E. Rusu, K. Szawan, K. Wagner and N. Wittenberg. (Mar. 9, 2018). *Jolocom Whitepaper*. Accessed: Jul. 16, 2019. [Online]. Available: https://jolocom.io/wp-content/uploads/2018/07/Jolocom-Technical-WP-_-Self-Sovereign-and-Decentralised-Identity-By-Design-2018-03-09.pdf
[23] D. Reed, J. Law and D. Hardman. (Sep. 29, 2016). *The Technical Foundations of Sovrin*. Accessed: Jul. 16, 2019. [Online]. Available: https://www.evernym.com/wp-content/uploads/2017/07/The-Technical-Foundations-of-Sovrin.pdf
[24] *Blockcerts Guide*. Accessed: Jul. 16, 2019. [Online]. Available: https://www.blockcerts.org/guide/
[25] M. S. Ferdous and R. Poet, "Portable personal identity provider in mobile phones," in *Proc. 12th IEEE Int. Conf. Trust, Secur. Privacy Comput. Commun. (IEEE TrustCom)*, Jul. 2013, pp. 736–745.
[26] *Access Control Lists: Overview and Guidelines*. Accessed: Jun. 11, 2019. [Online]. Available: https://www.cisco.com/c/en/us/td/docs/ios/12_2/security/configuration/guide/fsecur_c/scfacls.pdf
[27] R. S. Sandhu, "Role-based access control," in *Advances in Computers*, vol. 46. Amsterdam, The Netherlands: Elsevier, 1998, pp. 237–286.
[28] E. Yuan and J. Tong, "Attributed based access control (ABAC) for Web services," in *Proc. IEEE Int. Conf. Web Services (ICWS)*, Jul. 2005, p. 569.
[29] M. S. Ferdous, F. Chowdhury and R. Poet, "A hybrid model of attribute aggregation in federated identity management," in *Enterprise Security*. Berlin, Germany: Springer, 2017, pp. 120–154.
[30] D. W. Chadwick and G. Inman, "Attribute aggregation in federated identity management," *Computer*, vol. 42, no. 5, pp. 33–40, May 2009.

MD SADEK FERDOUS received the double master's degrees in security and mobile computing from the Norwegian University of Science and Technology, Norway, and the University of Tartu, Estonia, and the Ph.D. degree in identity management from the University of Glasgow. He is currently an Assistant Professor with the Department of Computer Science and Engineering, Shahjalal University of Science and Technology, Sylhet, Bangladesh. He is also affiliated as a Research Associate with the Centre for Global Finance and Technology, Imperial College Business School. He has several years of experience as a Postdoctoral Researcher in different universities in different European and U.K.-funded research projects. His current research interests include blockchain, identity management, trust management, and security and privacy issues in cloud computing and social networks. He has published numerous research papers and book chapters in these domains in different books, journals, conferences, workshops, and symposiums.

FARIDA CHOWDHURY received the joint master's degrees in networking and e-business centred computing from the University of Reading, U.K., the Universidad Carlos III de Madrid, Spain, and the Aristotle University of Thessaloniki, Greece, and the Ph.D. degree from the University of Stirling, U.K., where she investigated the effect of churn in NAT-ed structured peer-to-peer overlays. She is currently an Associate Professor with the Department of Computer Science and Engineering, Shahjalal University of Science and Technology, Bangladesh. Her research interests include networking, blockchain, big data, cloud computing, HCI, and security and privacy issues in social networks. She has published many articles in reputed journals and as book chapters as well as in different conferences and workshops.

MADINI O. ALASSAFI received the M.S. degree in computer science from California Lutheran University, Thousand Oaks, CA, USA, in 2013, and the Ph.D. degree in cloud computing security from the University of Southampton, U.K., in 2018. He is currently the Chairman and an Assistant Professor with the Information Technology Department, Faculty of Computing and Information Technology, King Abdulaziz University, Saudi Arabia. His current research interests include cloud computing and security, distributed systems, blockchain, the Internet of Things (IoT) security issues, cloud security adoption, risks, cloud migration project management, cloud of things, and security threats. He has published numerous research papers in different conferences, journals, and book chapters.