applied sciences

MDPI

Article

DNS-IdM: A Blockchain Identity Management System to Secure Personal Data Sharing in a Network

Jamila Alsayed Kassem †, **Sarwar Sayeed** †, **Hector Marco-Gisbert** *,†, **Zeeshan Pervez** † and **Keshav Dahal** †

School of Computing, Engineering and Physical Sciences, University of the West of Scotland, High Street, Paisley PA1 2BE, UK

* Correspondence: hector.marco@uws.ac.uk; Tel.: +44-141-849-4418

† Current address: University of the West of Scotland, High Street, Paisley PA1 2BE, UK.

Received: 6 June 2019; Accepted: 17 July 2019; Published: 24 July 2019

Abstract: Identity management (IdM) is a method used to determine user identities. The centralized aspect of IdM introduces a serious concern with the growing value of personal information, as well as with the General Data Protection Regulation (GDPR). The problem with currently-deployed systems and their dominating approach, with identity providers (IdP) and single-point services, is that a third party is in charge of maintaining and controlling the personal data. The main challenge to manage data securely lies in trusting humans and institutes who are responsible for controlling the entire activity. Identities are not owned by the rightful owners or the user him/herself, but by the mentioned providers. With the rise of blockchain technology, self-sovereign identities are in place utilizing decentralization; unfortunately, the flaws still exist. In this research, we propose DNS-IdM, a smart contract-based identity management system that enables users to maintain their identities associated with certain attributes, accomplishing the self-sovereign concept. DNS-IdM has promising outcomes in terms of security and privacy. Due to the decentralized nature, DNS-IdM is able to avoid not only the conventional security threats, but also the limitations of the current decentralized identity management systems.

Keywords: blockchain; identity management; self-sovereign; uPort; Sovrin; ShoCard

1. Introduction

Digital identity is a digital establishment of the particulars obtainable about an entity, and the identity management system ensures that only the valid users are authorized to gain access to that information. Due to the growing usage, the ability to authenticate digital identity has become more significant for the development of technologies, services, and standards [1]. In the era of social engineering, phishing, and spoofing, the identity term gets more complex, and the answer to the question “Who are you?” takes a whole other turn [2].

Identity management is a method of recognizing, validating, and authorizing participants to grant access to confidential data. In a number of schemas, service providers, called the relying party (RP), delegate the responsibility of issuing identities, manage credential to an identity provider (IdP), and rely on it to authenticate the access to certain web services [3]. This leverages the user's experience by offering a single identifier valid through different web services. Moreover, some centralized schemas have a single authoritative IdP that is used by several organizations [4]. The RP establishes, beforehand, a relationship with the IdP, and an example of that being Microsoft passport [5]. The main issue arises when the centralized identity management system deprives users of the rightful ownership of their identity data online. Subsequently, with the lack of control, a third party can abuse the trust granted by users, which raises some privacy and security flags. Similar to other centralized nature systems, the condensed single-point failure becomes a honeypot for malicious attacks.

On the other hand, decentralized approaches leverage several IdPs. Those systems agree on a shared protocol between IdPs and RPs that helps dictate the exchange of identities and authorities, like OpenID [6], Security Assertion Markup Language (SAML) [7], and Microsoft's CardSpace [8]. The thing that led to the concept of social sign-on (0Auth) was rapidly adopted by Google Friend Connect, Twitter, and Facebook. Such services increase the simplicity of access and personal information on social websites, but degrade security and privacy. Single sign-on (SSO) solutions enable HTTP and JavaScript redirection in order to exchange credentials with different layers of communication; however, it is vulnerable to attacks such as malicious identity provider attack, malicious Relying Parties (RP) / Service Providers (SP), and the replay attack.

Systems need to improve the level of security, information control flow, simplify authentication, and assertion of credential processes. The decentralized approach is a crucial turning point in identity management to address some issues, such as multiple password sign-ins, efficient management of identities, and delegating the authentication layer.

The blockchain is a decentralized, distributed, immutable database infrastructure [9], which revolutionized the idea of cryptocurrency. The wide penetration achievement showed interesting results, but the same technology can be deployed by improving different applications such as healthcare [10], IoT security [11,12], and asset management [13,14]. The blockchain can be deployed by reinventing new applications that have never been implemented before [15]. An example of that is the self-sovereign identity management systems. Self-sovereign identity is to accomplish full ownership of identity and tackle those security flags. The need for self-sovereign identity was strengthened by the EU GDPR (2018) [16], which has stressed the user's rights of increased transparency with regulations, privacy by design systems, data portability, and security.

With the growing need for security and privacy, the blockchain offers a perfect solution for delivering secure identity services [17]. The distributed ledger technology has given a new perspective on identity management systems, and new approaches transpire, aiming to enhance decentralization, user control, and transparency [18]. The blockchain provides a way out of the security and privacy holes that have been there since the dawn of the Internet, away from a central authority [19].

In the current age of digitization services like healthcare, social, and financial service, to name a few, all rely on user identities [20,21]. The management of user identities has grown far beyond validating a unique combination of user name and password. Digital services rely on identity attributes, which define an individual user in the real world, for instance education, profession, social status, and medical history, among others. This reliance has opened up further challenges of identity attribute verification as various distinct benefactors bestow these attributes: education by the education department, profession by professional bodies, and healthcare by health and social care services, etc.

The proposed DNS-identity management (IdM) offers a domain name system-like experience for identity management. Through DNS-IdM, an individual can make use of real-world attributes of his/her identify to create a digital identity that can be seamlessly utilized by service providers to validate the user and offer their services based on verified attributes. The use of the blockchain as a secure and trusted network to provision identities (with associated attributes) and exploitation of smart contracts to securely acquire identity attributes from respective benefactors sets apart DNS-IdM, not only from conventional identity management systems, but also from the latest developments in blockchain-based identity services.

Conventional identity management systems mainly rely on third parties to verify the claims about the identity attributes, which once verified, remain valid even if revoked or updated by the benefactors; whereas, blockchain-based identity services either solely rely on users to define their identity attributes or enable users to control the disclosure of the attributes based on their acceptability. DNS-IdM enables users to manage their identities, whereas for service providers, it enables them to make use of existing identity attributes and, if the required attributes do not exist, directly request them from benefactors through smart contracts. The entire process of identity management (i.e., attribute persistence, request, and verification) is seamlessly executed by the combination of a permissioned and permissionless blockchain coupled with smart contracts. In the intrinsic properties of the blockchain, tamper-resistant, permanence, and traceability make the DNS-IdM secure and trustworthy.

The major contributions of this paper are as follows:

- An analysis of three decentralized identity management systems revealing their weaknesses.
- A novel decentralized IdM architecture, with attribute associated validators, which overcomes the existing drawbacks and maximizes security.
- An evaluation to show that the DNS-IdM design is highly scalable, enhancing validators to be added to the network considering the new attributes introduced.
- Minimum disclosure by the RP smart contracts, which can be edited and customized to the best interest of the user before being added to the network.

The rest of this paper is organized as follows. Section 2 presents the background discussing the digital identity concept and the Ethereum-based IdM framework in depth. Section 3 analyses the three most advanced identity management systems by revealing their major limitations. Section 4 proposes the new architecture model that needs to be deployed on the Ethereum framework. Section 5 includes three experiments to demonstrate the security benefits of DNS-IdM. In Section 6, we summarize the use-case of DNS-IdM. Section 7 presents a comparison evaluation by focusing on security and scalability issues. Finally, the concluding Section 8 discusses the overall outcome of the work.

2. Background

The Internet lacked standards; thus, identifying people and organizations became explicit [22]. Websites started proposing their own accounts locally accessed by usernames and passwords. Moreover, the much-needed identity layer was not originated with the Internet; as a result, the centralized way of thinking has predominated ever since [23]. Aiming to solve this issue, architectures moved towards federated models closely behind decentralization.

