Withdrawn White Paper

Warning Notice

The attached white paper has been withdrawn, and is provided solely for historical purposes. It has been superseded by the document identified below.

Withdrawal Date	January 14, 2020
Original Release Date	July 9, 2019

Superseding Document

Status	Final
Series/Number	NIST Cybersecurity White Paper
Title	A Taxonomic Approach to Understanding Emerging Blockchain Identity Management Systems
Publication Date	January 14, 2020
DOI	https://doi.org/10.6028/NIST.CSWP.01142020
CSRC URL	https://csrc.nist.gov/publications/detail/white-paper/2020/01/14/a-taxonomic-approach-to-understanding-emerging-blockchain-idms/final
Additional Information	

National Institute of Standards and Technology
U.S. Department of Commerce

A Taxonomic Approach to Understanding Emerging Blockchain Identity Management Systems

Loïc Lesavre
Priam Varin
Peter Mell
Michael Davidson
James Shook
Computer Security Division
Information Technology Laboratory

This publication is available free of charge from:
https://doi.org/10.6028/NIST.CSWP.07092019-draft

July 9, 2019

National Institute of Standards and Technology
U.S. Department of Commerce

Abstract

Identity management systems (IDMSs) are widely used to provision user identities while managing authentication, authorization, and data sharing both within organizations as well as on the Internet more broadly. Traditional identity systems typically suffer from single points of failure, lack of interoperability, and privacy issues such as encouraging mass data collection and user tracking. Blockchain technology has the potential to support novel data ownership and governance models with built-in control and consent mechanisms, which may benefit both users and businesses by alleviating these concerns; as a result, blockchain-based IDMSs are beginning to proliferate. This work categorizes these systems into a taxonomy based on differences in architecture, governance models, and other salient features. We provide context for the taxonomy by describing related terms, emerging standards, and use cases, while highlighting relevant security and privacy considerations.

Keywords

blockchain; credential; data ownership; decentralized identifier; distributed ledger; identity management; public key infrastructure; self-sovereign identity; smart contract; user-controlled identity wallet; zero-knowledge proof.

Additional Information

For additional information on NIST's Cybersecurity programs, projects and publications, visit the Computer Security Resource Center. Information on other efforts at NIST and in the Information Technology Laboratory (ITL) is also available.

Public Comment Period: *July 9, 2019 through August 9, 2019*

National Institute of Standards and Technology
Attn: Computer Security Division, Information Technology Laboratory
100 Bureau Drive (Mail Stop 8930) Gaithersburg, MD 20899-8930
Email: blockchain-idms-paper@nist.gov

All comments are subject to release under the Freedom of Information Act (FOIA).

Acknowledgements

The authors wish to thank all contributors to this publication and their colleagues who reviewed drafts of this report, contributed technical and editorial additions, and supported this work. This includes NIST staff and associates Dylan Yaga, Frederic De Vaulx, Angela Robinson, and Katya Delak.

Audience

This publication is designed for readers with some knowledge of blockchain technology who wish to understand at a high level how blockchain identity management systems work. It is not intended to be a technical guide; the discussion of the technology provides a conceptual understanding. Note that some examples, figures, and tables are simplified to fit the audience.

Executive Summary

Identity management systems allow one to provision identities to users, while managing authentication, authorization, and data sharing both within organizations as well as on the Internet. With traditional identity management, organizations usually store the credentials (e.g., a password) of each user they interact with, or with federated models, they use a third party to store this information. This creates interoperability, security, and privacy concerns due to the privileged position of the entity that controls the identity information.

A possible solution to these issues is found in the use of blockchain technologies for identity management: they can reduce, or even remove, the need for a third party. At a high-level, they can transform data governance and ownership models, and benefit both individual users and businesses. More specifically, it can enable users to control their data and share select personal information to relying parties. It can also enable businesses to streamline their operations by relying on verified user information without having to maintain the infrastructure themselves.

A large number of blockchain-based identity management approaches are currently being explored, implemented, and commercialized. Many of them use, or plan to use, smart contracts, the privacy capabilities gained from zero-knowledge protocols, and other scalability solutions atop the underlying blockchain. This is an emerging field and the features, capabilities, security, and privacy of these proposed systems are often unclear.

Identity is a far-reaching topic, and the systems being designed to support it can take architectural forms that are both on-chain and off-chain, and follow types of governance models spanning from top-down approaches to "self-sovereign" bottom-up ones. Each system has different control and delegation capabilities, as well as scalability constraints.

This work breaks down identifier and credential architectures, discusses their reliance to blockchains, and possible combination patterns. It looks at the levels at which on-chain registries are created, and who can control them. It investigates "bring-your-own" blockchain address schemes, along with credentials being issued as off-chain objects. It does not attempt to judge between the different architectures and models, but instead, highlights their differences.

We first offer a terminology for blockchain identity management as well as a list of associated standards and building blocks. We then provide a breakdown of distinguishing properties and architectures. Next, we discuss public registries, and then, system governance. Finally, we cover some of the security concerns that might affect these systems, along with additional considerations on core blockchain protocols, zero-knowledge proofs, presentation sharing and data mining, as well as ecosystem convergence. To make this discussion less abstract, we offer two use cases.

This will help the reader navigate how blockchain-based identity management systems work and what they can offer. It may be useful for the reader to better understand and build identity management systems, and can contribute towards designing efficient architectures. It will also enable the reader to be aware of what is possible, and thus, better able to distinguish between the many emerging systems.

Table of Contents

List of Appendices

1 Introduction

A large number of blockchain-based identity management approaches are being explored, implemented, and commercialized. This is a new field, and the features, security, and privacy of these proposed systems are often unclear. While many of the approaches hold great promise, most projects rely on the prerequisite of using a blockchain platform that is reliable, secure, scalable, and, sometimes, publicly accessible. Thus, blockchain-based identity management systems are an emerging area that should be watched and carefully evaluated as a potential, but not guaranteed, breakthrough for digital identity and data ownership.

1.1 Background

Identity management systems (IDMSs) are a foundational infrastructure for interactions between entities (individuals, organizations, or things) to support commerce, education, health care, government services, and many other aspects of society. An IDMS must allow entities to authenticate while at the same time distributing information about those entities to enable the granting of access privileges of differing levels or types.

With traditional identity management, businesses store credentials about each user with which they interact (e.g., a password). This enables a user to directly authenticate to the business (or more technically "relying party") with which they need to interact, as shown in Figure 1. However, the user is burdened with needing to separately authenticate to each business using different credentials. In addition, businesses are not able to automatically obtain and evaluate verified identifying information about each user.

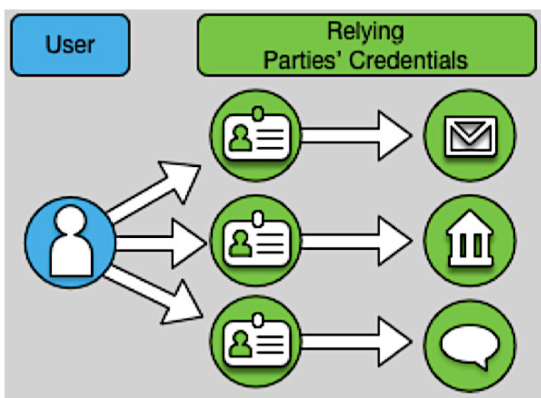

Figure 1: Traditional Identity Management (copied from [1])

More recently, federated identity management systems [1] enable credential service providers to maintain user credentials on behalf of various relying parties. This enables single-sign-on capabilities where a user utilizes a single set of credentials to access a large number of services, as shown in Figure 2. This frees up the user from having to maintain many passwords. However, it creates security and privacy concerns given the privileged position of the credential service provider between the user and relying parties. For example, it presents a single point of failure that could inhibit the user's access to the relying parties or, even worse, could enable the credential service provider to masquerade as a user.

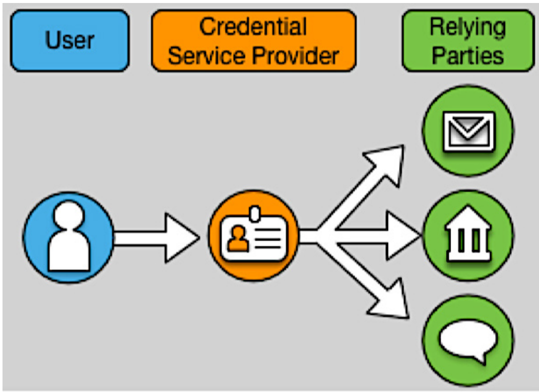

Figure 2: Federated Identity Management (copied from [1])

A possible solution to these issues may be found in the use of blockchain technologies for identity management; blockchains can remove the need for traditional credential service providers and enable direct user to relying party interactions with verified information.

From the subject's perspective, blockchain-based IDMSs allow them to own their data, or control who owns it, while being able to share verified information with relying parties to facilitate a certain set of actions (e.g., business transactions). This architecture may enable the subjects to be in a better position to give their consent. For example, a subject can cryptographically approve a transaction prior to some relying party executing it on their behalf (e.g., a bank could not open an account for a user without their attested prior approval).

From the relying party's perspective, blockchain-based IDMSs allow them to verify that some user information needed to initiate a transaction is valid without having to store the personal information themselves. A key implication is that it lowers privacy and security burdens, and may facilitate bootstrapping new business activities as well as automating existing ones.

In summary, blockchain-based IDMSs have the potential to greatly enhance security and privacy, and grant built-in control and consent capabilities for both users and relying parties. However, there are tradeoffs to be made and it will be necessary to carefully evaluate the emerging solutions.

1.2 Purpose and Scope

This document provides an introduction to the different blockchain identity management approaches currently being explored and implemented. The purpose is not to review each solution individually, but to provide a taxonomic approach that will give the reader different viewpoints and methods by which blockchain-based identity management can be designed. In this way the document highlights the different features and characteristics that are possible while exploring the opportunities, challenges, and risks they represent.

As an emerging field, weaknesses may become evident that negate the apparent advantages or other data models may emerge with even greater benefit. It may take years for the proper blockchain infrastructure, the identity management platforms, and related user tools to mature and be deployed at scale. While the time for most readers to deploy these capabilities lies somewhere in the future, we argue that the capabilities and architecture designs discussed in this paper represent a major improvement over traditional identity management systems and thus, that this field deserves careful consideration and scrutiny today. We hope that this paper provides the foundational tools to enable such an ongoing evaluation.

1.3 Disclaimers and Clarifications

We will be referring to "blockchains" throughout this paper. However, this work may be extended to any kind of Distributed Ledger Technology (DLT). This paper refers to blockchain, smart contract capabilities, and related concepts without recommending or endorsing any particular protocols. Any products or protocols mentioned are for explanatory purposes only and do not imply any endorsement or suitability for use.

1.4 Blockchain Identity Management Initiatives and Guidance

Some organizations have already written guidance on blockchain in identity management. The European Union recently published *Blockchain for Government and Public Services* [2] and *Blockchain and Digital Identity* [3]. In the United States, there have been initiatives led at the state level such as the Illinois Blockchain Initiative [4]. The American Council for Technology and Industry Advisory Council (ACT-IAC) published a *Blockchain Primer* [5] to introduce how blockchain could impact the Federal Government as well as a *Blockchain Playbook* [6] to introduce how it could be applied to the U.S. Federal Government for different purposes, including identity management.

There are a handful of blockchain-based IDMS pilots as well. Some organizations have already adopted the use case of diploma and certificate issuance on the blockchain, such as the Massachusetts Institute of Technology with Blockcerts and Learning Machine [7]. Some jurisdictions are experimenting with blockchain-based IDMSs at different levels, such as Estonia (for electronic medical records) [8], the City of Zug in Switzerland using uPort (on the Ethereum blockchain) [9], and the Provinces of British Columbia and Ontario in Canada using the Verifiable Organizations Network (on the Sovrin blockchain) [10]. Note that many projects are currently under active development, characterizing the growing interest in blockchain-based identity management.

1.5 Document Structure

The rest of this document is organized as follows:

- **Section 2** introduces blockchain technology and smart contracts at a high-level.
- **Section 3** defines terminology and discusses emerging standards and building blocks for blockchain identity management.
- **Section 4** introduces and discusses a taxonomy in the form of distinguishing properties, which are then used to characterize different architecture designs.
- **Section 5** introduces some of the security concerns and their mitigation mechanisms.
- **Section 6** provides additional considerations.
- **Section 7** introduces potential use cases.
- **Section 8** is the conclusion.
- **References** lists the references used throughout the document.
- **Appendix A** provides a list of acronyms and abbreviations used in the document.
- **Appendix B** contains a glossary for selected terms defined in the document.

2 Blockchains and Smart Contracts

We invite the readers, who have little or no knowledge of blockchain technology and who wish to understand at a high level how it works, to read *Blockchain Technology Overview* NISTIR 8202 [11]. It defines blockchain as "tamper evident and tamper resistant digital ledgers implemented in a distributed fashion (i.e., without a central repository) and usually without a central authority (i.e., a bank, company or government). At their basic level, they enable a community of users to record transactions in a shared ledger within that community, such that under normal operation of the blockchain network no transaction can be changed once published."

The technology is called blockchain because the transactions are grouped and published separately in distinct data structures called blocks, which are cryptographically linked together, duplicated, and distributed in a peer-to-peer network to prevent tampering of previously published transactions. Blockchain accounts are based on asymmetric-key cryptography and allow participants to sign transactions. The transactions are added to blocks that must be validated by the nodes that are running the blockchain's peer-to-peer node client. Consensus models determine which node gets the privilege of publishing the next block (see Section 4.6 on *Consensus Comparison Matrix* of NISTIR 8202).

The paper discusses two important categories that pertain to our investigation of identity management systems: "Blockchain networks can be categorized based on their permission model, which determines who can maintain them (e.g., publish blocks). If anyone can publish a new block, it is *permissionless*. If only particular users can publish blocks, it is *permissioned*. In simple terms, a permissioned blockchain network is like a corporate intranet that is controlled, while a permissionless blockchain network is like the public internet, where anyone can participate. Permissioned blockchain networks are often deployed for a group of organizations and individuals, typically referred to as a consortium."

Some blockchains have highly expressive native smart contract capabilities which are often useful for blockchain identity management solutions. A smart contract is defined as: "a collection of code and data (sometimes referred to as functions and states) that is deployed using cryptographically signed transactions on the blockchain network (e.g., Ethereum's smart contracts, Hyperledger Fabric's chaincode). The smart contract is executed by nodes within the blockchain network; all nodes that execute the smart contract must derive the same results from the execution, and the results of execution are recorded on the blockchain. […] The smart contract code can represent a multi-party transaction, typically in the context of a business process. In a multi-party scenario, the benefit is that this can provide attestable data and transparency that can foster trust, provide insight that can enable better business decisions, reduce costs from reconciliation that exists in traditional business to business applications, and reduce the time to complete a transaction. […] Smart contracts must be deterministic, in that given an input they will always produce the same output based on that input." Furthermore, a source of off-chain data that serves as input for a smart contract is referred to as an "oracle".

Note that the owner of a blockchain identity management system does not necessarily own the blockchain upon which this system is built. In fact, an entity can deploy an identity management system without having to build or maintain the underlying blockchain infrastructure that is being leveraged.

3 Fundamentals of Blockchain Identity Management

Prior to us introducing our taxonomy in the next section, this section details key terminology, common roles and objects, emerging supportive standards, essential building blocks, and a blockchain identity management communication stack. These terms, standards, and abstractions are used by most blockchain identity management systems.

3.1 Terminology

Specialized terminology is used for blockchain-based identity management schemes. Unfortunately, the terminology is not always consistent among the various projects and standards. Further complicating matters is that some domain-specific terms are related to identity management in general while others are specific to blockchain identity management. Understanding the following terms is necessary in order to understand the concepts discussed in this paper.

Claim: A characteristic or statement about a *subject* made by an *issuer* as part of a *credential*.

Credential: A set of one or more *claims* made by an *issuer.* A *credential* is associated with an *identifier*.

Custodian: An *entity* acting on behalf of another *entity* with respect to their identifiers and/or credentials.

Entity: A person, organization, or thing.

Holder: A *custodian* holding a *credential* on behalf of a *subject*.

Identifier: A blockchain address or other pseudonym that is associated to an *entity*.

Issuer: An *entity* that issues a *credential* about a *subject* on behalf of a *requester* and owns one or more *identifiers.*

Presentation: Information derived from one or more *credentials* that a *subject* discloses to a *verifier* (working on behalf of some *relying party*) to communicate some quality about a *subject*.

Relying Party: An *entity* that receives information about a subject from a *verifier.*

Requester: An *entity* that makes a request to an *issuer* to issue a *credential* about a *subject*.

Subject: An *entity* that acts as a regular participant in a given identity management system and owns one or more *identifiers*.

System Owner: An *entity* that owns a given identity management system.

Verifier: An *entity* that verifies the validity of a *presentation* on behalf of a *relying party*.

3.2 Blockchain-based Identity Management Roles and Object Relationships

With this terminology we can identify the common roles that occur in blockchain-based IDMSs and the relationships between these roles. We can also identify common objects found in these systems and the relationships between those objects.

Figure 3 provides a high-level overview of the identity management roles defined in Section 3.1.

- *Requesters*, *Issuers*, and *Subjects* are involved in credential issuance.
- *Subjects*, *Verifiers*, and *Relying Parties* are involved in presentation disclosure.
- *Requesters* ask for the issuance of a credential from *Issuers*. *Issuers* provide credentials to *Subjects*.
- *Subjects* reveal presentations to *Verifiers.*
- *Verifiers* verify presentations on behalf of *Relying Parties*.

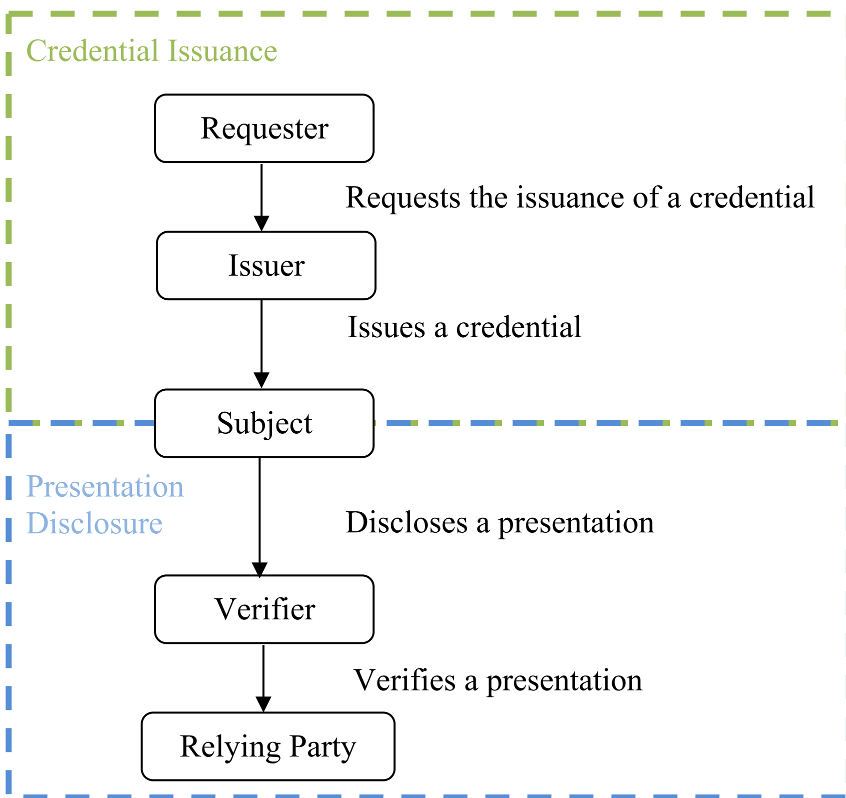

Figure 3: Identity Management Roles

Note that these roles are not exclusive. For instance, a subject and an issuer can both take the requester role or a subject and a verifier can both be a relying party. Depending on the IDMS, the approval of a subject may be required to issue a new credential to that subject.

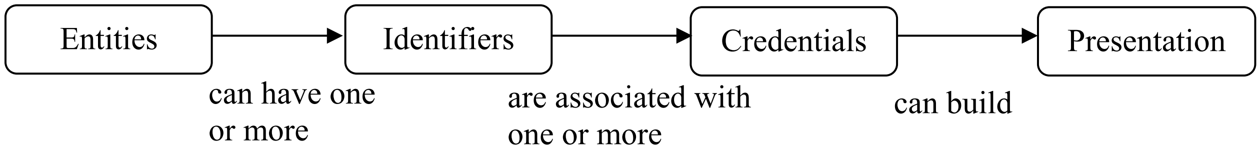

Figure 4: Hierarchy of IDMS Objects

Figure 4 provides a high-level overview of the objects that entities interact with in a blockchain IDMS. The figure shows that entities can have one or more identifiers, that identifiers are associated with one or more credentials, and that presentations are derived from credentials.

3.3 Emerging Standards

There is a set of emerging standards that support blockchain-based IDMSs including:

- Decentralized Identifiers and Verifiable Credentials, from the World Wide Web Consortia (W3C)
- Open Badges, from Mozilla and IMS Global
- Universal Resolver and Identity Hubs, from the Decentralized Identity Foundation (DIF)

In the subsequent sections of this paper, we will be using the terms identifiers, credentials, and presentations, but will not necessarily be referring to standards of this section.

In addition, we will refer to blockchain network specific standards such as Ethereum Request for Comments (ERCs) and Bitcoin Improvement Proposals (BIPs).

Decentralized Identifiers – W3C:

Decentralized Identifiers (DIDs) [12] are identifiers whose purpose is to facilitate the creation of persistent encrypted private channels between entities without the need for any central registration mechanism. They can be used, for example, for credential exchanges and authentication. An entity can have multiple DIDs, even one or more per relationship with another entity (see *Pairwise-pseudonymous and Single-use Identifiers* in Section 4.3). When an entity has one DID per relationship with other entities, it is called a pairwise pseudonymous DID. Ownership of a DID is established by demonstrating possession of the private key associated with the public key bound to the DID.

A DID method is a public, standard set of schemes by which to create, resolve, update, and delete DIDs. These methods allow for DID registration, replacement, rotation, recovery, and expiration within an IDMS.

A DID has the following format:

“did:” + <did-method> + “:” + <method-specific-identifier>

As an example, a DID for a “NIST DID method” could look like: did:nist:0x1234abcd.

As part of a DID method, a DID resolver allows one to take a DID as input and to return the associated metadata, called a DID document and formatted as a JavaScript Object Notation for Linked Data (JSON-LD) object [13]. JSON-LD is a JSON-based format used to serialize linked data and build interoperable services. According to W3C's primer [14], a DID document is comprised of the following standard elements:

- A DID that identifies the subject of the DID document
- A set of public keys used for authentication, authorization, and communication mechanisms
- A set of authentication methods used for the DID subject to prove ownership of the DID to another entity
- A set of authorization and delegation methods for allowing other entities to operate on behalf of the DID subject (i.e., custodians)
- A set of service endpoints to describe where and how to interact with the DID subject
- A Uniform Resource Identifier (URI) to uniquely identify terminology and protocols that allows parties to get a common understanding of the identifier
- A timestamp for auditing
- A signature for integrity

Universal Resolver – DIF:

While DID documents can be retrieved through using a DID resolver, there are advantages to having a more general resolver that can communicate with multiple decentralized identifier systems (including DID systems). The Universal Resolver [15] achieves this goal; it enables application code to be written to a single resolver interface that enables communication to multiple decentralized identifier systems. A DID-based blockchain IDMS that supports the Universal Resolver must define and implement a DID Driver that links the Universal Resolver to their system-specific DID Method for reading DID documents. This allows applications relying on the IDMS to query DIDs in a common interface so they do not have to deal with fetching the system-specific DID methods themselves. This takes place according to the steps shown in Figure 5.

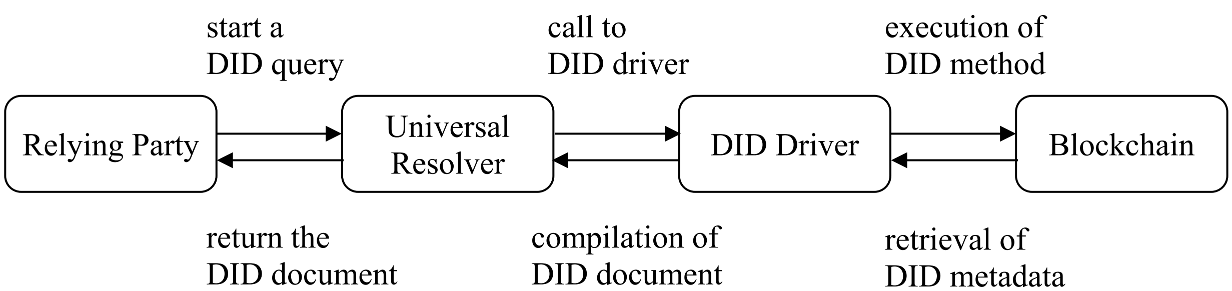

Figure 5: DID Lookup using the Universal Resolver

Verifiable Credentials and Verifiable Presentations – W3C:

The Verifiable Credentials specification [16] defines a format for credentials that can be exchanged between DIDs (using JSON-LD). A Verifiable Credential is a tamper-resistant credential that is cryptographically signed by its issuer.

A Verifiable Credential includes:

- DIDs for the subject and the issuer
- URI to uniquely identify the credential
- Claims data or metadata to access it
- URI to uniquely identify terminology and protocols that allows parties to get a common understanding of the identifier
- Expiration conditions
- Credential status (active, suspended, or revoked)
- Cryptographic signature of the issuer

The W3C specification also defines Verifiable Presentations. A Verifiable Presentation is a tamper-resistant presentation derived from a Verifiable Credential and cryptographically signed by the subject disclosing it.

A Verifiable Presentation includes:

- URI to uniquely identify contexts
- URI to identify the presentation
- One or more verifiable credentials, or data derived from them
- Cryptographic signature of the subject

Open Badges – Mozilla and IMS Global:

Open Badges [17] is another approach to credentials, which are referred to as badges. There are three core data classes used to instantiate a badge: Assertions, BadgeClasses, and Profiles. They have the following features:

- The "Assertion" class contains data about the entity that received the badge (the entity about which something is being asserted), the issuance timestamp, as well as instructions for verifying the information hosted in the badge. Additional properties can also be made available, such as a revocation status or an expiration date.
- The "BadgeClass" class adds context to the type of credential that is enclosed in the badge by listing its name and category, the criteria used to describe how to earn the credential, as well as a reference to the entity that issued the badge.
- The "Profile" class brings more information (e.g., name, email address, phone number, public keys) about the entities linked to the badge (e.g., the badge issuer, recipient, and endorser).

Just like DID documents and Verifiable Credentials, Badges take the form of JSON-LD documents that can be encoded into Quick Response (QR) codes, allowing easier integration into applications.

Identity Hubs – DIF:

Identity Hubs [18] are encrypted personal datastores connected together, using both edge devices (e.g., smartphones, personal computers) and cloud storage. They are used to securely store and share identity data when such sharing is approved by the owner.

An Identity Hub is made of one or more Hub instances, which can run on a personal device or be hosted by a provider. Each Identity Hub is linked to a given DID and can be integrated with the Universal Resolver. The data attached to a DID is replicated and stored across a set of Hub instances. This architecture was designed to avoid single points of failure as well as to let a subject manage access permissions granularly.

3.4 Building Blocks

The building blocks of blockchain identity management systems vary, but at a high-level, they are commonly comprised of the following technical components:

Blockchain:

A blockchain can support the management of keys and identifiers by acting as a Decentralized Public Key Infrastructure (DPKI).[1] Note that the blockchain may be application-specific such as Hyperledger Indy [22] and/or may support a native smart contract platform. In most cases, the DPKI, sometimes augmented by separate protocols atop the blockchain, forms a decentralized identifier system (called DID method if it follows the DID specification). In addition to keys and identifiers, credentials may also rely on the blockchain.

Second Layer Protocol:

A decentralized identifier system may rely on both a blockchain and a separate protocol on top of it, often referred as "second layer" (off-chain) protocol. These protocols can be used to build scaling solutions by "off-loading" operations away from the blockchain layer. That way, smart contracts can be designed such that blockchain transactions (triggered by function calls) track a set of operations rather than a single one. For example, the SideTree protocol [23] (run by SideTree nodes that are separate from those of the underlying blockchain) allows one to bundle DID operations together before posting them onto a blockchain.[2]

[1] NIST Special Publication (SP) 800-32 [19] defines a Public Key Infrastructure (PKI) as follows: "[A PKI] binds public keys to entities, enables other entities to verify public key bindings, and provides the services needed for ongoing management of keys in a distributed system". Note that the company Evernym was awarded a grant from the U.S. Department of Homeland Security in 2017 to develop a decentralized key management solution based on NIST SP 800-130, *A Framework for Designing Key Management Systems* [20]. This became the key management foundation of the Sovrin [21] IDMS; the Sovrin codebase was then added to the Hyperledger Foundation open-source projects under the name of Hyperledger Indy.