In this section, we start discussing the concept of digital identity and IdM in depth. The decentralized blockchain framework is then elaborated to understand the working architecture of this technology discussed in the later sections.

2.1. Digital Identity

Digital identity is a proxy for real-life identity [24]. It equates fundamental trust that grants entities on the network the knowledge of with whom they are transacting. It may be represented as a set of claims revolving around a person, and those claims are made by individuals or organizations. This set of data can hold attributes such as name, date of birth, and address and can be used to identify a distinct user. Naturally, the digital identity can be used to enable its holders to gain personalized access to resources and services. Currently, various clients comprise several digital identities online, which they use every day, and without thinking much of the inconveniences, people sign in with different identities such as Facebook, Outlook, Twitter, Amazon, etc. Unfortunately, there exist a plethora of problems with the current executed models [25]. Digital identities are still soiled and fragmented with the usage of different service providers. Moreover, there was an approximation of 173,000 identity theft cases in the U.K. in 2016 [26]. The Sovrin foundation white paper described digital identity as the hardest and oldest problem on the Internet; additionally, mimicking offline identification is still far fetched [27]. Hence, in order to meet the practical challenges, a new approach is required.

2.2. Identity Management

Identity management has emerged as one of the most important and difficult homeland security challenges [28]. Identity management systems comprise complexity and scalability on one side, and privacy and security requirements on the other, creating more challenges, hindering usability [29]. Various approaches have been proposed to manage identities in a proper way and also to find a good trade-off between user experience and security. The concept of a single sign-on has been introduced, and open standards such as OpenID [6], 0Auth2 [4], and SAML [7] are being used. OpenID has been used by Facebook for a while before they moved to Facebook Connect [30]. Nevertheless, IdM is quite difficult to handle when it comes to security challenges [28], and in order to meet the challenges, several schemas and models resulted in a balance between complexity and security.

Figure 1 presents a digital identity life cycle of the online identity framework. In an optimistic scenario, online transactions would mimic face-to-face ones by having clients presenting identity documents such as a passport or driving license. In those cases, trust is delegated to official intuitions that give out those documents. In this architecture, users do not own their identities; instead, they rely on services called identity providers [31]. Those services are often centralized and can be revoked, accessed, or lost by the third party.

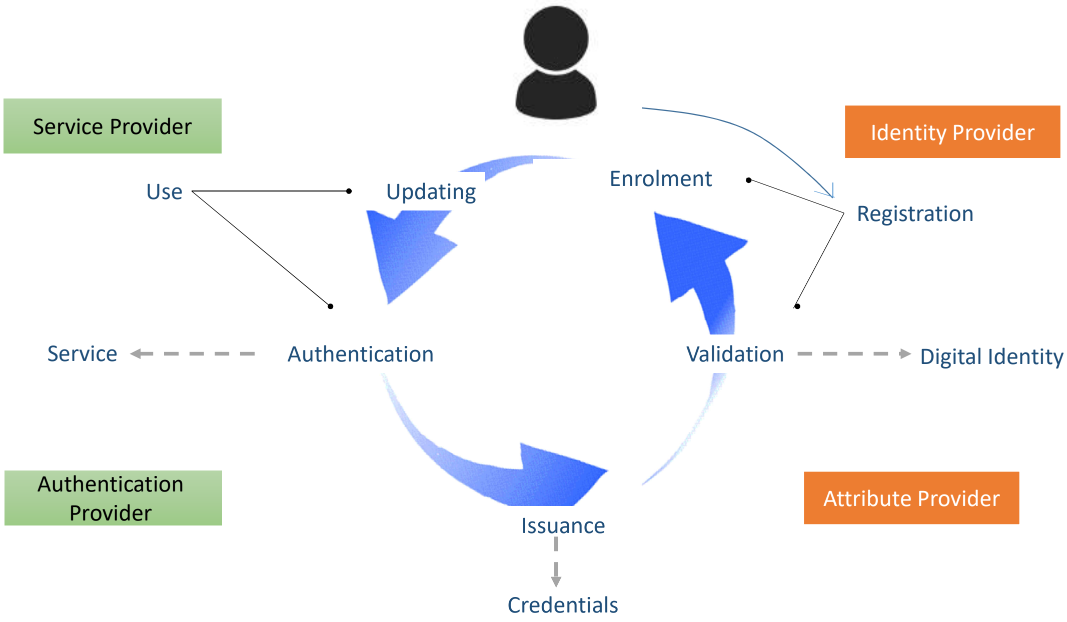

Figure 1. Digital identity life cycle and key roles.

2.3. The Decentralised Framework

The blockchain is a decentralized distributed ledger technology, which enables developers to deploy decentralized applications (DApp) [32]. The Ethereum blockchain was introduced to end-developers as a generalized technology standing on similar concepts as Bitcoin. Ethereum has dedicated smart contact-oriented programming languages that help to build DApps for different purposes. According to the Ethereum yellow paper [33], Ethereum aims to provide a tight, integrated end-to-end system for implementing software on a yet-unexplored compute paradigm. In 2013, Ethereum was introduced as the "Next generation of smart contracts and the DApp platform", available as an open source. The main goal of Ethereum was to put together and contribute to the concepts of on-chain meta-protocols, scripting, and altcoins [34]. Ethereum can benefit from accessing an inexpensive platform that offers a set of security features such as authentication, resistance from Distributed Denial of Service (DDoS), server-less infrastructure, and decentralized infrastructure.

Though it was revealed that the blockchain network is vulnerable to various attacks [35], it is a secure technology compared to the centralized system with centralized attributes.

Ethereum has the concept of state, which is made out of "accounts" objects. The state contains the accounts within the network, and they consist of two kinds, external and contracts. Accounts are addressed by 20-byte addresses, and the "accounts" state is transitioned, as well as changed with the value and information being exchanged. External accounts are effectively used to exchange Ether and start contracts, whereas contracts are instances of the contract code deployed to the blockchain. Different instances of contracts are indexed with different addresses. As a type of account, contracts can hold a balance, which means that a contract can be payable and also can exchange Ether.

2.4. Scalability

A block in the blockchain consists of a digital piece of information. The decentralized nature of this technology has brought huge demand in the technology field. However, the vast popularity may cause a slow down of the network. The Ethereum blockchain network is able to handle only 15 transactions per second [36]. The volume of transactions has made Ethereum network congested, raising the price of the gas [37].

The gas is viewed as the money side of the application. It is a way of measuring cost in terms of computing, not at the level of money. Gas varies concerning the operations that run on nodes, and those operations can send a transaction or execute codes, such as adding and looping of smart contracts. Operations that are done by miners on the Ethereum blockchain incur a computing cost on each one of them. In return, they get rewarded for it with Ether coins. All validators also incur the cost, but only successful miners get rewarded. While executing the contract's functions, the gas consumption varies as well. Gas, costs per operations, is allocated by Ethereum, and it is not up to every single computer to put down how much computing it performs. Overcoming the scalability issue will help developers handle an increased number of transactions and bring down the load of the principal chain by having the bulk transaction moved to another layer.

3. Identity Management Challenges

In this section, we focus on three identity management systems that are imposed on the blockchain network. Table 1 presents a comparison of the three identity management systems. The state of current identity systems is problematic, as mentioned in the uPort paper [38]. Moreover, digital identity systems that are widely deployed are fragmented, mostly deliver poor user experience with the repetitive logins, and are soiled by multiple service providers [38]. The centralization architecture also results in valuable data being compromised by attackers [39].

Table 1. Identity management system's comparison table. IdM, identity management.