[2] The SideTree protocol (released as a DIF project) has been implemented to develop decentralized identifier systems (that follow the DID specification) by Microsoft on top of the Bitcoin protocol (the DID method is called Identity Overlay Networks [24] (ION)), and by Transmute Industries (with ConsenSys) on top of the Ethereum protocol (the DID method is called Element).

In addition to the scalability benefits, second layer protocols may have a different level of privacy than transactions in the underlying blockchain. Finally, second layer protocols do not function as standalone blockchains, rather they require one or more blockchains to operate. A key implication is that second layer protocols can help promote the development of interoperable, blockchain-agnostic systems by allowing for the integration of multiple blockchains without necessarily requiring any fundamental change to their codebase.

Smart Contracts:

Blockchains may support smart contracts, which are vital to many blockchain-based IDMSs (some of them implementing all the logic in the form of smart contracts). The power of smart contracts is that they can act as a trusted third party given that the blockchain network guarantees the execution of their code. This enables blockchain-based IDMSs to use smart contracts to replace many functions formerly assumed by the traditional credential service provider in non-blockchain identity management solutions, and potentially increase trust in these systems. In particular, they are currently used to implement on-chain registries and governance structures.

Credential Storage Methods:

A foundational architectural feature for blockchain IDMSs is the method (or methods) by which credentials are stored (see Section 4.4.2 on *Credential Architectures*). Some blockchain-based IDMSs allow for storage of credentials using a blockchain while others store the credentials off-chain. Off-chain credentials may be stored by a subject in a wallet application (explained in the *User-Controlled Identity Wallet* paragraph below) or by a third party custodian to whom the subject has delegated this role.

Data Exchange Models:

To request, issue, disclose, and verify credentials and/or presentations (e.g., for authentications), blockchain-based IDMSs commonly leverage data exchange formats such as JSON Web Token (JWT), Security Assertion Markup Language (SAML), and eXtensible Data Interchange (XDI).

User-Controlled Identity Wallet:

A user-controlled identity wallet is an application that primarily aims at allowing a subject to hold identifiers and corresponding private keys, as well as credentials. It also serves as an interface for entities to interact with one another. For example, the subjects can receive and approve credentials from the issuers, and disclose presentations to relying parties. Actions can be initiated automatically through Application Programming Interfaces (APIs) calls that may be triggered by a user through scanning QR codes. Depending on the system identifier architecture (see Section 4.4.1), a subject may be able to generate an identifier on their own directly in a wallet (it may thus be done offline).

Identity wallets may be linked to cloud data custodians to benefit from various services such as data and/or private key storage, backup, and recovery mechanisms. Wallets that are proposed as a service by a third party that controls a user's private keys are called custodial wallets.

In addition, identity wallets may act as a control center as entities can approve requests for information, thereby giving their consent to perform some action. It may also be a gateway to access and use applications and services (e.g., a decentralized application store).

Identity wallets may take various forms such as dedicated hardware wallets or mobile applications (or even paper wallets, private keys being simply printed out and kept somewhere safe). They may also come natively in a browser, an operating system, or as extensions.

Application Libraries:

There exist application libraries and APIs that facilitate the integration of applications supporting the various identity management roles (e.g., requester, issuer, relying party, and verifier roles). Note that Hyperledger Aries [26] is a framework released by the Hyperledger Foundation that offers several client-side components and wallet services integration to support interactions between participants in blockchain-based IDMSs.

3.5 Blockchain Identity Management Stack

The Decentralized Identity Foundation published the draft protocol stack [27] shown in Table 1. It shows a breakdown of blockchain identity management layers with the aim of facilitating the emergence of portable and interoperable solutions. Note that adjacent layers do not have to be built as separate applications and can be grouped together if desired for simplicity, scalability, or to more closely align with adopted standards. While DID-specific, the stack should be similar for approaches using other decentralized identifier systems.

Table 1: Proposed Identity Stack (from the Decentralized Identity Foundation [27])

Layer	Description
Application	Application(s) that interact with a given identity management system through library integrations and API calls
Implementation	Libraries that integrate the system in third-party applications
Payload	Message format(s) - such as JWT - used to exchange data between participants
Encoding	Method(s) for encoding data at both the encryption and payload layers
Encryption	Method(s) for encrypting messages between participants as well as encrypting the data held by the identifier owner
DID Authentication	Method(s) to authenticate a participant using their DID
Transport	Transport protocol(s) used for sharing data between participants and devices, such as Hyper-Text Transfer Protocol (HTTP) or a QR code
DID Resolution	DID Resolver used to convert a DID into its corresponding DID document
DID Operation	Create, Read, Update, and Delete operations for a DID document
DID Storage	Method for storing DID Documents and DIDs
DID Anchor	Network that serves as medium for DIDs

4 Blockchain Identity Management System Taxonomy

This section discusses how blockchain identity management systems are constructed and what differentiates the various approaches. We examine system authority models, identifier origination schemes, and credential issuance schemes in Section 4.1. We then evaluate methods for identifier and credential management in Section 4.2, and presentation disclosure in Section 4.3. Section 4.4 looks at different system architecture designs and Section 4.5 discusses the use of public registries and related implications. We conclude our taxonomic analysis with a higher level discussion of system governance options in Section 4.6.

4.1 Authority Model

This section discusses the different control models for blockchain IDMSs and the different ways for such systems to establish new identifiers for their users.

4.1.1 Top-down vs Bottom-up Organizational Structures

The authority model of a system specifies how it is controlled. The two main approaches are top-down and bottom-up (with the latter being frequently associated with "self-sovereign" identity schemes). Note that they form a spectrum of authority models that can support various use cases and serve as a novel medium to represent different types of power structures (with appropriate power delegation mechanisms).

Top-down Approach:

A system owner acts as a central authority that has control over identifier origination and/or credential issuance. Power may be delegated through roles to create a hierarchical structure. This model may be appropriate for organizations that want to explore distributing their processes and architectures to better meet their needs and provide enhanced control and privacy for the users while keeping ownership of the system and control of its governance, as discussed in Section 4.6. An example system using this approach is described in *Smart Contract Federated Identity Management without Third Party Authentication Services* [28].

Bottom-up Approach:

No single entity acts as a central authority that has control over identifier origination and/or credential issuance. Participants manage their own identifiers, but must still follow the rules of the IDMS (often enforced through a set of smart contracts). This approach relies on a web-of-trust, since there is no central authority. Note that this does not exclude the possibility of some entities playing more significant roles than others in designing and maintaining the system architecture and incentives.

4.1.2 Identifier Origination Schemes

There are many possible methods for creating new identifiers within blockchain IDMSs. The generation of blockchain addresses is achieved directly by the subjects (who control the associated private keys). Blockchain addresses alone, however, do not fully meet the need of identity management; there must be additional logic to use them as identifiers in a IDMS.

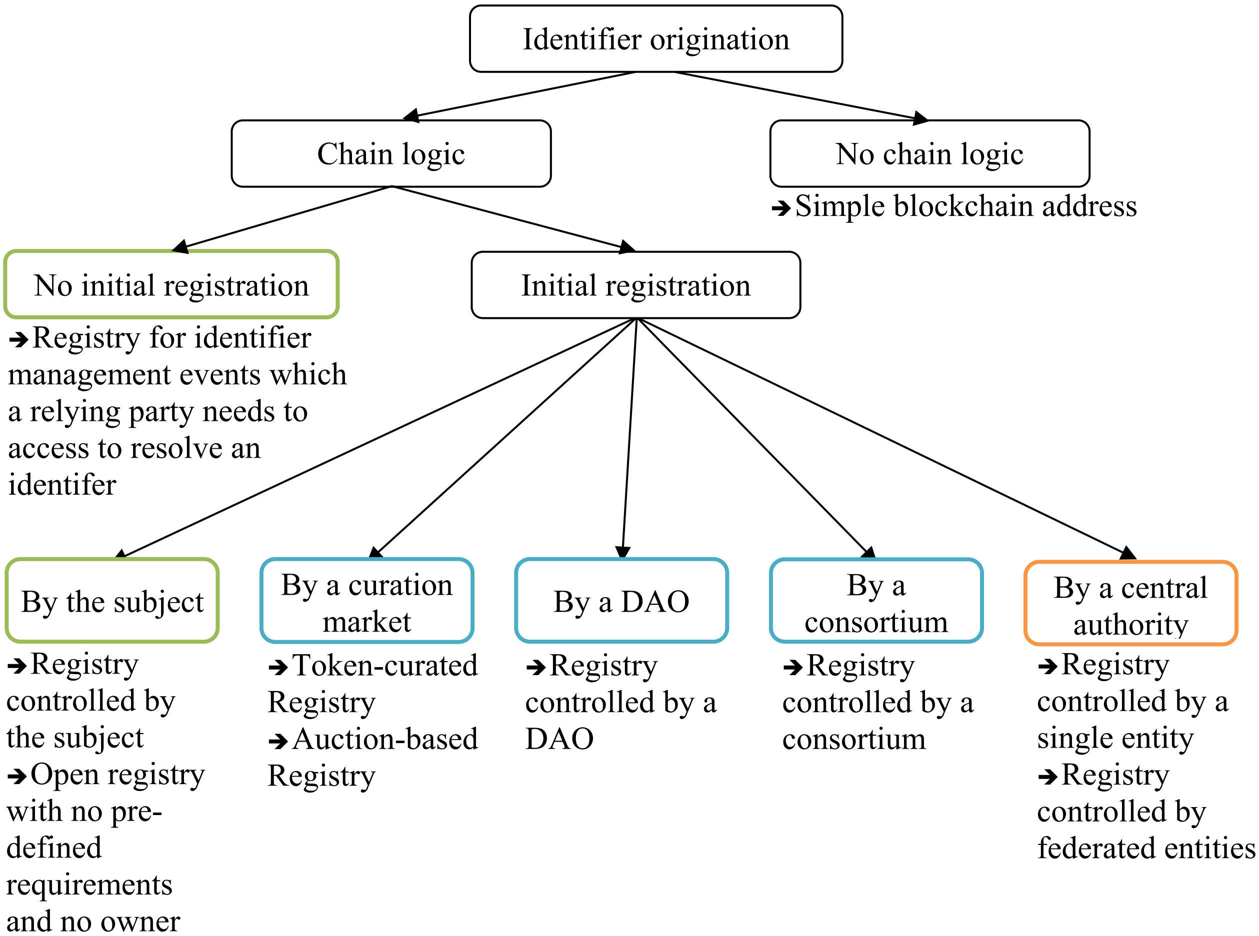

Figure 6: Identifier Orgination Schemes

Figure 6 contains a diagram showing different methods that can be used by systems to originate identifiers. Identifier origination based on a central authority with a top-down approach is shown in the bottom right (in orange). Schemes involving no initial registration or a self-registration following a bottom-up authority approach are on the left (in green). Finally, schemes involving a curation market (see Section 4.5 on *Public Registries and Reputation Management Implications*), a Decentralized Autonomous Organization (DAO), or a consortium can lean towards one side or the other depending on how the permissions are implemented and controlled by the participants (in the middle of the figure in blue). An example of DAO-controlled identifier registration for Internet Protocol (IP) addresses can be found in [29]. Section 4.4.1 on *Identifier Architectures* provides different approaches for implementing these identifier origination schemes.

4.1.3 Credential Issuance Schemes

A credential is issued to a subject by an issuer following a request by a requester. The approval of the subject may be required and the issuer may be compensated for issuing the credential (e.g., through some marketplace mechanism built into the protocol).

With the top-down authority model, credential issuance may be controlled or regulated by a central authority (see Section 4.1).

In the bottom-up authority model, any user can issue a credential to another user.[3]

A credential might also be self-issued by a subject. This would be used, for example, when a subject wants to publicly share information such as a public key, a service endpoint to make themselves reachable, or consent preferences to help other users know how to interact with them.

Note that a credential may be required to be issued according to a standardized nomenclature.

4.2 Identifiers and Credentials Management

This section discusses lifecycle and custody issues related to identifiers and credentials. This includes creation, issuance, discoverability, transferability, recovery, suspension, and revocation.

4.2.1 Lifecycle

Lifecycle Determination at Origination:

The lifecycle of a given identifier or credential can be set at the time of origination such that there will be no need for outside intervention in the future (e.g., making it expire after a certain amount of time or making it irrevocable). This can enable an identifier or credential to take a lighter, self-supporting form in order to let the subject be more independent (see *Bring-your-own Blockchain* Address in Section 4.4.1.2 and *Off-chain Object* in Section 4.4.2.2). In the case where the identifier or credential is irrevocable, a relying party may not need to be actively connected to the identity management system in order to verify the credential or identifier. Alternatively, if an identifier or credential does not have its lifecycle fixed, entities need access to the blockchain to verify them.

Suspension and Revocation:

An identifier or a credential may be suspended or revoked by the issuer, the holder, or when predefined conditions are met.[4] Furthermore, performing these actions may require approval from the participants involved.

4.2.2 Custody and Delegation

This section discusses the custody and delegation processes for identifiers and credentials, including ownership, storage, and transferability. Control over an identifier and/or a credential can be delegated to a custodian for a certain period of time. This can enable marketplaces to provide services while acting on behalf of the subject, such as storage, management of control and consent preferences and relationships with relying parties, recovery mechanisms in case of loss, and authenticated communication channels.

[3] There are additional advanced schemes to issue credentials anonymously and without relying on any trusted issuer by using the techniques in [30], but the claims for which these credentials are issued must be verifiable by anyone participating in that system. Another credential issuance scheme is using a threshold of mutually distrusting parties as in [31].

[4] Blockchains can help make the revocation process more transparent and secure; for instance, CertLedger [32] is a scheme that is comparable to Google's Certificate Transparency (CT), while preventing the "split-world" attack that is possible against CT.

The identifiers themselves may be stored publicly on a blockchain or may remain privately stored and shared off-chain, depending on the IDMS (see Section 4.4.1 on *Identifier Architectures*). Users may lose their private keys associated with an identifier, which may be recovered through a variety of mechanisms: a custodian designated by the user, a list of user-appointed trustees (social recovery), time delay mechanisms, and/or a central authority. Also, an identifier may be abandoned and what is owned by the identifier transferred to another. This may be done for key rotation purposes and not just when the private keys are lost. In Sovrin [21] for example, programs - called "agents" – can act on behalf of an identifier and help them perform tasks such as interacting with the ledger, transacting with other agents, or serving as backup datastores.

In general, credentials are not transferable from one subject to another. However, transferability can be appropriate for specific use cases, such as representations of ownership (e.g., a certificate proving ownership of a good that a subject may then be able to transfer on their own if and when selling the good). Systems may implement this using some form of a non-fungible token (see Section 4.4.2 on *Credential Architectures*).

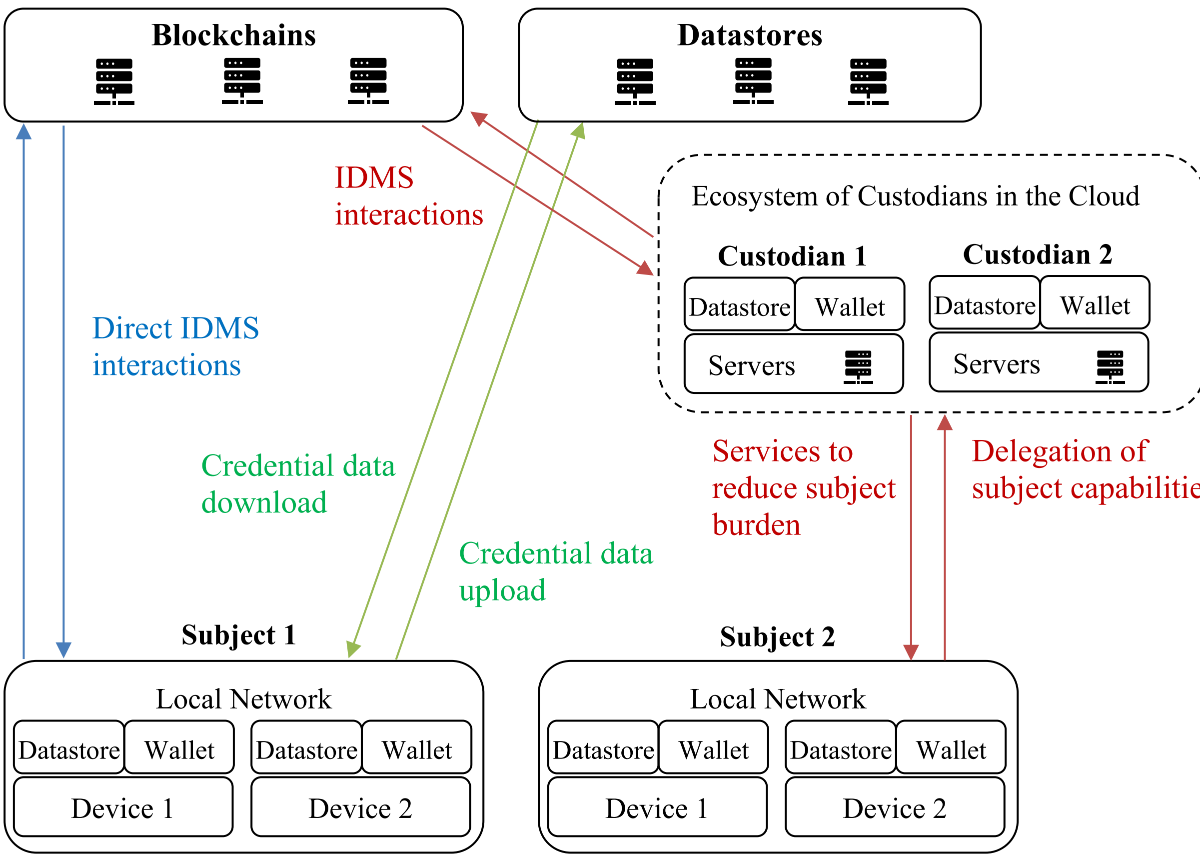

Figure 7: Interactions between subjects, custodians, and decentralized systems

Figure 7 is a diagram of different interactions between a subject and an identity management system; these interactions are either direct or delegated through an identifier custodian.

4.3 Presentation Disclosure

A presentation is a quality derived from one or more credentials, which allow subjects to authenticate themselves and to share verified information with a relying party. This can reduce, or even remove, the need for a third party. The sharing of a presentation from a subject to a relying party is called presentation disclosure. This relationship comes with its own management, control, and consent considerations, which the following properties attempt to characterize.

Subjects can control the release of their data with relying parties (e.g., businesses, applications) and may do so at differing levels of granularity to limit information being released to the minimal necessary. Note that a subject may be compensated for presentation disclosures (e.g., with rewards and reputation systems built-in monetization schemes).

Selective Disclosure Mechanisms:

A presentation disclosure may involve sharing an entire credential, one or more claims from a credential, or a quality derived from a credential. A presentation can include a minimal amount of information to interact with a relying party on a need-to-know basis, with a zero-knowledge proof to verify. Subjects may therefore have the ability to avoid oversharing information.

Zero-knowledge proofs are cryptographic schemes where a prover is able to convince a verifier that a statement is true, without providing any more information than that single bit (that is, that the statement is true rather than false).

Consider a patron who is stopped by the bouncer while attempting to enter a bar, because the bouncer must be convinced that the patron is at least 21 years old. The patron shows the bouncer their driver's license, the bouncer quickly looks for a birthday, and then the patron can enter if they are of age. In this scenario, the bouncer learns far more information about the patron than would be ideal, and a particularly malicious bouncer may be able to learn enough about the patron that they can commit identity theft. Contrast this example with one that employs a zero-knowledge proof scheme. The prover (the patron) proves to the verifier, the bouncer, the statement "I, the prover, am at least 21 years old". They are able to do so without revealing their birthday, driver's license number, or any other information. The patron then enters the bar with their identity and privacy secure, but a different, underage patron is unable to create a convincing proof.

Zero-knowledge protocols (those utilizing zero-knowledge proofs) are a major area of active research (see Section 6.2 for a high-level technical overview).

Pairwise-pseudonymous and Single-use Identifiers:

Users may be able to maintain a set of special purpose identifiers that are not linked to the primary identifier, which enables users to maintain a level of anonymity. For example, users may use pairwise-pseudonymous identifiers, where they have a unique dedicated identifier for each relationship they have with a third-party.

Alternately, they may use single-use identifiers that are discarded after a particular interchange [33].

BIP-32 [34] can be used to create multiple unlinkable identifiers from a single master key. Note that identifier unlinkability schemes can be combined with selective disclosure mechanisms.[5]

Unicast, Multicast, and Broadcast Disclosure Modes:

A presentation can be disclosed to a single relying party, a group of relying parties, or everyone. Public disclosure has reputation management implications (see Section 4.5) and is often used by relying parties who publicly disclose a presentation about themselves in order to prove who they are and to justify that they have a valid reason to request presentations and to receive personal information from participants.

Usability and Cost:

A presentation may require on-chain processing at the time of disclosure by the subject. Alternatively, a self-contained presentation can be disclosed by the subject without interacting with the blockchain. The relying party receiving the self-contained presentation may still need blockchain access to process and verify it. These considerations result in solutions with varying usability and costs. Some actions can be achieved off-chain, quickly, and at no cost. Other ones may be free of cost, but require access to the published blocks. Finally, actions may be delayed by transaction processing time and induce costs for paying the blockchain miners to process a transaction.

In the case of smart contract based systems on top of permissionless blockchains, third-party entities may pay smart contract transaction fees on behalf of the users so that they do not have to deal with holding and spending the native digital currency of the blockchain themselves.

4.4 System Architecture Designs

This section focuses on the architectural design options that can be made when building a blockchain identity management system. Pieces of the system can be constructed as distinct modules or can be combined into monolithic architectures, although some designs are mutually exclusive. Note that some of them rely on on-chain registries and logic (generally implemented in the form of smart contracts on a blockchain that can be, depending on the purpose, permissionless or permissioned) that may be augmented by system-specific off-chain schemes.

We first discuss architectures for identifiers, then credentials, and finally, more complex combinations of architectures for identifiers, credentials, or both.

4.4.1 Identifier Architectures

This section discusses the technical means to implement the identifier origination schemes introduced in Section 4.1.2.

[5] For example, [35] presents a system built atop Bitcoin that uses Brands' commitment scheme to let users selectively disclose their credentials via zero-knowledge proofs.

4.4.1.1 On-chain Registry

Credentials Registry Acting as Identifier:

For each identifier participating in the system a dedicated smart contract is deployed that can store credentials for that identifier. This architecture typically follows a bottom-up authority model approach, which is well-suited to permissionless blockchains in order to foster greater decentralization. The deployment of a new contract for every identifier allows participants to own their own contract, and thus, have control over their own identifiers. This comes at the expense of higher cost since many contracts must be deployed, more data must be posted on a blockchain, and there may be slower processing speeds due to the number of transactions on a blockchain. These aspects can hinder scalability and there are possible interoperability issues if different identity management contracts are deployed by different users (or simply different versions of the same contract). This may be mitigated by using standards such as ERC-725 *Proxy Account,* a proposed Ethereum standard that follows this architecture. It allows other smart contracts to take action based on verifiable identity information contained in ERC-725 smart contracts. In addition, ERC-734 *Key Manager* can complement them by allowing subjects to delegate certain capabilities to custodians of their choice.

Global Identifiers Registry:

A single monolithic smart contract, or set of integrated contracts, is deployed that acts as a global registry for storing and managing all identifiers. It is logically centralized but physically decentralized to the extent that the blockchain nodes are distributed. This approach can follow either the top-down or bottom-up authority models, as the logical centralization does not imply control by a single entity. The writer of the smart contract can encode a variety of possible governance models. This can range from the entity deploying the contract having complete control of the system, having only limited control of it, or having no control of it. In the case of no control, the governance of the contract would be run by participating users (e.g., with a DAO). The registry can contain all the necessary logic and data to resolve identifiers to their metadata (e.g., DID documents when the DID specification is followed) or may contain only hashes which are mapped to the actual metadata stored elsewhere.

Anchors Registry:

A single monolithic smart contract is deployed that acts as a global registry that registers the hashes of identifier management operations that are grouped together into bundles, or "anchors". The bundling (grouping) of identifier management operations is executed by a second layer protocol that sits on top of the blockchain to which the anchors registry is located. The protocol then adds the hashes of those anchors in the registry, and uses decentralized storage systems such as the Inter-Planetary File System (IPFS) [36] to store the anchors data (identifier management operations). The Element [25] decentralized identifier system based on the SideTree protocol (second layer protocol) on top of the Ethereum blockchain follows this architecture.

Note that an anchors registry (coupled with a second later protocol) may be used for any on-chain registry (e.g., one that supports credentials).

4.4.1.2 Bring-your-own Blockchain Address

Any blockchain address is a valid identifier and can be immediately used without having to be registered beforehand. Identifier creation and storage is usually done locally in the identity wallet. This architecture follows a bottom-up authority model where the user is self-reliant; identifier creation takes place offline without any gatekeeper, and at no cost.

Identifier management (by the subjects) and use (by the verifiers), however, may require on chain capabilities. This differs though from the identifiers registry smart contract architecture because identifiers are initially not registered and stored on-chain, making them non-discoverable by default.

This architecture may help the system operate at scale since no blockchain transactions are needed for initial identifier creation. Users control their identifiers, as with the per-identifier smart contract architecture, and may gain privacy advantages as identifiers need not be publicly viewable. Moreover with identifier creation being cheap, users may utilize pairwise pseudonymous identifiers (or unique, one-time identifiers) to enhance their privacy when interacting with relying parties (see *Pairwise-pseudonymous and Single-use Identifiers* in Section 4.3). On-chain logic may be necessary to implement additional functionalities such as identifier management capabilities. For example, it will be needed to access the chain to resolve an identifier. The information necessary to do this must be stored on a blockchain and, likely, managed through some smart contract.

ERC-1056 *Lightweight Identity* is a proposed Ethereum standard that follows this architecture and that is used by uPort [37]. DID operations are stored in the form of Ethereum events. Resolving a DID to its DID document consists in iterating over the DID operations that may have been posted by the subject. Note that protocols that define and implement DID methods to build DID documents for bring-your-own blockchain identifiers may be further developed in a way to interact with multiple blockchains. Note that, in the case of Ethereum, blockchain log data cannot be queried from other smart contracts; however, an external method can be designed to access the chain and iterate over the logs to build a document (as in uPort).

4.4.2 Credential Architectures

This section discusses architectural designs for storing and managing credentials. The choice of design may depend on how identifiers are managed. The credential can be stored on-chain or off-chain.

On-chain credentials often only require on-chain storage for the hashes of the credentials, with the non-hash data being stored on any data store a subject has access to, be it a designated custodian or a decentralized storage system such as IPFS [36].[6]

[6] In addition to using IPFS with an on-chain pointer, the research literature has demonstrated a number of designs for how to store credentials (and other data) off-chain securely. For example, [38] uses a blockchain for enforcing access control policies on an off-chain data store, where the off-chain data store is implemented as a distributed hash table (such as Kademlia). An alternative system, described in [39], uses centralized and decentralized databases linked together by a blockchain in order to

The integrity of the data may be checked by the receiving party by hashing the credential and comparing the hash with the one found on the blockchain. Note that the hashes are often stored either in the form of state variables or in the form of blockchain logs, the latter being sometimes cheaper than on-chain storage (e.g., Ethereum events).

Credentials can also be stored fully off-chain, either directly on the subject's device and/or by a designated custodian. There may still be, however, additional mechanisms to handle revocation.

There are usability, privacy, and security issues related to where credentials are stored and how they are managed.

4.4.2.1 On-chain Registry

Per-identifier Credentials Registry:

In this architecture, credentials are managed as entries in a per-identifier smart contract that acts as a container as defined in Section 4.4.1.1. This architecture can give the subject unilateral control over their credentials. As owner of the contract, a subject can remove any credential they want without the approval of the credential issuer. Also, their approval is required, in addition to the one of the credential issuers, for the issuance of a credential (see Section 4.1.3).