IdM Systems	Control		Security		Privacy				
	Policy Management	Explicit Consent	Basic Security	Multi-Lateral Security	Anonymity Support	User Empowering	Data Minimization	Remote Admin	Privacy Standard
uPort	✓	✓	✓	✗	✓	✓	✓	✓	✗
Sovrin	✗	✓	✓	✓	✓	✓	✓	✗	✓
ShoCard	✓	✓	✓	✓	✓	✓	✗	✗	✓

There are several designs of identity management systems deployed on the blockchain trying to accomplish self-sovereign identities such as Cambridge Namecoin [40], Blockverify [41], and Cambridge Blockchain [42]. However, this section analyses the most advanced identity management systems, uPort, Sovrin, and ShoCard, and also reveals the current challenges these models are facing.

3.1. uPort

uPort is an open source framework that provides a decentralized identity. It aims to provide a decentralized identity for specific services like emailing and banking [38]. uPort is built on top of the Ethereum blockchain [17], and makes use of the smart contracts feature. It is operated by deploying smart contracts and supported by the interaction of contracts. Contracts are uniquely addressed in the system by a 160-byte token. Figure 2 presents the general architecture of uPort, illustrating how a transaction occurs [18].

uPort does not comprise a centralized server or authenticate the identity of the owner. Parties over the Internet have no way of knowing whether the initiated claim is legitimate. uPort offers a recovery emergency option that links uPortID's as the user trustees. Trustees vote to replace the public key that associates the user to the service with the uPortID. However, even with social recovery option offered, if the ownership is compromised, trustees can be compromised as well. Trustees are publicly linked to a certain uPortID, and in the same sense, they can be compromised. uPortID spoofing might go undetected by the controller, which makes it vulnerable, and it can also be spoofed permanently. Moreover, uPort users have control over their data and offer minimal disclosure. No information is needed to bootstrap constraints, and there is also no inherited linkability between uPortIDs, which comes at the cost of complexity and user experience. Other than that, the attributes are decentralized while storing InterPlanetary File System (IPFS) [43] and can be encrypted individually by enhancing more security. However, the registry is centralized; as a result, the JSON data are still visible, and attributes' meta-data can be leaked. Thus, the relationship between identity providers and relying parties is not secure.

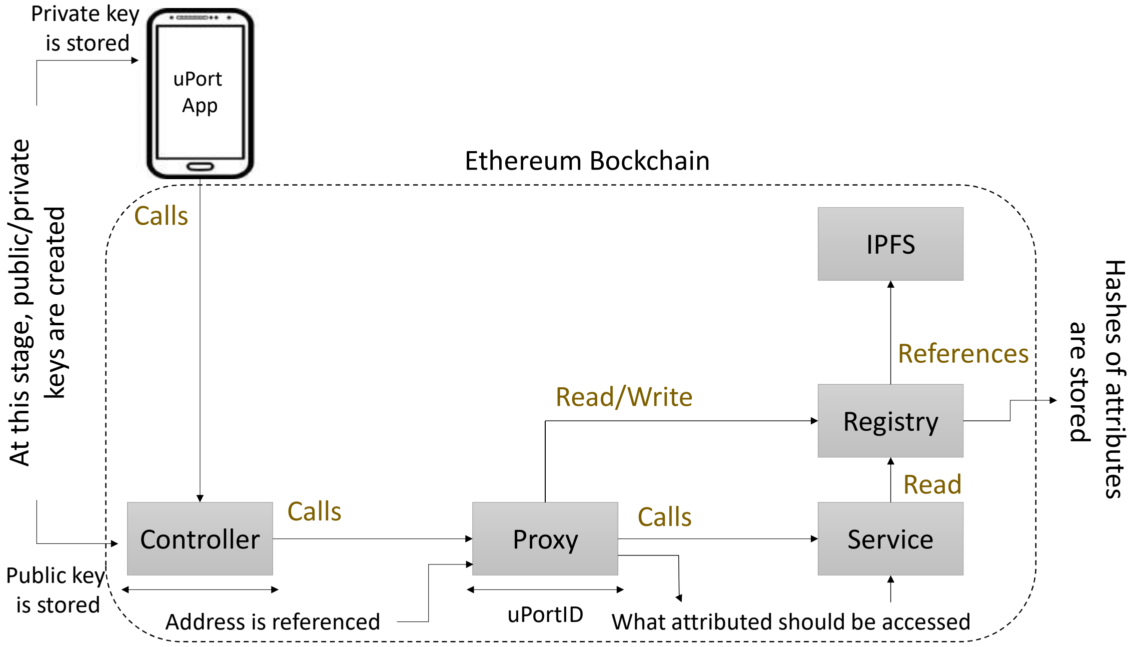

Figure 2. The general architecture of uPort. IPFS, InterPlanetary File System.

On another note, users have potential control over their data to read/write their registry, and it can have a harmful impact as users are capable of deleting negative attributes from their benefits. uPort offers discreet services, but there is no search directory to find IDs and connect. In addition to the limitations, uPort comprises two major disadvantages. Firstly, a government could control its citizen while making the digital transfer by leveraging identity, and secondly, if an attacker manages to get hold of a user's private key, the confidential information could be stolen. On the other hand, the recovery delegates of distinct uPort users are open to the blockchain, which makes the delegates a prime target of attacks, while an attacker intends to exploit users' identity [44]. Moreover, confidential data, which are accumulated in the Chasqui server temporarily, are not encrypted and can result in data being compromised.

3.2. Sovrin

Sovrin is an open-source, decentralized identity network [45]. Unlike uPort, only trusted institutions can run nodes that can take part in the consensus protection. Those privileged trustees are called stewards and can be any institutions, such as banks and universities. The Sovrin ledger is not a public ledger, rather it is a permissioned ledger and runs according to the Sovrin trust framework.

Several identities are created by Sovrin to maintain contextual separation for privacy purposes. Each identity naturally has a pair of private or public keys. The identity is managed either by the user or appointed by a "guardian service". The illustration, in Figure 3, presents the architecture of the Sovrin identity management system, which demonstrates how everything is transacted. Moreover, the Decentralized Identifier (DID) public key is a data structure that has user identifier meta-data transact with the identifier. The Sovrin ledger contains transactions associated with a specific identity. The identity is written, distributed, and replicated among permitted nodes, which are referred to as stewards. The nodes run an enhanced version of redundant Byzantine fault-tolerant protocol of Aublin [17], and the nature of the architecture makes the ledger be permissioned, where trust relies both on the people and the code.

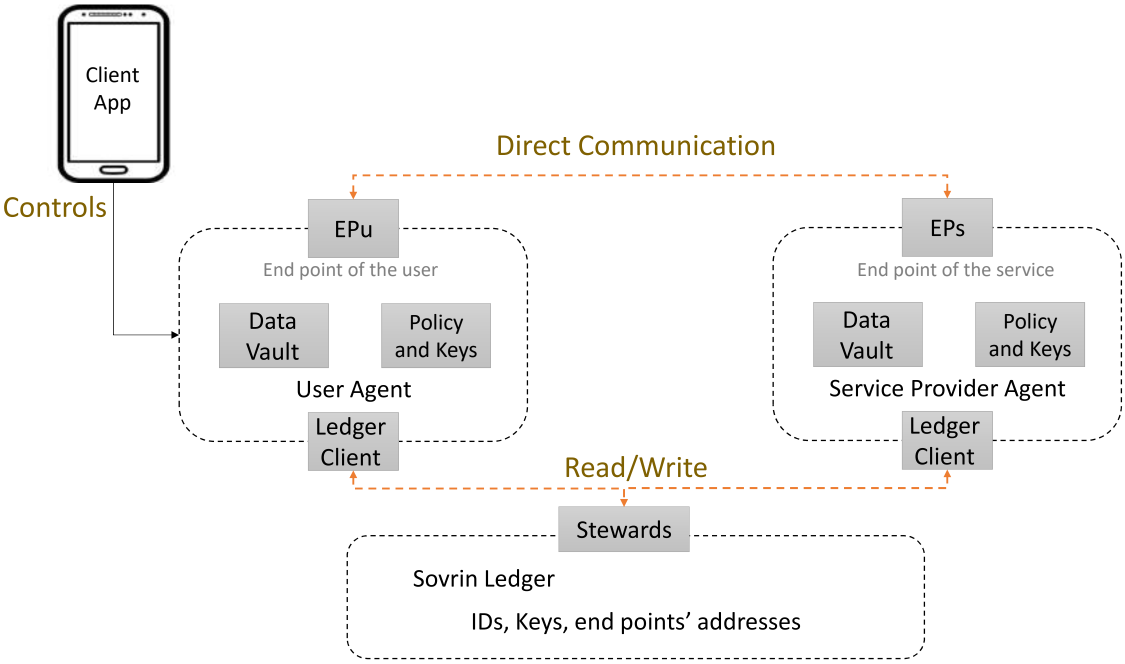

Figure 3. The architecture of the Sovrin identity management system.

Sovrin takes full control over users identity, and the users decide what attributes they want to share with relying parties. Users make use of the Sovrin ledger to identify the correct network endpoints and use either their own or the agent's phone storage. On the other hand, verifying the parties the data are shared with is a challenge that might be addressed by the web-of-trust approach. Moreover, there is no third party to share keys other than the agents. Information secrecy is protected since users can choose only to share identities. Another major limitation of Sovrin is user dependence on a determined solution. However, user experience was not considered in the design architecture, making it a complex and cumbersome design architecture.

3.3. ShoCard

ShoCard is a blockchain-based identity authentication platform that aims to provide a trusted identity by protecting users' identity [46]. It uses distributed ledger technology (DLT) to bind a user identifier, existing trusted credentials, and additional identity attributes together with cryptographic hashes to be considered for similar to face-to-face transactions. ShoCard uses Bitcoin time-stamping for signed cryptographic hashes of the user's identity information. Figure 4 shows how the transactions occur in a ShoCard network.

The ShoCard server functions as an intermediary to manage the certification exchanged between users and RPs. However, the design is more centralized than relying on the DLT, which means that if a company ceases to exist, then identities are unusable. Users are needed to upload credentials, requiring more information than needed; for instance, a scanned copy of a passport must be uploaded. Moreover, ShoCard stores encrypted data, but data can be associated with both ShoCardIDs and relying parties, which compromises privacy.

ShoCard generally supports unidirectional identifiers, although omnidirectional identifiers are needed to accomplish an ecosystem where certifications can be reused. ShoCard supports a multitude of identity providers and provides a consistent and straightforward user experience. On the other hand, it is ambiguous whether the users will be motivated to use an IdM with such limitations and willing to be educated on the implications of managing their identity on the blockchain technology.

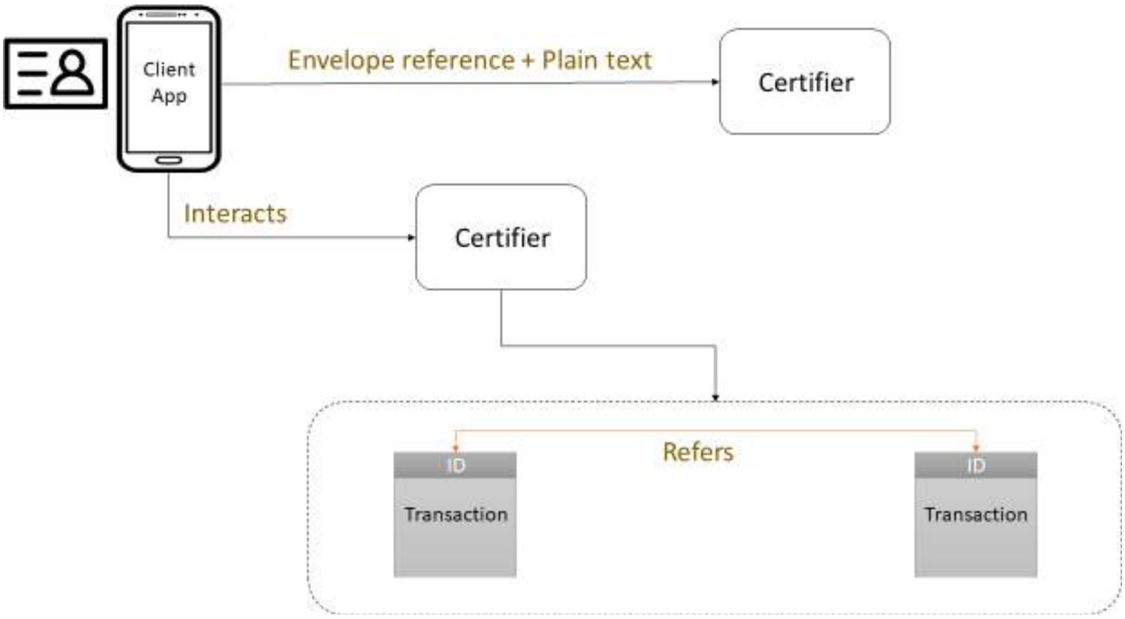

Figure 4. The architecture of ShoCard.

4. Proposed Architecture

This section analyses the novel architecture of the blockchain-based identity management system. The decentralization nature of the blockchain fades several concerns instantaneously; for instance, the linkability of data and the high level of control third parties have over data.

Figure 5 illustrates the new architecture of DNS-IdM. The novel design allows a third party to identify users, and by that, a customized service is offered, while still conserving user's ownership of the data. The aim is to accomplish decentralization with self-sovereign identity management and tackling that with the best approach on the level of security, privacy, and performance.

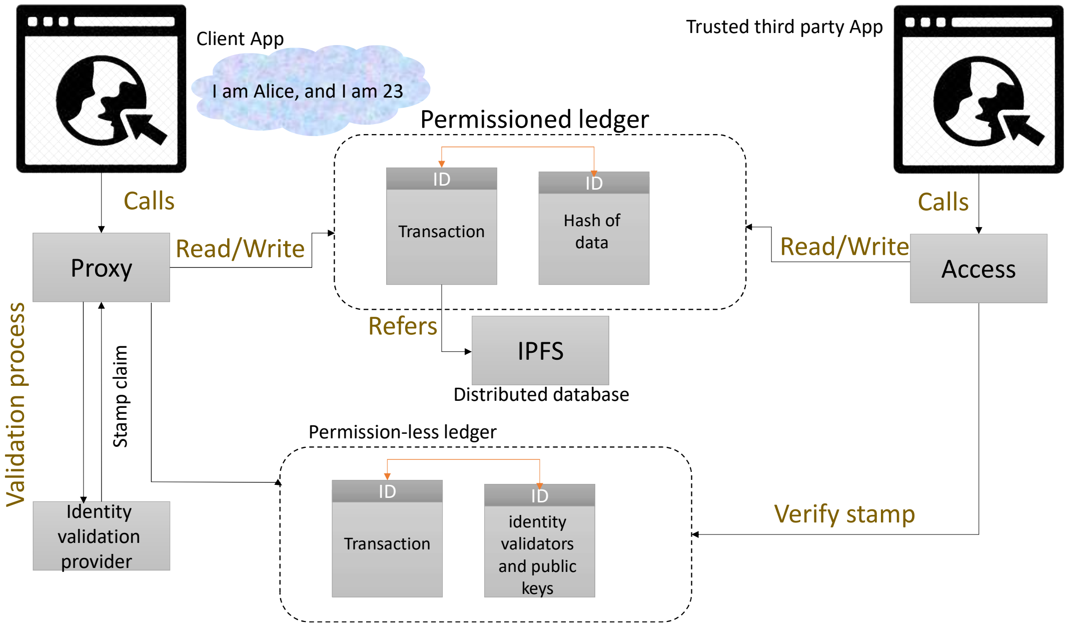

Figure 5. The new design architecture.