While subjects can manage their own on-chain credentials in this way, this architecture is heavily reliant on on-chain transactions. This can hinder system scalability due to blockchain transaction costs and the relatively slow processing speed for transactions. The architecture thus can make it expensive to use privacy features such as pairwise pseudonymous identifiers for every relationship (see *Pairwise-pseudonymous and Single-use Identifiers* in Section 4.3). ERC-735 *Claim Holder* is a proposed Ethereum standard that follows this architecture and can be utilized jointly with ERC-725 *Proxy Account*.

Global Credentials Registry:

In this architecture, credentials are registered and managed as entries in a single smart contract. It is logically centralized for the entire system but physically decentralized to the extent that the blockchain nodes are distributed. Usually, the identifier that deployed the contract initially owns the system. However, that authority can be delegated, transferred, or limited depending on how the contract is coded. Thus, this architecture requires the initial owner to set up a governance model that establishes the rules and permissions for managing credentials. This may necessitate handling concepts such as reputation and negative credentials (see Section 4.5). Credential management involves on-chain transactions and access, which impacts the usability and cost of presentation disclosure as discussed in Section 4.3, as well as privacy as discussed in Section 5. This architecture can be used as a registry for revoking credentials. A relying party then is able to verify the validity of the off-chain credential.

---

allow users to exclude others from using their data, while still allowing the data to be searchable (which can be useful for areas such as medical research). Finally, Calypso [40] is a more advanced construction with auditable access control, which uses threshold cryptography to protect access to data.

Another use for this architecture is to allow a user to publish credentials about themselves and share information publicly such as a public key or a service endpoint.[7] ERC-780 *Ethereum Claims Registry* is a proposed Ethereum standard that follows this architecture. ERC-1056 (see Section 4.4.1.2 on *Bring-your-own Blockchain Address*) also implements a credentials registry, although it is limited to self-issued credentials (and is based on blockchain logs).

Non-fungible Token Registry:

In this architecture, a credential takes the form of a Non-Fungible Token (NFT). An NFT is a unique, not interchangeable token that is owned and may be managed and traded. Minting and management of the tokens are performed through a NFT factory smart contract (that acts as a registry that manages the NFTs). NFT-based credentials primarily aim at fitting use cases that deal with digital ownership, especially, but not exclusively, when it is meant to be transferable (see Section 4.2.2 on *Custody and Delegation*). The minting of specific tokens can implement application-specific token formats, rules, and requirements and therefore provide token lifecycle management capabilities. In addition, this architecture can use interoperable token formats thus enabling a marketplace for transferable credentials. NFTs can either be issued individually or to a group (a distribution method also called "airdrop"). These capabilities come at the expense of the need for participants to issue blockchain transactions and have blockchain access (See *Usability and Cost* in Section 4.3).

The ERC-721 *Non-fungible Token Standard* is a proposed Ethereum standard that follows this architecture. As an example, 0xcert provides a framework for building decentralized applications that aim at creating and managing ERC-721-compliant NFT-based credentials [42].

Entitlement to a User-mintable Non-fungible Token:

In this architecture, a credential takes the form of an entitlement to let a user mint a pre-defined and pre-assigned NFT at a future date or condition.

This can be achieved through system-specific NFT factory smart contract designs. As an example, Centrifuge [43] allows one to turn credentials, of which the hashes are stored on-chain, into NFTs. The proof that one is entitled to mint a given NFT is verified through the Merkle root hash (stored on-chain) of some of the off-chain credential data.

This may also be achieved for a group of subjects through the use of a Merkle airdrop (see definition in Glossary in Appendix B), which allows group distribution of the entitlement to redeem an NFT. This scheme is highly scalable in that it requires only one transaction by the issuer and is independent of the size of the group. No management support is needed after the distribution as all of the activity comes from the subject side.

[7] Advanced cryptographic primitives, such as the hash-based accumulator employed in [41], can allow a registry to retain a constant-sized storage regardless of how many credentials are registered.

A credential is private by default, and a subject can redeem it only if they want to use or transfer it. However, the list of all the identifiers the Merkle airdrop was issued to must be available to the subjects to redeem their NFT (both the private key and the list of all the identifiers included in the Merkle airdrop are needed to build the Merkle proof and mint the NFT). Note that for a Merkle airdrop, the tokens must be "pulled" by the users, while for a traditional airdrop, the tokens are "pushed" to the user and even those who do not want to receive them.

4.4.2.2 Off-chain Object

In this architecture, a credential takes the form of an off-chain object that acts as a self-contained vehicle for transmitting information directly between parties. This can go hand in hand with the *Bring-your-own Blockchain Address* architecture discussed in Section 4.4.1.2 to establish a lightweight identity management system that can operate at scale. It best matches use cases where the lifecycle of a credential is predetermined. However, verification of a credential (see *Lifecycle Determination at Origination* in Section 4.2.1) may require chain access (see *Usability and Cost* in Section 4.3). In particular, if revocability is permitted, on-chain artifacts are required for one to check if the credential was revoked, such as with a credential revocations registry (see *Off-chain Objects coupled with Global Credentials Registry* in Section 4.4.3).

It can provide a high level of user control as the subjects own their own credentials. It ensures privacy by default and need not be constrained to a specific blockchain. This architecture may use, for example, the JWT format (see Section 3.4 on *Building Blocks*), as in Blockstack [44].

4.4.3 Combination Patterns

It is possible to combine the architectures for identifiers, credentials, or both. This section provides some examples of how this is being done, but is not exhaustive.

Global Identifiers Registry coupled with Per-identifier Credentials Registry:

An IDMS can be designed so that identifiers are stored in a global registry, but each identifier has their own dedicated smart contract for storing and managing credentials.

The Smart ID project from Deloitte [45] follows this architecture. Note that the global identifiers registry may also serve as a smart contract factory to create and manage all of the per-identifier credentials registry smart contracts.

Global Registry for Both Identifiers and Credentials:

A single smart contract can implement both an identifiers registry and a credentials registry as described in Section 4.4.1.1 and 4.4.2.1.

This approach is followed in *Smart Contract Federated Identity Management without Third Party Authentication Services* [28]. Another example of this approach is SCPKI (Smart Contract-based PKI) [46], which stores all identifiers and credentials on a single smart contract, and allows relying parties to use a web of trust to decide whether or not an identifier is authorized to perform some action. SCPKI can also be extended with blind signatures in order to provide privacy [47].

Another example is that of BlockPKI [48], which can generate one or more smart contracts per identifier in the system. These per-identifier contracts (called "certificates") contain a set of credentials and are used to store signatures from certificate authorities; once enough signatures have been gathered in a contract, they are aggregated and then sent along with the certificate data to a global credentials registry contract. Relying parties can use this global credentials registry to verify signed certificates in the system.

Off-chain Objects coupled with Global Credentials Registry:

Off-chain objects can be used as the primary way to issue and share credentials while relying on a central registry smart contract to publicly store the service endpoint URLs and public keys necessary for the participants to discover and authenticate one another.

A credentials registry can also be leveraged to act as a revocation registry for off-chain credentials. Such a registry is used in both uPort [37] (it is based on ERC-780 and deployed on the public Ethereum blockchain) and Hyperledger Indy [22]. In the latter, an issuer can control a revocation registry that relies on a cryptographic accumulator (protocol that allows one to prove a membership in a set; see Section 6.2 on zero-knowledge protocols) to let relying parties verify whether a given credential was revoked by the issuer or not without compromising the privacy of the registry.

Off-chain Objects coupled with Global Identifiers Registry for Issuers:

Issuers have their identifiers stored on an on-chain registry. They can issue off-chain credentials directly to any blockchain addresses controlled by the subjects. Verifiers only need to verify that the signatures of the credentials issuers match those on the on-chain registry.

Non-fungible Tokens with Global Credentials Registry:

Rules and permissions based on a central registry, which may be implemented in a smart contract, can be implemented to restrict the context in which transfers of NFT-based credentials take place (if they are allowed). This way, parties that trust each other can transact securely and according to the agreed-upon rules.

This can be leveraged, for example, to establish Know-Your-Customer (KYC) checks for exchanges of tokens as in the Transaction Permission Layer Protocol [49] with the ERC-1616 *Attribute Registry* Ethereum standard proposal.

4.5 Public Registries and Reputation Management Implications

Some blockchain IDMS architectures rely on on-chain registries, and therefore, may have publicly readable data stored in a central location (e.g., smart contracts). This can be leveraged by subjects wanting to share public information about themselves (e.g., a service endpoint at which they can be reached if they wish to be discoverable). It can also be used by organizations wanting to build reputation systems such as public institutions (e.g., TheOrgBook project of the Government of British Columbia [50] running on the Verifiable Organization Network, "a public repository of verifiable claims about organizations") and e-commerce platforms (e.g., product and seller ratings).

The public centralized architecture does not necessarily imply that the user privacy is violated or that users do not have control over their identity. Schemes may use hashing or encryption to protect publicly posted data and varying degrees of granularity can be implemented enabling users to manage their own credentials and associated reputation. One important design feature is whether or not user consent will be required prior to a credential being issued to that user; the user may view certain claims about themselves as being negative and not want them published. Some systems allow unilateral claim issuance while others require user approval. If the user can not stop the claim from being issued, they may then want to get a counter-claim issued. A reputation system may be used to track the reputation of issuers, which verifiers then can evaluate. Note that such systems must protect themselves from, and may be subject to, attacks designed to inappropriately alter user reputation.

Sybil Attacks and Structural Barriers:

Reputation systems need to protect against Sybil attacks, where an attacker pretends to be many people at once, by imposing a structural barrier. For systems with access control (that may sit on top of either a permissionless or permissioned blockchain), it can take the form of identifier verification and the use of roles and permissions (e.g., TheOrgBook [50]). For open systems, the structural barrier can be made of a cost to register, exist in, and/or exit the system. This makes attacking the system disproportionately expensive compared to the benefits the attack would produce. While transaction fees act as a basic cost structure, more advanced ones relying on game theoretic concepts can be designed to achieve objectives such as disincentivizing participants from leaving an identity to regain newcomer status and ensuring participants do not get an advantage by issuing multiple identities.[8]

An example of such a cost structure are “token-curated registries”, which feature an incentivized voting game to let a community of participants decide whether an entry should be added or removed from the registry. These Sybil-resistance mechanisms can be based on staking funds (e.g, with collateral and/or escrow contracts), reputation, or work (committing a certain amount of resources for a certain period of time).

4.6 System Governance

Blockchain-based IDMSs must have a governance structure that makes the system trustworthy to its participants. Approaches can vary significantly, and often involve a combination of both on-chain and off-chain organizational structures.

The on-chain structures can consist of smart contracts deployed on some underlying blockchain (either permissioned or permissionless); users are thus required to trust both the governance models of the smart contract-based system and the underlying blockchain. Alternatively, solutions exist where a blockchain is developed and deployed for the sole purpose of supporting an IDMS, called “application-specific” blockchains.

[8] [51] describes three other types of generic attacks against a reputation system - bad-mouthing, ballot-stuffing, and whitewashing, and proposes a blockchain-based solution to mitigate them. [52] is another blockchain-based reputation system designed for reputation in file-sharing networks or for e-commerce, while [53] aggregates social media reputation.

There may be security tradeoffs between these approaches. If the blockchain is not application-specific, governance of the blockchain itself is an important topic but not the focus of this paper, which examines the identity application specifically (a few applicable considerations are provided for reference in Section 6.1 on *Underlying Blockchain Considerations*).

A set of the higher level recurring governance traits are discussed below.

Ownership and Funding:

A system can be owned by a for-profit organization (e.g., a company), a non-profit organization (e.g., a foundation), a consortium, a government agency, an open-source community, and/or a DAO.

It can be directly financed through traditional fundraising and monetized by the entities that administer it. It can also rely on crowdfunding, through an Initial Coin Offering (ICO) for example. Note that token-holders are not necessarily share-holders of the system in that the tokens may not give any piece of ownership of the system. Finally, the system may have no dedicated funding at all, and be maintained solely on a volunteer basis by the members of the community.

Operating Model:

An IDMS can be designed and administered as a permissioned system to meet the internal needs of the members of an organization or a group of organizations. This means that only an approved set of users may access and maintain the system.[9] This permissioned system might be offered as a proprietary service to customers or it might be deployed internally. Access control takes place either at the smart contract level (that sit on top of an underlying blockchain) and/or at the blockchain protocol level (i.e., a permissioned blockchain).

Note that all permissioned blockchains require identity management systems to determine who the validators are (for example, in a proof of authority consensus model). This may take place off-chain (typically the validator nodes have a list of the other nodes that they want to connect with), or via smart contracts on-chain. Changes to the list of validators may then be administered through on-chain voting by administrators.

Alternatively, an IDMS may form an open protocol and/or ecosystem that can be used and integrated by anyone. It can be a general-purpose ecosystem, or an application-specific one (e.g., credit scoring with Bloom Protocol [55]). Furthermore, an IDMS can involve users authenticating at the application level, or at the ecosystem level such as in Blockstack [44]. The latter differs from traditional "single sign-on" identity management in that identifier origination, credential issuance, and presentation disclosure are not necessarily controlled by a single entity.

---

[9] A second layer protocol can be used as an access control mechanism for permissioned blockchains. For example, the ChainAnchor scheme [54] offers this, while allowing users to transact pseudonymously and maintain transaction unlinkability: users can selectively disclose their transactions if asked to (e.g., for regulatory purposes) without revealing their other transactions. This scheme makes use of the "Enhanced Privacy ID" zero-knowledge protocol.

In some systems, tokens may be utilized to design an incentive structure and boost certain desired behaviors from the participants (e.g., through earning rewards) to facilitate ecosystem coordination, self-sustainability, and growth (it can be based on various game theory techniques). The incentive structure can be extended to built-in monetization schemes to buy and sell services. More specifically, they may be coded directly as part of the functions that implement actions such as credential issuance and presentation disclosure.

Internal Rules Management:

Every system will have rules that dictate how participants interact with a given system. These rules are often implemented and enforced through smart contract code that is visible to all participants. Since the underlying blockchain enforces correct execution of the smart contract, users can trust that these rules will be executed correctly. These rules may also specify how changes to the rules themselves are managed (e.g., how the system is upgraded). Allowing such rule changes may prove beneficial - even necessary - for mitigating security issues or adding new features. However, allowing arbitrary changes can hinder user trust in the system, especially changes done without user consent. Thus, the upgradability of these systems can be treated carefully so that expectations regarding the immutability of contracts remain valid [56]. It may be important that there exist platforms to communicate and facilitate decision-making among stakeholders of a system (e.g., to raise awareness of the desired benefits and the associated risks of a certain proposal).