4.1. System'S Requirements

The information has the right to be owned by the user. Based on that, users will have the right to write and read data off the database. That can be implemented by users creating new accounts on the ledger, as shown in the registration process. The data should be stored in a way that makes it infeasible for unwanted parties to access them, implemented through a permissioned Ethereum ledger, however having in mind that the whole point of owning an identity online is to be able to use it, show it to the intended parties, and that to be done in the exchange of customized data. Other parties should also be able to read data off of the database, but with minimal disclosure. The user should allow the other party to access a particular attribute other than the complete information. That is implemented by customizing smart contracts relevant to the service provided. In addition to that, the validation process should be accompanied by any data addition done by the user. If users are given the full control of their data, adding false data or neglecting false attributes might occur. Therefore, this novel architecture considers performance and maintains easy usage for clients.

4.2. Registration

A private Ethereum network needs to be set up first, and the user's data are maintained on a ledger accessed only by the clients and particular service providers. This is allowed in the blockchain by keeping an array of registered addresses, as shown in the `Register()` function in Listing 1. The modifier is activated automatically to check whether the user should get registered to gain access to the service by contacting the registration contract.

Listing 1: Use of register function that keeps an array of registered addresses.

```
// The first contract to activate
contract Registration {
address[] clientlist;
function Register () determinepublic {
clientlist.push(msg.sender);
}
}
```

In order to make it convenient for the users, the proxy contract does the required task in the background on behalf of the users. Listing 2 shows that the modifier terminates the contract while the conditions are not met, which limits the blockchain of unwanted access.

Listing 2: Termination of contracts when conditions are not met.

```
modifier OnlyClient() {
for (uint i = 0; i < clientlist.length; i++)
msg.sender == clientlist[i];
require(msg.sender == clientlist[i], ‘‘Can’t access until registered’’);
}
```

4.3. Read/Write Values

The registered user gains access to the ledger and is able to read, as well as make necessary changes to the values of the ledger. The values are referred to as a “struct” of attributes. Each attribute consists of an attribute type, which will be used further ahead to specify which validator contract should be referred to add the “truth” stamp to the data added. Moreover, the attribute should have an actual value containing details such as “Alice”, “23”, or “Student”. Subsequently, there should exist a flag that could be set by the validator to include truthfulness. All that should be mapped to a specific address and can be associated with the users by their address.

4.4. Validation of Identity Values

Unlike uPort, which lacks attribute validation, the proposed system offers a layer of verification to ensure that the identity data maintain authentication. To address that concern, each identity attribute needs to be validated by a specialized validation provider shown in Listing 3. For example, medical record data can be introduced and validated by a hospital. Hence, a validator function may be called as a prerequisite for adding an attribute.

Listing 3: Validation of identity attribute.

```
function addattribute (string TypeInserted, string ValueInsterted) public {
bool flag = checkingstamp(TypeInserted, ValueInsterted);
require(flag = true, ‘‘Invalid’’);
identity[msg.sender].AttributeType = TypeInserted;
identity[msg.sender].Value= ValueInsterted;
}
```

4.5. Validation Contract Lookup

While each attribute type is validated by a specialized contract, it might be unclear how the proposed system knows which contract to call for validation. This is where the DNS contract comes in handy. It gives public access to the entries on a permission-less blockchain. The contract is named DNS, due to the similarity of the roles, trying to return the route of reaching a service. Listing 4 presents that a route lookup is done by the route function, which returns the address of the specific validator contract to be called. The blockchain holds the addresses of these contracts, as well as the types of the attributes they validate. Therefore, prior to an attribute being mined and added successfully, it should go through the validation process. The ledger will be accessed, and the validator contract address will be retrieved.

Listing 4: A route function executes root lookup.

```
function route (string _type) public view returns (address) {
for (uint i = 0; i <= contracts.length; i++) {
if (keccak256(contracts[i].datatype) == keccak256(_type)) {
break;
}
}
return contracts[i].contractaddress;
}
```

The other functionality offered is shown, in Listing 5, in the following setDNS function, where new validation contract entries are to be added to the database. That is accomplished by pushing the new contract's address and the attribute type validating to the array of entries. The following ensures scalability and the independence of the services nature.

Listing 5: Use of setDNS function to add new validation contract entries.

```
function setDNS (address _a, string _b ) public {
contracts.push(cont ({
datatype: _b,
contractaddress: _a
}));
}
```

Once all the above conditions are met, the user is free to add it for miners to mine it and include in the blockchain network.

4.6. Service Providers Accessing Identity Attributes

The third party builds a customized smart contract that allows access to a limited attribute based on the type of service it provides. It enables contacting the DNS contract to retrieve the data about the validator and also checks for itself that the data are authenticated by validating the signature. If the returned address matches the claimed address, then it is considered as authenticated.

5. Security Analysis

This section analyses DNS-IdM in terms of security and robustness. Our analysis involves a three-tier security experiment to demonstrate that the proposed system is a comprehensive protection against various threats. Moreover, our analysis evidences that the added features help to overcome several challenges that the existing IdM's are facing at the moment.

5.1. Authentication Protection: Preventing Phishing

In traditional systems, there is no way for the user to authenticate the IdPs and RPs. The fact that makes phishing attacks possible is based on the ability of the attackers to convince users to reveal their credentials and personal data.

Figure 6 shows a phishing attempt on the DNS-IdM, assuming that the end goal of the attacker is to gain access to the banking service customized to the user. The very first step of the attacker may involve sending a malicious email containing a link to redirect the user to a legitimate-looking application asking for personal data attributes. Any information provided by the user can be utilized by the attacker to access the ledger through the proxy, to gain service ultimately. However, access to the ledger requires permissions, and it is restricted to the client address. Moreover, the adversary also fails to validate his/her claim unless he/she has access to users official documents, such as a bank card to scan. Therefore, the attacker fails from his/her attempt.

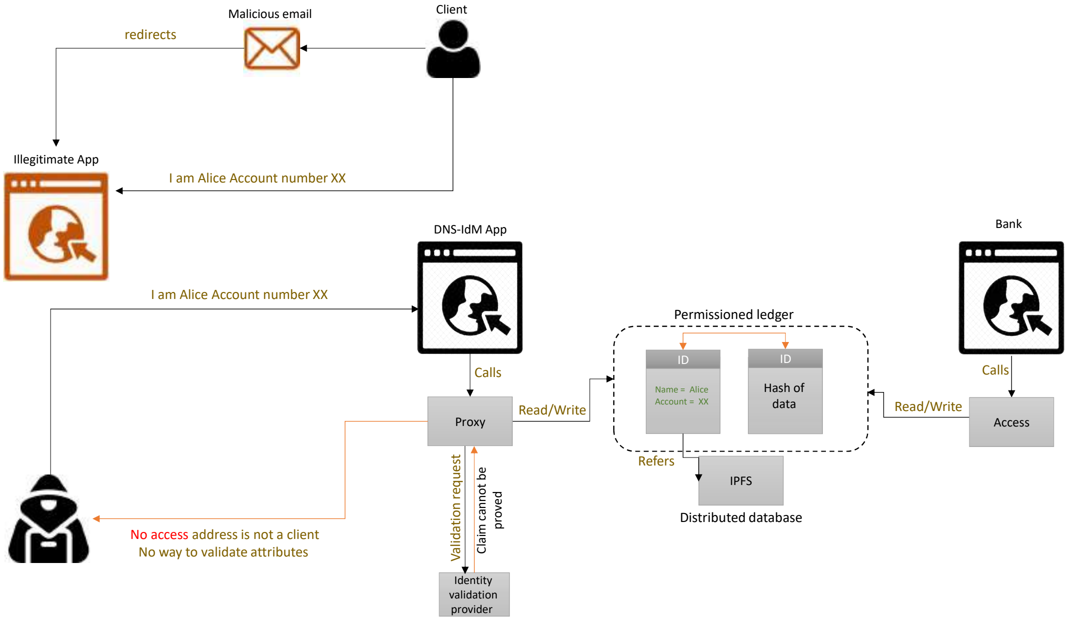

Figure 6. Attempted phishing attack on the DNS-IdM system.

The DNS-IdM is able to defend against phishing attacks as the proxy contract terminates any unwanted access from unregistered addresses. Addresses are mapped to identity attributes, and claims are validated before getting stamped by the identity. Hence, features such as restricted address-based authentication and the validation criteria secure the system against phishing attacks.

5.2. Ownership Protection

Consider a scenario where Alice holds a registered Ethereum account (nonce, balance, codeHash, storageRoot). An account is referred to by an address, the processed Keccak-256 hash of the public key, and authenticated using the private key. Figure 7 shows an experiment that aims to compromise the ownership of the identity in the DSN-IdM system. Alice's address maps to an array of identity attributes added and maintained on the permissioned ledger, preventing Eve from spoofing the ownership due to the authentication schema in place.

DNS-IdM ensures users have a protected and private ownership. The implemented Elliptic curve digital signature algorithm (Ethereum key generation) provides a secure authentication scheme. Addresses are mapped to identity attributes, and access can be gained upon meeting certain conditions. In addition to that, the identity data are maintained and stored in a completely decentralized way in a permissioned ledger. Hence, DNS-IdM protects against unauthorized access, by the ownership and privacy being preserved.

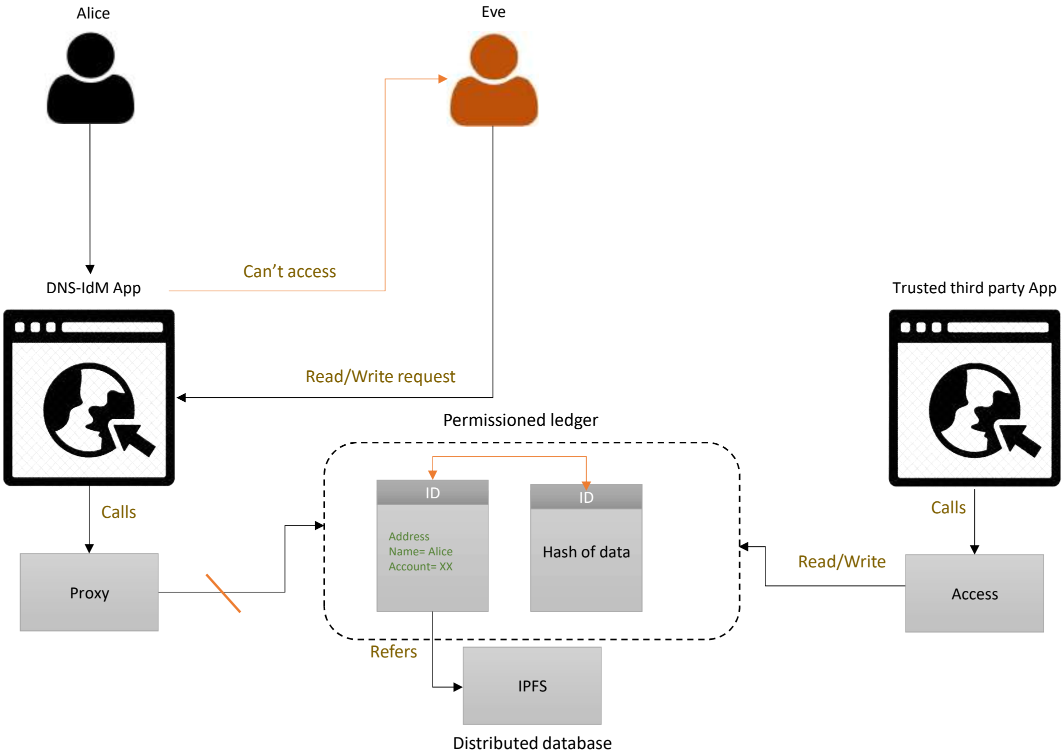

Figure 7. Providing protected and private ownership to users.

5.3. Identity Mapping Validation

Validation can be significant while dealing with identity management. An experiment in Figure 8 illustrates the importance of validation in terms of security. Assuming a scenario where Eve, a malicious user, intends to make a false claim about her identity. Eve is already a registered user, meaning she is permitted to issue a claim. Eve makes a query to the permissioned ledger claiming that she is Alice. The proxy handles the query and contacts the validator to issue a validation challenge for Eve. Eve will not be able to provide a way to validate her identity; hence, the attributes will be added as unstamped. A third party needing that specific attribute to give back a customized service will be aware that the claim is unstamped and, hence, reject it.

The identity validation provider in DNS-IdM offers validation schemas specifically associated with a particular attribute type. Moreover, DNS-similar-lookup is done to make it possible to link each attribute type with the validation contract address. Hence, the validation layer ensures that the users have reasonable control over their data, however restricting any negative values from being stamped.

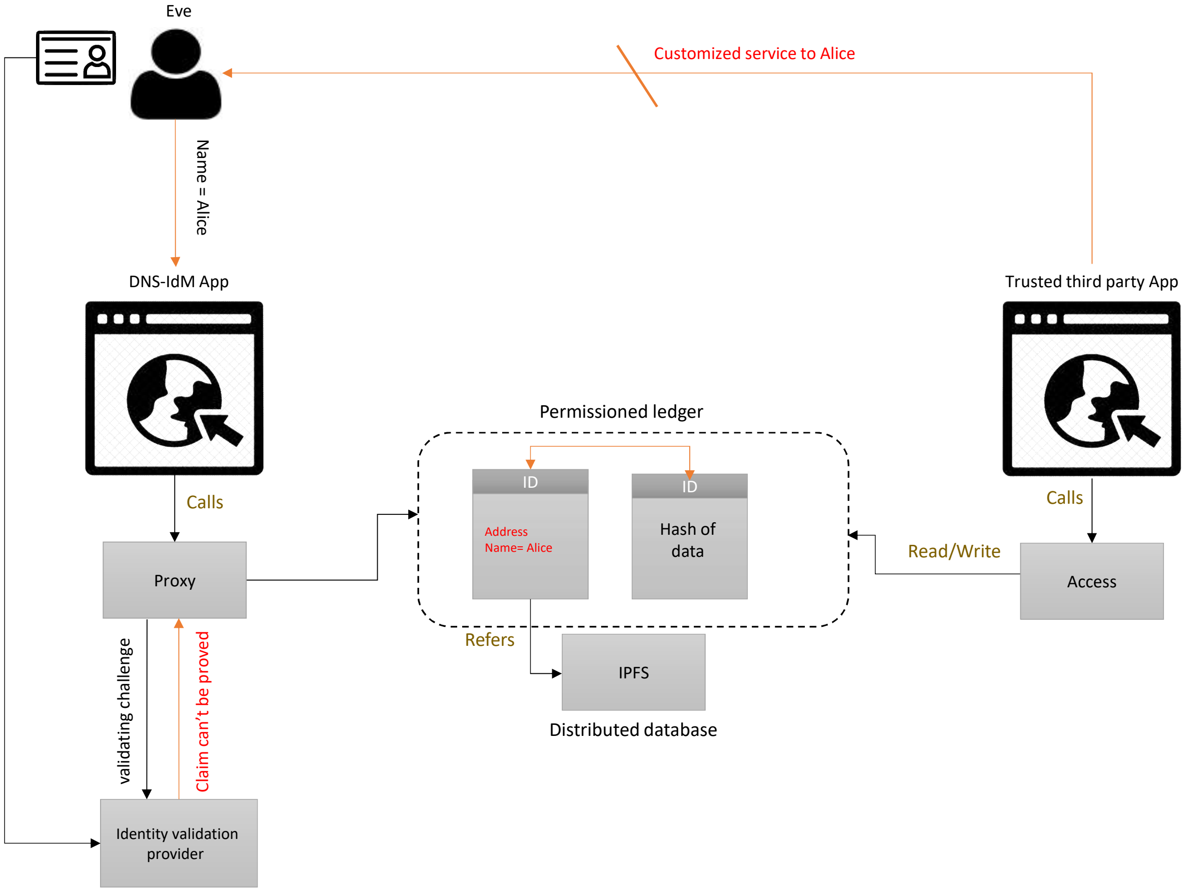

Figure 8. Enhanced security of DNS-IdM with the validation layer.