The modifications to the smart contracts can be actively governed by the system's users through a voting system (like Bloom's polling mechanism [55]) or through a Decentralized Autonomous Organization (DAO). The modifications may also be enforced with a time delay to let participants opt-out of the system if they are not satisfied with the rule changes. Lastly, it is possible that a system may have multiple versions live simultaneously (for example, both the upgraded and non-upgraded versions). This allows participants to opt into updating to the new version. Finally, note that time-stamped entries in an on-chain public registry (immutable and tamper-resistant) can facilitate accountability by serving as support for posting update proposals using accounts with identifiers registered in the system.

Software Management:

The management of the software for a system is a vital governance issue as the software implements the rules and maintains the system, but also provides the users' portal into the system (e.g., in the form of decentralized applications, sometimes called "dapps"). Blockchains can provide significant security advantages for identity management systems, but if the user software is vulnerable, corrupted, or malicious these protections mean little.

The software can be managed by the developers as an open-source project shared publicly on a version control platform such as Github, or the software can be proprietary. Development patterns can be leveraged to enable smart contract upgradability (e.g., a registry contract that points to the latest version of the main contract of the system or an interface contract that is inherited by the system and defines a set of key functions and parameters). Periodic third-party audits, automated tests, and reports can also be performed and disclosed to help assess whether the rules are properly enforced.

External Influences:

A given blockchain-based IDMS can be subject to external influences (that may depend on its operating model) such as:

- Regulatory compliance requirements (e.g., the European Union's General Data Protection Regulation), and law enforcement.
- Industry alliances (e.g., the Ethereum Enterprise Alliance, Hyperledger Foundation, Decentralized Identity Foundation, Trusted IoT Alliance) and standards bodies (e.g., International Organization for Standardization (ISO), Internet Engineering Task Force (IETF)) that publish specifications, formats, protocols, and patterns.
- Peer-reviewed research and bug bounty programs.
- Social norms and user expectations.

A key implication is that they introduce a certain framework of disclosure and transparency, which might directly affect or even require certain protocol designs. This may help participants be aware of, supportive of, and ideally, educated about, the rules of the platform. Community expectations may play a significant role in holding the administrators of a system accountable (especially if the community has the means to opt-out at a reasonable cost and to port their accounts to another provider).

5 Security and Risk Management

Blockchains can provide security advantages to a variety of applications by removing or reducing the need for trusted third parties. Second layer protocols can add more flexibility and may help better scalability, privacy, and interoperability. These foundational building blocks can provide enhanced integrity and resiliency. However, blockchains do not solve all security issues, and careful examination of the risks and challenges of blockchain usage is needed.

Some of these issues and associated mitigations are discussed below.

Private Data Leak:

When a user shares personal data with a relying party, the relying party may share that data outside of the context of the IDMS. This is a significant problem for any identity management system where user personal data is shared. However, this can be minimized by the use of minimal presentation disclosure mechanisms. For example, zero-knowledge protocols may be utilized to share presentations that contain only the necessary information for a given interaction to relying parties rather than full credentials.

Separately, architectures that put less data on-chain may in general be more privacy preserving, but it depends on the exact architecture being used and how that data is being stored (e.g., unencrypted, encrypted, pointers to outside repositories, or hashes). Finally, vulnerabilities may be found in the authentication and messaging protocols used by a given system to support peer-to-peer data transmissions.

Metadata Tracing:

Pattern analysis techniques may be applied by attackers to on-chain metadata and possible interceptions of messages between parties. They may look at, for example, the time that transactions or credentials were submitted to the blockchain, which issuers signed them, or the IP addresses that they were broadcast from. This information may be leveraged by attackers to compromise the confidentiality of Personally Identifiable Information (PII). This correlation risk can be minimized by decoupling users from a unique persistent identifier through the use of pairwise pseudonymous identifier (or more advanced identifier unlinkability techniques). Zero-knowledge proofs may also be used to obfuscate the details of blockchain transactions.

Replay Attacks and Impersonation:

A rogue relying party can attempt to collect user credentials and presentations in the aim of fooling another relying party into believing that they are that user. This kind of man-in-the-middle attack can be mitigated through relying parties using certain challenge response protocols and encrypted tunnels such that the subjects must always prove their identity (that they know the private key for the identifier associated with the transaction).

Private Key Compromise:

In most IDMSs, knowledge of a private key for an identifier is equivalent to owning the identifier. Thus, preventing the compromise of private keys is essential. Keys can be compromised due to errors in key generation, storage, or use, or can be stolen by malicious actors. Human errors can be mitigated through well-designed tools for key management and secret sharing (typically that is a user-friendly identity wallet); as discussed in [57], a system may be secure only if it is usable. Once lost or stolen, identifier recovery mechanisms may be implemented to enable a subject to regain control of an identifier (see Section 4.4.2.1). In general, architectures that provide more privacy may reduce the risk of being targeted and having private keys stolen.

Data Withholding Attacks and Data Availability Issues:

When users manage their identifiers and credentials themselves, they benefit from a high-level of autonomy and can ensure the availability of their data. An alternate approach is for users to choose to rely on custodians to hold and manage their data for convenience. However, custodians can misbehave, compromising the ability of the user to access their identifiers and credentials. Although proper delegated control restrictions can help constrain such a rogue custodian, this does not prevent data withholding attacks. Even a well-behaved custodian can experience temporary service disruptions (or even go out of business), thus making user data unavailable.

Therefore, it may be important for a subject to implement data redundancy by storing multiple copies of identifier and credential data in locations that are either directly controlled by the user (such as identity wallets across different personal devices) or delegated to custodians with proper access and control permissions in place. This could involve identity hubs as mentioned in Section 3.3 on *Emerging Standards*. Note that these are issues with traditional IDMSs and that the use of blockchain can be seen as a potential improvement.

Quantum Computers:

Blockchain networks depend on cryptography for their security, in particular, on public-key cryptography. If a sufficiently powerful quantum computer is built in the future, the most widely used public key cryptographic algorithms in blockchain systems will become insecure. This represents a long term concern for identity data stored on a blockchain. Note that this concern applies to the entire Internet; it is not just a concern for blockchain technology.

Smart Contract Flaws:

The smart contracts implemented to support the blockchain-based IDMS may have security flaws. Such contracts are usually short and concise, but nonetheless there have been flaws discovered in published smart contracts that enabled them to be compromised.

Audits, tests, and the use of well-audited libraries can help mitigate this risk. Furthermore, data integrity at the smart contract level may be achieved by establishing permissions to prevent unauthorized participants from accessing and modifying user identifiers and credentials.

System Governance Design Flaws:

Some blockchain identity management system architectures (e.g., top-down authority models) may incorporate logic that creates single points of failure. For example, they may provide a certain type of participant a high level of privilege that could be improperly used.

This can be mitigated against by instituting appropriate separation of authorities between participants along with a security analysis of the system to identify single points of failure with respect to bad actors in the system. Furthermore, governance architectures that rely on game theoretic incentives have their own risks (e.g., see Section 4.5 on *Public Registries and Reputation Management Implications*).

Oracles and Second Layer Protocol Compromise:

A blockchain IDMS may integrate off-chain data, logic, and processing in the form of oracles and second layer protocols. Should they get compromised, the on-chain part of the system may not be able to identify the threat adequately and cope with the compromised data, resulting in a "garbage in, garbage out" situation. It is therefore important to ensure that necessary checks and balances are in place.

6 Additional Considerations

This section provides additional considerations regarding some of the fundamental topics of blockchain identity management discussed previously.

6.1 Underlying Blockchain Implications

Blockchains have unique properties and underlying governance implications that must be considered while designing an identity management system or deciding on one to use. *Blockchain Technology Overview* NIST-IR 8202 [11] in Section 7.2 *Users Involved in Blockchain Governance* states: "the software developers, publishing nodes, and blockchain network users all play a part in the blockchain network governance". Below are some key considerations.

Data Persistence and Privacy:

Any data added to a blockchain will be available permanently. This can have substantial ramifications for privacy in multiple ways:

- If personal information is encrypted and then stored on a blockchain, confidentiality for that data will be lost if the encryption algorithm is broken.
- Over time, as more and more individual metadata is shared with various relying parties and credential issuers, it can be correlated with on-chain data in order to link users and their activities (see *Metadata Tracing* in Section 5 on *Security and Risk Management*).

While the effects of metadata tracking in these systems requires more study, the permanence of blockchain data will affect anyone who uses a blockchain-based IDMS. However, note that there are systems being developed and implemented into production that may allow the building of finer privacy solutions.

Consensus Algorithms, Time Delays, and Data Integrity:

Working with blockchains means that their operations rely on distributed consensus algorithms. There are a wide variety of consensus algorithms – including both permissioned and permissionless ones – and they have different properties that may be important to schemes built on top of ledgers that use them. A consequence of this is that a scheme built on top of blockchain A may have different security, integrity, and usability considerations than an otherwise identical scheme built on blockchain B.

The simplest example of this is the expected delay between broadcasting a transaction and having it included in a block. Permissioned consensus algorithms tend to find blocks within seconds, whereas the Bitcoin network, for example, experiences an approximately 10-minute delay between finding new blocks. If an on-chain claim were issued on Bitcoin, it could take an hour or more before it is recognized by relying parties. Verifiers often need access to this blockchain data to compare revealed information against public hashes of that data or query an on-chain revocation registry. The time delay for releasing blocks, or for reading and processing newly published ones, can affect the view the application has of the current data.

Blockchain Forks:

Another potential issue is that of chain splits, such as that which occurred between Ethereum and Ethereum Classic. When some kinds of disputes arise between users or stakeholders in a blockchain system, a single chain can split into two chains with a shared history up until the point of the split. If a smart contract existed on the chain prior to the split, it will have its state, history, and logic copied to both chains. This can cause confusion for users, especially during the time around the split. It may present further issues, such as replay attacks, such that a transaction that is valid on one chain is also valid on the other – even if the transaction is only intended for a single chain. This may require relying parties and users to monitor both chains for some period of time.

Blockchain Resiliency:

As NIST-IR 8202 states, "Traditional centralized systems are created and taken down constantly, and blockchain networks will likely not be different. However, because they are decentralized, there is a chance that when a blockchain network "shuts down" it will never be fully shut down, and that there may always be some lingering blockchain nodes running. A defunct blockchain would not be suitable for a historical record, since without many publishing nodes, a malicious user could easily overpower the few publishing nodes left and redo and replace any number of blocks."

For an IDMS built on top of an underlying blockchain, it is important to carefully monitor the validators' activity and to establish security thresholds and metrics to ensure that the increased risk of attacks on a declining blockchain are understood and considered acceptable. When a blockchain is deemed insecure, an identity management system may be migrated to a more secure one.

6.2 Introduction to Zero-Knowledge Protocols

Zero-knowledge protocols (abbreviated ZK protocols, or ZKP) can play a fundamental role in blockchain-based identity management systems for transaction confidentiality, user identification, and presentation disclosure. Credentials can be taken as input to build presentations using zero-knowledge proofs, which allow subjects to control the amount of information disclosed to relying parties and the context the presentation takes place in (see Section 4.3 on *Presentation Disclosure*).

The notion of zero-knowledge was first introduced in 1985 [58] and has since evolved into a class of algorithms with several practical applications [59, 60]. This section presents a high-level overview of zero-knowledge protocols and their role in identity management. We encourage the reader to explore specialized publications such as [61] to gain a deeper understanding of zero-knowledge protocols. Note that ZKProof.org is an initiative led by industry and academia to standardize the use of zero knowledge proofs.

Definition and Properties:

There are at least two parties in a ZK protocol: a prover and a verifier. The prover aims to convince the verifier that a statement is true without revealing any additional information. There are four common statement types, though the following is not an exhaustive list:

- An equality statement (the subject's bank account balance is equal to x), or non-equality statement.
- An inequality statement (the subject's bank account balance exceeds x).
- A range statement (the subject's bank account balance is within interval $[a, b]$), or out-of-range statement.
- A membership statement (the subject is on the client list of bank X), or non-membership statement.

Generally, there are two kinds of ZK protocols: interactive and non-interactive. In an interactive ZK protocol, the prover and verifier engage in at least three rounds of communication exchange. Such protocols permit the verifier to submit challenges to the prover, whereby the prover replies with responses that reinforce the validity of the prover's original statement. There is no challenge-response interaction in non-interactive ZK protocols, though there is sometimes a common reference string shared in advance by both parties.

A ZK protocol produces a proof which is sent to the verifier. For statement S, prover P, and verifier V, the resulting proof π must satisfy the three following properties to be considered secure:

- Completeness: If S is true, then π will convince V that S is true with overwhelming probability.
- Soundness: If S is false, then the probability that P can convince V that S is true is negligible.
- Zero-knowledge: If S is true, then V learns nothing from π besides the fact that S is true.

The soundness property captures the inability for a prover P to convince the verifier V of a false assertion. If, for example, P can cheat with probability 1/3, then the ZK protocol may need to be repeated n times to reduce the soundness error from 1/3 to $1/3^n$.

The zero knowledge property can be statistical, or computational. If the verifier is assumed to have unlimited computational resources but learns no additional information from the protocol, then the protocol is considered to achieve statistical zero knowledge. If the zero knowledge property holds by some assumption about the verifier's computational power, then the protocol achieves computational zero knowledge.

Usability and Cost:

The scalability and cost of ZK protocols depend on the succinctness of the proof. It measures the required storage size of the proof, the proving time, and the verification time; these considerations are of special interest for blockchain-based ZKP schemes, with the blockchain having its own limited storage and transaction speed.

Note that the trusted setup phase that is required for some zero-knowledge protocols (e.g., the zk-SNARKs protocol implemented in Zcash [60]) involves a significant initial cost, but then enables verifications of the proof to require fewer resources (it allows a statement to be proven many times by verifiers that have limited time and resources).

6.3 Presentation Sharing and Data Mining

This section discusses protocols – such as those based on zero-knowledge – to control the context in which presentations may be used by relying parties for data mining and data exchanges with third parties (i.e., other relying parties). Note that they represent advanced research topics, and could trigger the emergence of novel data broker business models.

Convincing Power:

When a subject discloses a presentation to a relying party (as discussed in Section 4.3), information is revealed, that cannot be undone, and the relying party may share that information to other relying parties. However, in some schemes such as interactive zero-knowledge protocols, relying parties are, by design, unable to convince other relying parties that a statement (that a subject convinced them was true beforehand) is true. An interactive proof typically only convinces a single verifier that has established a direct and authenticated contact with the prover. In contrast, non-interactive protocols may convince multiple verifiers simultaneously, and possibly at a later date.