6. DNS-IdM Use Case

In this section, we elaborate on the use case of DNS-IdM architecture. We first define each of the phases of the proposed architecture and then put them in a real working scenario. The DNS-IdM design enables a third party to identify users and by that offer customized services, while still conserving users' ownership of the data. In the following, our proposed blockchain-based identity management system is composed of:

- Account registration to gain access to the permitted ledger that holds the user's identity data.
- Identity attribute scalability, where different attributes can be added to the system.
- Users adding values to their identity data database.
- Verification of the attribute value by associated verification contracts.
- A permission-less ledger that holds the attribute-specific verification contracts.
- Conditioned access of service providers to the identity data ledger.

While in a working scenario, we assume Alice is 23 and wants to prove that to a third party server. Alice does not want to reveal any other data other than that and wants to take control of her information. Alice registers to the DNS-IdM, which allows her to gain access to the permissioned ledger so that she can promote her data. She will have an account entry that can ensure her access to her data.

However, Alice wants to maintain several attributes to her identities such as name, age, and profession by having the data written to the ledger. To minimize the freedom of erasing negative attributes or adding false information, a proxy contract takes charge throughout the validation process to approve attributes. Different attributes are stamped by smart contracts that can take advantage of different methods to validate information such as documents and biometrics. Those smart contracts are maintained by different authorities depending on the type of attribute. Name and age can be validated by the "e-ID validation provider" smart contract, and the profession can be validated by a different node.

Just like DNS, a similar approach is needed to know whom to contact and how to validate the attributes. Precisely, it is also required to determine why another permission-less ledger is maintained holding different addresses and keys. Alice accesses the web application, claiming that her name is Alice and she is 23. Automatically, a proxy is called to handle the validations. The proxy accesses the permission-less ledger to determine the address of the contract. The validator processes the claim and asks for the scanning of the e-ID, a way of authenticating data. After that, the claim is stamped, which sets the flag to one, and the transaction refers to the address of the validator. Signed data are sent back to the proxy, and Alice is free to promote that on the Ethereum blockchain.

After that, the service provider needs to know just two attributes, name and age, and a customized smart contract is coded based on the nature of the service that comes with the permission to access the ledger. The service gets to read off of the ledger certain attributes and can access the DNS-alike ledger to get the address to verify the stamp.

The whole procedure ensures that privacy is maintained by keeping the permission exclusive to the user, and also, ownership is within the hands of the user. Even though the users are in full control of their data, that feature is being maintained, and the users cannot abuse this feature. Take into consideration that different attributes can be validated differently. For example, a university would find it sufficient to scan the ID to prove membership or a hospital would need biometric evidence. The whole working scenario comprises the concept of DNS, where entries can be added and updated. For example, if a service provider joins the network, it would like to customize and provide its validator contract that is feasible to accomplish, and by that, scalability is provided.

7. Evaluation

In this section, we evaluate the proposed architecture from both the security and performance point of view. The current IdM systems cannot accomplish secure, privacy-friendly, and user-friendly features spanning the service over organizations [29]. Thus, it is imperative to evaluate the new DNS-IdM architecture to determine its effectiveness. Table 2 shows a security-based comparison of DNS-IdM with the three advanced IdM systems discussed in Section 3.

Table 2. The security comparison of DNS-IdM with 3 advanced identity management systems.

Parameter	uPort	Sovrin	ShoCard	DNS-IdM
Protected Ownership	✗	✗	✗	✓
User Control and Consent	✓	✓	✓	✓
Directed Precise Identity	✗	✓	≈	✓
Human Integration	✗	✗	≈	✓
Validation Layer	✗	✗	✓	✓
Privacy Friendly	✗	✗	✗	✓

7.1. Security Analogy

The blockchain framework can maintain identities and a vast number of nodes, away from centralization; therefore, concerns such as single-point failures can easily be addressed and do not affect the overall flow of the network. The probable phishing attacks also no longer exist in DNS-IdM, where the authenticating process does not include redirecting HTTPs. However, the same attack is very much feasible against the uPort system, since the private key is generated and stored locally. The private key controls the uPortID, and once it is compromised, no user authentication steps are in place to protect. This vulnerability makes it possible for the attacker to control all personal data permanently. Moreover, uPort lacks identity attributes validation, which implies that the attacker can make false claims. Phishing attacks might not be a threat to the Sovrin system, due to the fact that authentication cannot be replayed (random one-time nonce). With the Public Key Infrastructure (PKI)-no password infrastructure, there is nothing for attackers to steal and attempt a phishing attack. Countering this, authentication is done both ways, and the service provider's signature is matched to a well-known public key.

In DNS-IdM, users can gain the ownership of their identity data and maintain them on a secure ledger, and by that, privacy concerns are addressed with the absence of a third party. It enables the control to be handed to the user and away from IdPs. Considering a similar scenario, other systems are not as robust as DNS-IdM. With the uPortID recovery schema, the user's uPortID is transparently linked to the trustees uPortID, meaning that it can be a vector of attack that targets the ownership and identity privacy. While the controller is compromised and the trustees of uPortID is replaced, the attacker simply gets full control over the user's identity. A similar concern might arise with Sovrin systems, as well. Due to users high reliance on agencies and stewards, confidential data can reach unauthorized parties, minimizing privacy and ownership. From the protected ownership aspect, ShoCard plays the role of the middle man, and that might lead to creating unreliability about whether the ShoCardID would still exist, while the company is non-existent. The users will not be able to make use of their certifications; hence, it puts the user's identity ownership at significant risk. From the privacy aspect, even though data stored in the ShoCard servers are encrypted, it may be possible to map a specific ShoCardID to a relying party.

The significance of the validation layer and how it enhances security to DNS-IdM were already discussed in Section 5. However, while comparing other systems, we revealed that uPort offers no validation requirements to the identity attributes provided by the user. Although the Sovrin system relies on the web-of-trust as a sort of validation, it lacks a cut edge validation layer. As for ShoCard, credentials are bootstrapped with the user's identity as a validation method, such as a passport or driving license, which can provide validation. On the other hand, this approach does not ensure minimal disclosure of personal data, and the ShoCard seal can include more data than intended or needed.

7.2. Performance and Scalability

Besides security, it is also important to assess the performance. If the DNS-IdM fails to perform according to user needs, then it will not be adopted by potential users. In order to assess the performance of the proposed architecture, the concept of "gas" must be evaluated. The gas is implemented when the developer is deploying a contract and wants to set a price of how much can be paid for the miner when a transaction is successfully mined into a block. The gas price is the actual bridge between gas and the money that is being effectively paid to deploy services.