Schemes also exist where ZK protocols allow for privacy-preserving querying of credential revocation registries (e.g., some cryptographic accumulator schemes).

Benefits of Credential Properties:

Presentations can take the form of credentials to benefit from properties credentials have. For instance, a presentation may have the ability to be accessed conditionally (see *Entitlement to a User-mintable Non-fungible Token* in Section 4.4.2.1) and to be transferable (see *Non-fungible Token Registry* in Section 4.4.2.1). Such presentations can also be used to derive a limited number of presentations, like in the scheme described in [62].

Presentation Encapsulation:

A relying party that receives presentations may be able to encapsulate them into another presentation and disclose it to another relying party. In that case, the issuer of the encapsulated presentation is not the issuer of the original presentation. However, it allows the relying party to verify a snapshot of a presentation to another relying party (a timestamp and signature may be added).[10]

[10] Non-interactive ZK protocols (NIZK protocols) being potentially transferrable, the original verifier could turn around and claim the original NIZK proof as their own while interacting with third party verifiers. [63] provides a way to tie NIZK proofs to the identity of the original prover, such that when the original verifier presents it to a third party, the third party will understand that it was the original prover, and not the original verifier, who issued the original proof.

6.4 Ecosystem Convergence

A key catalyzer for the development of the decentralized identity management ecosystem is the development of standards, recommendations, and cross-ledger integrations. Contingent to this is the identification of criteria, patterns, and best practices to understand which architecture designs are relevant, depending on the use cases at stake, and how to assemble them into suitable solutions. This will help inform decisions on how to use an existing system as a service, integrate one to a given solution, or build and deploy a new one.

Universal Wallets:

Standards such as BIP-32 and ERC-20 facilitated the emergence of interoperable cryptocurrency wallets. Additionally, the ecosystem of password managers (for storage and management of traditional identifiers and credentials) can be seen as mature.

In this context, the different architecture designs and components discussed in this paper, and standards such as emerging Ethereum ones (e.g., ERC-1056, ERC-780, ERC-725, ERC-734, ERC-735), may facilitate the emergence of interoperable user-controlled identity wallets, which integrate identifiers and credentials, alongside cryptocurrencies and other digital assets.

This can create a layer of abstraction for the users, who could access and manage all their services and applications from a single identity management interface; these services may integrate and/or rely on different identity management systems. This can concretely take the form of a software suite with standalone applications and extensions for browsers and operating systems. It may serve as a gateway to interact with third party marketplaces, applications, or stores of applications. It can also integrate digital asset exchange platforms and identity management custodians to reduce the burden for the users and provide additional services.

As discussed in Section 3.4 on *Building Blocks*, custodial wallets are provided by a third party that controls a user's private keys. Additional cryptographic schemes can be used to choose a trusted third party that is not the custodian service provider itself. For example, ZenGo [64] has developed a wallet that uses threshold signatures to create two secret shares that take the role of a user's private key when they are combined (which controls assets and/or credentials). More shares may be used to create schemes that require more than one trusted third party.

Secret shares are also featured in the Horcrux protocol [65].[11] It uses the Biometric Open Protocol Standard [66][12] to power blockchain-based authentication with biometric information.

[11] In this protocol, biometric data is collected by a device owned by the user, then divided into multiple shares. One of these shares is sent to a dedicated server, which selects a blockchain, creates a DID for the biometric share and stores the resulting DID document using off-chain storage providers. The other biometric shares can similarly be assigned to other blockchains, creating more DIDs. As a result, the original biometric data can act as a junction between different identity management platforms. This can help create more robust, blockchain-agnostic solutions.

[12] The Biometric Open Protocol Standard (BOPS) was introduced by IEEE under reference 2410-2018. It provides a framework to support biometric authentication methods. This standard also offers guidance for identification, access control, and auditing capabilities. Dedicated Application Programming Interfaces (API) designs, device requirements, and security and privacy considerations are also introduced.

Cross-Ledger Integration:

There are several ways blockchain-based identity management systems can integrate with one another and/or be part of a common larger structure:

- Universal resolver: As mentioned in Section 3.3 on *Emerging Standards*, the blockchain agnostic Universal DID Resolver maintained by the DIF allows the integration of any identity management system, which can then be queried by the users through a common interface.
- Second layer protocols: As mentioned in Section 3.4 on *Building Blocks*, second layer protocols such as the SideTree protocol [23] may also be used to interact with one or more blockchains simultaneously and in a blockchain agnostic manner.
- Bridges: The capabilities of a given system may be integrated in another system by implementing the libraries provided by the former system in the form of on-chain logic (e.g., smart contracts) in the latter system. For example, Cordentity [67] is a Corda smart contract that integrates Hyperledger Indy capabilities in the Corda platform. Thus, Corda ledger transactions can be contingent on credentials managed with a Hyperlerdger Indy-based blockchain. In addition, SecureKey has explored integrating Hyperledger Indy capabilities in Verified.me, its Hyperledger Fabric-based identity management system [68].

7 Use Cases

There are many uses for blockchain identity management, which can be intended to be public facing, privacy-preserving (to provide solutions for individual users), or both. They include financial services, reusable identities to support Anti-Money Laundering (AML) and KYC laws, verification of certificates, traceability of assets, and supply chain management.

These uses can be relevant for applications in various areas such as:

- Education: for the issuance of transcripts, diplomas, and certifications that can then serve as verified credentials during job applications.
- Healthcare: for the issuance of prescriptions, submission of claims to health insurance, and sharing of health records.
- Banking: for account opening, fraud prevention, proof of funds, credit risk evaluation, as well as ownership, exchange, and trading of financial assets.
- Government services: for the issuance of driver's licenses and birth certificates, maintaining public registries of voters.
- Public safety: for managing sets of equipment and reliable communication permissions.
- Manufacturing: for representing ownership of 3D models.
- Transportation: for the identification of autonomous vehicles.
- Data brokerage: for exchanges of datasets.

We provide below two use cases in the aim of further assisting the reader in their understanding of blockchain identity management.

Renting a Vehicle:

In this use case, we consider an individual that proves to a car rental company that they meet all the requested requirements without disclosing more information than what is strictly needed.

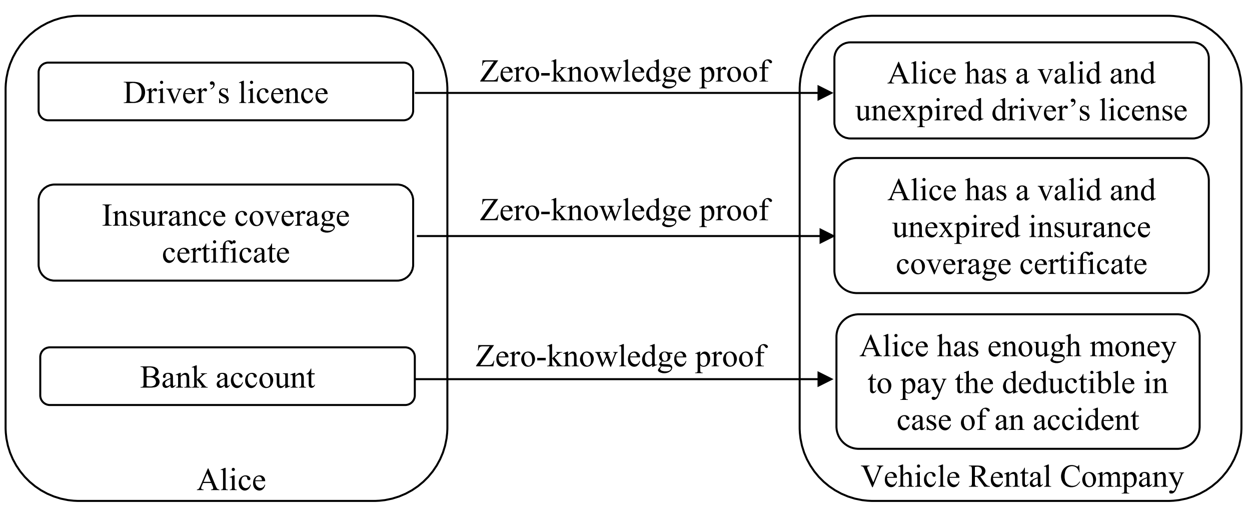

Figure 8: Minimal Disclosure to Rent a Car

This takes place through a system that enables the individual to build and disclose proofs derived from credentials that they own on a given IDMS. The credentials are: an unexpired and valid driver's license, an insurance certificate (showing that the individual has sufficient coverage), and a bank statement (showing that the individual has the means to pay the deductible in case of an accident). Rather than sharing the credentials in their entirety to the rental company, the presentation built by the system allows the individual to combine the derived information from each credential (as shown in Figure 8) and proves that the individual meets all the requirements. It may not even be necessary to disclose the full name of the individual.

An alternative version of this scenario is that of an employee that rents a vehicle on behalf of the company that they work for. In this case, the company can delegate access to some of its credentials to the employee, so that information derived from these credentials can then be added to the presentation the employee discloses to the car rental company.

Exchanging Concert Tickets and Coupons:

In this use case, we consider a system controlled by a company that enables the issuance of tickets and coupons for concerts, conferences, and other events, while allowing the users to sell or exchange those tickets and coupons on their own.

The system is owned by a ticketing company that controls initial identity proofing and user registration. Once registered, event organizers can issue transferable tickets (in the form of non-fungible tokens) to registered users. Although the initial registration is controlled by the system owner, users can transfer tickets on their own (without any further approval being necessary from the system owner). For instance, a ticket owner may be able to exchange it for one at another date, give it to a friend, or even sell it. After attending a concert, an individual may keep the ticket as a souvenir and add it on social media to connect with other attendees and artists.

The system also implements a loyalty program to get rewards and attend other events. It periodically distributes redeemable coupons (in the form of a Merkle airdrop of non-fungible tokens) to the customer base that can be used to claim a discount to attend new events. While these coupons have an expiration date and were issued to a certain group of individuals, they are transferable. That way, an individual that receives a coupon can transfer it to a friend, thus allowing the event organizers to reach a wider target audience.

8 Conclusion

Blockchain-based identity management is an emerging field that holds great promise in providing improvements over the traditional and federated models currently in use. This paper provided the reader with a general understanding of the benefits, challenges, and opportunities of such systems. It discusses the foundational building blocks of blockchain identity management systems and the current standardization efforts. It then identified different system properties that can be achieved through different architectural designs using a taxonomic approach. The paper reviewed select security and risk management issues as well as other considerations. It finished with some example use cases highlighting the utility of these systems.

Of special importance, the paper discussed the ability for blockchain identity management systems to reduce, or even remove the need for a trusted third party in the authentication and credential passing process with relying parties. Many other capabilities can be built into these systems and this paper reviewed such improvements and the different architectures that can support them. Critical to many of these benefits are the related technologies of smart contracts to act as trusted third parties, the use of zero-knowledge proofs to avoid oversharing information, and second layer protocols to build more scalable and private solutions.

Despite having great promise, this field is still emerging and it is unclear if it will provide a usable, secure, and scalable replacement for today's non-blockchain identity management systems. If or when this happens, blockchain-based identity management systems would become a fundamental architectural component of tomorrow's Internet.

References

[1] Temoshok D, Abruzzi C (2018) Developing Trust Frameworks to Support Identity Federations. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Interagency or Internal Report (IR) 8149. https://doi.org/10.6028/NIST.IR.8149

[2] Lyons T, Courcelas L, Timsit K (2018) Blockchain for Government and Public Services. (European Union Blockchain Observatory & Forum).

[3] Lyons T, Courcelas L, Timsit K (2019) Blockchain and Digital Identity. (European Union Blockchain Observatory & Forum).

[4] The Illinois Blockchain Initiative (2017) *Illinois Partners with Evernym to Launch Birth Registration Pilot*. Available at https://illinoisblockchain.tech/illinois-partners-with-evernym-to-launch-birth-registration-pilot-f2668664f67c

[5] De Vaulx F, Hager T (2017) *Blockchain Primer: Enabling Blockchain Innovation in the U.S. Federal Government*. (ACT-IAC Emerging Technology Community of Interest, Blockchain Working Group). Available at https://www.actiac.org/act-iac-white-paper-enabling-blockchain-innovation-us-federal-government

[6] De Vaulx F, Hager T (2018) *Blockchain Playbook for Federal U.S. Government*. (ACT-IAC Emerging Technology Community of Interest, Blockchain Working Group). Available at https://www.actiac.org/act-iac-white-paper-blockchain-playbook-us-federal-government

[7] Durant E , Trachy A (2017) *Digital Diploma debuts at MIT*. (MIT News). Available at http://news.mit.edu/2017/mit-debuts-secure-digital-diploma-using-bitcoin-blockchain-technology-1017

[8] e-Estonia (accessed 02/23/19) *FAQ Estonian Blockchain Technology*. Available at https://e-estonia.com/wp-content/uploads/faq-a4-v03-blockchain-1-1.pdf

[9] Offerman A (2018) *Swiss City of Zug issues Ethereum blockchain-based eIDs*. (Joinup). Available at https://joinup.ec.europa.eu/collection/egovernment/document/swiss-city-zug-issues-ethereum-blockchain-based-eids

[10] VON (accessed 03/17/19) *Verifiable Organizations Network*. Available at https://vonx.io

[11] Yaga D, Mell P, Roby N, Scarfone K (2018) Blockchain Technology Overview. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Interagency or Internal Report (IR) 8202. https://doi.org/10.6028/NIST.IR.8202

[12] Reed D, Sporny M, Longley D, Allen C, Grant R, Sabadello M (2019) Decentralized Identifiers (DIDs) v0.12 – Data Model and Syntaxes for Decentralized Identifiers. (W3C Credentials Community Group)

[13] Sporny M, Kellogg G, Lanthaler M (2014) JSON-LD 1.0 - A JSON-based serialization for

linked data. (W3C)

[14] Hughes A, Sporny M, Reed D (2019) A Primer for Decentralized Identifiers - An introduction to self-administered identifiers for curious people. (W3C Credentials Community Group)

[15] Sabadello M (2017) *A universal resolver for self-sovereign identifiers*. (Medium - Decentralized Identity Foundation). Available at https://medium.com/decentralized-identity/a-universal-resolver-for-self-sovereign-identifiers-48e6b4a5cc3c

[16] Sporny M, Longley D, Chadwick D (2019) Verifiable credentials data model 1.0 – Expressing verifiable information on the Web. (W3C)

[17] IMS Global (2018) Open Badges v2.0.

[18] Decentralized Identity Foundation (accessed 03/17/19) *DIF Identity Hubs*. Available at https://github.com/decentralized-identity/identity-hub/blob/master/explainer.md

[19] Kuhn DR, Hu VC, Polk WT, Chang S-J (2001) Introduction to Public Key Technology and the Federal PKI Infrastructure. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-32. https://doi.org/10.6028/NIST.SP.800-32

[20] Barker E, Smid M, Branstad D, Chokhani S (2013) A Framework for Designing Cryptographic Key Management Systems. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-130. https://doi.org/10.6028/NIST.SP.800-130

[21] Sovrin (accessed 02/22/19) *Sovrin*. Available at https://sovrin.org

[22] Hyperledger (accessed 02/22/19) *Hyperledger Indy*. Available at https://www.hyperledger.org/projects/hyperledger-indy

[23] Decentralized Identity Foundation (accessed 05/13/19) *Sidetree Protocol Specification*. Available at https://github.com/decentralized-identity/sidetree/blob/master/docs/protocol.md

[24] Decentralized Identity Foundation (accessed 05/13/19) *ION*. Available at https://github.com/decentralized-identity/ion

[25] Decentralized Identity Foundation (accessed 05/20/19) *Element*. Available at https://github.com/decentralized-identity/element

[26] Hyperledger (accessed 05/29/19) *Hyperledger Aries Proposal*. Available at https://wiki.hyperledger.org/display/HYP/Hyperledger+Aries+Proposal

[27] Terbu O (2019) *The Self-sovereign Identity Stack*. (Medium - Decentralized Identity Foundation). Available at https://medium.com/decentralized-identity/the-self-sovereign-

identity-stack-8a2cc95f2d45

[28] Mell P, Dray J, Shook J (2019) Smart Contract Federated Identity Management without Third Party Authentication Services. *Open Identity Summit 2019*. https://ws680.nist.gov/publication/get_pdf.cfm?pub_id=925957

[29] Angieri S, García-Martínez A, Liu B, Yan Z, Wang C, Bagnulo M (2018) An experiment in distributed Internet address management using blockchains. *arXiv preprint* arXiv:1807.10528.

[30] Garman C, Green M, Miers I (2014) Decentralized Anonymous Credentials. *Network and Distributed System Security Symposium 2014*. https://www.ndss-symposium.org/wp-content/uploads/2017/09/07_3_1.pdf

[31] Sonnino A, Al-Bassam M, Bano S, Meiklejohn S, Danezis G (2018) Coconut: Threshold issuance selective disclosure credentials with applications to distributed ledgers. *arXiv preprint*. https://arxiv.org/abs/1802.07344

[32] Kubilay MY, Kiraz MS, Mantar HA (2018) CertLedger: A New PKI Model with Certificate Transparency Based on Blockchain. *arXiv preprint*. https://arxiv.org/abs/1806.03914

[33] Goodell G , Aste T (2019) A Decentralised Digital Identity Architecture. *SSRN Electronic Journal*. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3342238

[34] Wuille P (2012) *BIP32: Hierarchical Deterministic Wallets*. Available at https://github.com/bitcoin/bips/blob/master/bip-0032.mediawiki

[35] Augot D, Chabanne H, Clémot O, George W (2017) Transforming face-to-face identity proofing into anonymous digital identity using the bitcoin blockchain. *2017 15th Annual Conference on Privacy, Security and Trust*, (IEEE), pp 25-2509.

[36] IPFS (accessed 02/21/2019) *IPFS*. Available at https://ipfs.io

[37] Uport (accessed 02/20/19) *Uport*. Available at https://uport.me

[38] Zyskind G, Nathan O, Pentland A (2015) Decentralizing privacy: Using blockchain to protect personal data. *2015 IEEE Security and Privacy Workshops*, (IEEE), pp 180-184.

[39] Bertram S, Georg C-P (2018) A privacy-preserving system for data ownership using blockchain and distributed databases. *arXiv preprint* https://arxiv.org/abs/1810.11655

[40] Kokoris-Kogias E, Alp EC, Siby SD, Gailly N, Gasser L, Jovanovic P, Syta E, Ford B (2018) Calypso: Auditable sharing of private data over blockchains. *Cryptology ePrint Archive*. https://eprint.iacr.org/2018/209/20180806:124914

[41] Patsonakis C, Samari K, Kiayias A, Roussopoulos M (2019) On the Practicality of Smart Contract PKI. *arXiv preprint* https://arxiv.org/abs/1902.00878

[42] 0xcert (accessed 02/22/19) *0xcert*. Available at https://0xcert.org

[43] Centrifuge (accessed 05/23/19) *Centrifuge*. Available at https://centrifuge.io

[44] Blockstack (accessed 02/22/19) *Blockstack*. Available at https://blockstack.org

[45] SmartIdentity (accessed 05/20/19) *Smart Identity*. Available at https://github.com/SmartIdentity/smartId-contracts

[46] Al-Bassam M (2017) SCPKI: a smart contract-based PKI and identity system. *Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts*, (ACM), pp 35-40.

[47] Azouvi S, Al-Bassam M, Meiklejohn S (2017) Who am I? Secure identity registration on distributed ledgers. *Data Privacy Management, Cryptocurrencies and Blockchain Technology,* (Springer), pp 373-389.

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

[49] TPL (accessed 04/23/19) *Transaction Permission Layer Protocol*. Available at https://tplprotocol.org

[50] Province of British Columbia (accessed 05/20/19) *TheOrgBook*. Available at https://github.com/bcgov/TheOrgBook

[51] Schaub A, Bazin R, Hasan O, Brunie L (2016) A trustless privacy-preserving reputation system. *IFIP International Conference on ICT Systems Security and Privacy Protection*, (Springer), pp 398-411.

[52] Dennis R, Owen G (2015) Rep on the block: A next generation reputation system based on the blockchain. *2015 10th International Conference for Internet Technology and Secured Transactions*, (IEEE), pp 131-138.

[53] Yasin A, Liu L (2016) An online identity and smart contract management system. *2016 IEEE 40th Annual Computer Software and Applications Conference*, (IEEE), pp 192-198.

[54] Hardjono T, Pentland A (2019) Verifiable anonymous identities and access control in permissioned blockchains. *arXiv preprint* https://arxiv.org/abs/1903.04584

[55] Bloom (accessed 02/22/19) *Bloom*. Available at https://bloom.co

[56] Marx S (2019) *Upgradeability Is a Bug*. (Medium - Consensys Diligence). Available at https://medium.com/consensys-diligence/upgradeability-is-a-bug-dba0203152ce

[57] Dunphy P, Petitcolas FA (2018) A first look at identity management schemes on the blockchain. *IEEE Security & Privacy* 16(4):20-29.

[58] Goldwasser S, Micali S, Rackoff C (1989) The knowledge complexity of interactive proof

systems. *SIAM Journal on Computing* 18(1):186-208.

[59] Del Pino R, Lyubashevsky V, Neven G, Seiler G (2017) Practical quantum-safe voting from lattices. *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, (ACM), pp 1565-1581.

[60] Sasson EB, Chiesa A, Garman C, Green M, Miers I, Tromer E, Virza M (2014) Zerocash: Decentralized anonymous payments from bitcoin. *2014 IEEE Symposium on Security and Privacy*, (IEEE), pp 459-474.

[61] ZKProofs (accessed 05/13/19) *Zero-Knowledge Proofs*. Available at https://zkp.science

[62] Augot D, Chabanne H, Chenevier T, George W, Lambert L (2017) A user-centric system for verified identities on the bitcoin blockchain. *Data Privacy Management, Cryptocurrencies and Blockchain Technology,* (Springer), pp 390-407.

[63] Katz J, Ostrovsky R, Rabin MO (2004) Identity-based zero-knowledge. *International Conference on Security in Communication Networks*, (Springer), pp 180-192.

[64] KZen (accessed 06/07/19) *ZenGo*. Available at https://zengo.com

[65] Othman A, Callahan J (2018) The Horcrux protocol: a method for decentralized biometric-based self-sovereign identity. *2018 International Joint Conference on Neural Networks*, (IEEE), pp 1-7.

[66] IEEE (2017) *IEEE 2410-2017 - IEEE Standard for Biometric Open Protocol* (IEEE Standards Association). https://standards.ieee.org/standard/2410-2017.html

[67] Luxoft (accessed 05/29/19) *Cordentity*. Available at https://github.com/Luxoft/cordentity

[68] Douglas S (2018) *SecureKey Technologies to explore interoperability between Verified.Me and Hyperledger Indy*. (SecureKey) Available at https://securekey.com/press-releases/hyperledger-indy

[69] Jones M, Bradley J, Sakimura N (2015) RFC 7519: Json Web Token (JWT) (Internet Engineering Task Force (IETF)).

[70] Cheikes BA, Waltermire D, Scarfone K (2011) Common Platform Enumeration: Naming Specification Version 2.3. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Interagency or Internal Report (IR) 7695. https://doi.org/10.6028/NIST.IR.7695

Appendix A—Acronyms

Selected acronyms and abbreviations used in this paper are defined below.

ACT-IAC	American Council for Technology and Industry Advisory Council
AML	Anti-Money Laundering
API	Application Programming Interface
BIP	Bitcoin Improvement Proposal
DAO	Decentralized Autonomous Organization
DID	Decentralized Identifier
DIF	Decentralized Identity Foundation
DPKI	Decentralized Public Key Infrastructure
DLT	Distributed Ledger Technology
DNS	Domain Name System
ERC	Ethereum Request for Comments
ETH	Ethereum
FIM	Federated Identity Management
HD	Hierarchical Deterministic
HTTP	Hyper-Text Transfer Protocol
ICO	Initial Coin Offering
IDMS	Identity Management System
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IP	Internet Protocol
IPFS	Inter-Planetary File System
ISO	International Organization for Standardization
ITL	Information Technology Laboratory
JSON	JavaScript Object Notation
JSON-LD	JavaScript Object Notation for Linked Data
JWT	JSON Web Token
KYC	Know Your Customer
NFT	Non-Fungible Token
NIST	National Institute of Standards and Technology

NIST-IR	National Institute of Standards and Technology Internal Report
NIST SP	National Institute of Standards and Technology Special Publication
PII	Personally-Identifiable Information
QR	Quick Response
RBFT	Redundant Byzantine Fault Tolerance
RFC	Request For Comments
SAML	Security Assertion Markup Language
SDK	Software Development Kit
SSO	Single Sign-On
SSI	Self-Sovereign Identity
TLS	Transport Layer Security
UI	User Interface
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
W3C	World Wide Web Consortium
XDI	eXtensible Data Interchange
ZK	Zero-Knowledge
ZKP	Zero-Knowledge Protocol

Appendix B—Glossary

Airdrop

A distribution of digital tokens to a list of blockchain addresses.

Asymmetric-Key Cryptography

A cryptographic system where users have a private key that is kept secret and used to generate a public key (which is freely provided to others). Users can digitally sign data with their private key and the resulting signature can be verified by anyone using the corresponding public key. Also known as Public-key cryptography. [11]

Authentication

Verifying the identity of a user, process, or device, often as a prerequisite to allowing access to resources in an information system.

Consensus Model

A process to achieve agreement within a distributed system on the valid state.

Also known as a consensus algorithm, consensus mechanism, consensus method. [11]

Curation Market

A token-based organization model that aims at incentivizing and coordinating market participants around the curation of some information. *Term introduced by Simon de la Rouviere.*

Cryptocurrency

A digital asset/credit/unit within the system, which is cryptographically sent from one blockchain network user to another. In the case of cryptocurrency creation (such as the reward for mining), the publishing node includes a transaction sending the newly created cryptocurrency to one or more blockchain network users.

These assets are transferred from one user to another by using digital signatures with asymmetric-key pairs. [11]

Cryptographic Hash Function

A function that maps a bit string of arbitrary length to a fixed-length bit string. Approved hash functions satisfy the following properties:

1. (*Preimage resistant*) It is computationally infeasible to compute the correct input value given some output value (the hash function is "one way").
2. (*Second preimage resistant*) One cannot find an input that hashes to a specific output.
3. (*Collision resistant*) It is computationally infeasible to find any two distinct inputs that map to the same output. [11]

Decentralized Application

An application with self-enforceable backend code running on a decentralized ledger rather than a centralized server (it can rely on a set of smart contracts). Also known as "dapp".

Decentralized Autonomous

A system that is not controlled by a single entity or leader, and that,

Organization	instead, uses on-chain registries and logic to establish some form of self-sustainable organizational structure (e.g., through market incentives, network effects, and protocol designs).
Factory Smart Contract	A smart contract that creates, and sometimes, manages other smart contracts.
Hash	The output of a hash function (e.g., hash(data) = digest). Also known as a message digest, digest, hash digest, or hash value. [11]
JSON Web Token	A JSON Web Token (JWT) is a data exchanged format comprised of a header, a payload, and a signature where the header and the payload take the form of JSON objects. They are encoded and concatenated with the aggregate being signed to generate a signature. The standard was introduced by RFC 7519 from the IETF [69].
Linked Data	A method for interconnecting data structures to promote interpretability. *Term introduced by Tim Berners-Lee.*
Merkle Airdrop	A scheme to distribute the entitlement to redeem a digital token to a list of blockchain addresses in a single transaction rather than distributing the tokens themselves in a batch of transactions as in a standard airdrop. The list must be available to the participants so that they can build the proof needed to redeem the token (called Merkle proof, as it relies on a Merkle tree).
Merkle Tree	A data structure where the data is hashed and combined until there is a singular root hash that represents the entire structure. [11]
Mintable	Refers to the ability of a digital token to be created.
Node	An individual system within the blockchain network. [11]
Non-Fungible	Refers to something that is not replaceable or interchangeable.
Off-Chain	Refers to data that is stored, or a process that is implemented and executed, outside of any blockchain system.
On-Chain	Refers to data that is stored, or a process that is implemented and executed, within a blockchain system.
Token	A representation of a particular asset that relies on a blockchain.
Unlinkability	The extent to which a relying party is unable to link a given identifier to other ones a subject may own.
Uniform Resource Identifier	A compact sequence of characters that identifies an abstract or physical resource available on the Internet. [70]

1531