While executing a transaction in the DNS-IdM, gas requires being specified by the sender informing the miners how much gas he/she is willing to give. While the code is being executed, if the gas limit is exceeded, it fails to mine in the blockchain block. Gas is a reflection of contracts' performance and helps quantify it. After we deploy the contract on the private network, the gas balance must be measured before and after invoking a function.

Since the loops do not comprise a fixed number of iterations, the proposed IdM runs as an ongoing process relying on the storage values. Due to the set limit of gas, the transaction occurring must consume a limited amount of gas without going beyond the block size. If on any occasion, the number of iterations passes beyond the limit, then it will terminate the functioning of IdM.

Figure 9 presents the gas scalability of the registration contract, DNS contract, and attribute contract. It shows the gas scalability of the registration contract after several iterations are executed. The execution resulted in a constant value. The main operation used to process this contract was sstore, which can add to permanent storage memory, and the cost is irrespective of the number of iterations. Besides that, it shows the scalability of the DNS contract with the gas spent executing increasing iterations. The looping of the code and retrieving data from storage explain the approximately linear increasing of the cost. Moreover, it also presents the scalability of the adding attribute contract with the gas spent executing increasing iterations. The code includes accessing memory, adding data, and calling another function for validation.

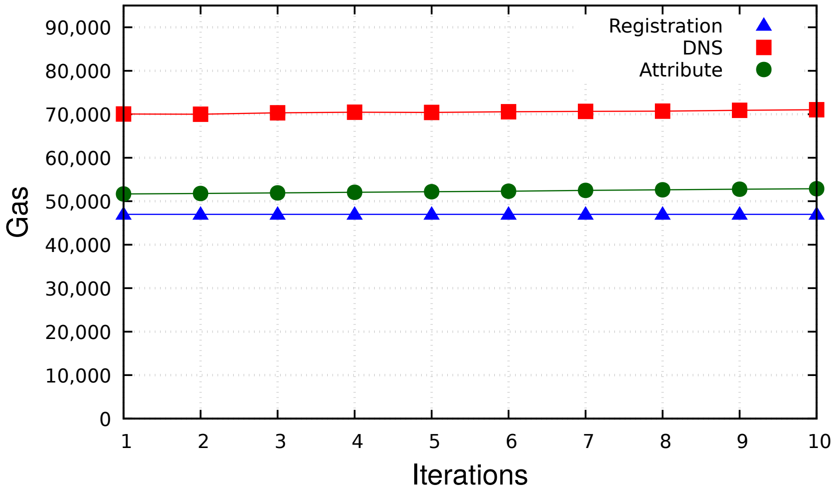

Figure 9. The scalability of the registration, DNS, and attribute functions with the increasing iterations.

8. Conclusions

Centralized identity management has long served as a secure entry point for digital services. However, as modern-day digital services tend to bank on identity attributes for user authentication and authorization, conventional identity management systems lose their efficacy and practicality in leveraging service providers to make service provisioning decisions and helping users to make validated identity claims.

In this paper, we critically analysed blockchain identity services, namely uPort, Sovrin, and ShoCard, revealing drawbacks to meet the needs of modern-day digital services. To overcome the limitations and weaknesses of identity attributes: persistence, request, and verification, in existing identity management systems, we proposed DNS-IdM. The proposed system follows a DNS-like approach that enables users and service providers to make identity attribute claims and verification using real-world identity attribute benefactors. The use of permissioned and permissionless blockchains, along with smart contracts enabled the secure and trustworthy management of identities. The experiment demonstrated the efficacy of DNS-IdM for real working applications with amicable overhead and security. The experiments illustrated that service providers can request identity attributes directly from benefactors, through smart contracts, and using blockchain transactions, users can manage and trace their identities.

For future work, we will explore the application of seamless attribute requests and service-based attribute validation. This will enable users to make use of all possible real-world identity attribute benefactors and also ensure the service provider can only request from those attributes that are needed for provisioning a specific service. In addition to this, we will also identify the facilitators and barriers for blockchain-based identity management services in developing compliance with digital standards.

Author Contributions: Writing—original draft, J.A.K., S.S., H.M.-G., Z.P. and K.D.

Funding: This research received no external funding.

Conflicts of Interest: Authors declare no conflict of interest.

References

1. Domingo, A.I.S.; Enríquez, Á.M. *Digital Identity: The Current State of Affairs*; BBVA Research: Madrid, Spain, 2018.
2. Smedinghoff, T.J. *Introduction to Online Identity Management*; 2008. Available online: https://www.uncitral.org/pdf/english/colloquia/EC/Smedinghoff_Paper_-_Introduction_to_Identity_Management.pdf (accessed on 1 March 2019).
3. Fett, D.; Küsters, R.; Schmitz, G. The Web SSO Standard OpenID Connect: In-depth Formal Security Analysis and Security Guidelines. In Proceedings of the 2017 IEEE 30th Computer Security Foundations Symposium (CSF), Santa Barbara, CA, USA, 21–25 August 2017; pp. 189–202. [CrossRef]
4. Kaur, E.G.; Aggarwal, E.D. A Survey Paper on Social Sign-On Protocol OAuth 2.0. *J. Eng. Comput. Appl. Sci.* **2013**, *2*, 93–96.
5. Simons, A. *Microsoft Passport and Azure AD: Eliminating Passwords One Device at a Time*; Technical Report; Microsoft, 22 July 2015; Available online: https://techcommunity.microsoft.com/t5/Azure-Active-Directory-Identity/Microsoft-Passport-and-Azure-AD-Eliminating-passwords-one-device/ba-p/244030 (accessed on 5 April 2019).
6. Van Delft, B.; Oostdijk, M. A security analysis of OpenID. *IFIP Work. Conf. Policies Res. Identity Manag.* **2010**, *343*, 73–84. [CrossRef]
7. Roda, M. *Federated Security Domains with SAS and SAML*; SAS Institute Inc., 2015; Available online: https://support.sas.com/resources/papers/proceedings15/SAS1385-2015.pdf (accessed on 15 April 2019).
8. Oppliger, R.; Gajek, S.; Hauser, R. Security of Microsoft's Identity Metasystem and CardSpace. In Proceedings of the Communication in Distributed Systems—15. ITG/GI Symposium, Bern, Switzerland, 26 February–2 March 2007; pp. 1–12.
9. Garzik, J.; Donnelly, J.C. Blockchain 101: An Introduction to the Future. In *Handbook of Blockchain, Digital Finance, and Inclusion*; Chuen, D.L.K., Deng, R., Eds.; Academic Press: Cambridge, MA, USA, 2018; Chapter 8, Volume 2, pp. 179–186. [CrossRef]
10. Gordon, W.J.; Catalini, C. Blockchain Technology for Healthcare: Facilitating the Transition to Patient-Driven Interoperability. *Comput. Struct. Biotechnol. J.* **2018**, *16*, 224–230. [CrossRef] [PubMed]
11. Fernández-Caramés, T.M.; Fraga-Lamas, P. A Review on the Use of Blockchain for the Internet of Things. *IEEE Access* **2018**, *6*, 32979–33001. [CrossRef]
12. Salah, K.; Ahmad Khan, M. IoT Security: Review, Blockchain Solutions, and Open Challenges. *Future Gener. Comput. Syst.* **2017**. [CrossRef]
13. Verma, D.; Desai, N.; Preece, A.D.; Taylor, I. A blockchain based architecture for asset management in coalition operations. *Proc. SPIE* **2017**. [CrossRef]
14. Zakhary, V.; Amiri, M.J.; Maiyya, S.; Agrawal, D.; El Abbadi, A. Towards Global Asset Management in Blockchain Systems. *arXiv* **2019**, arXiv:1905.09359.
15. Lee, J. BIDaaS: Blockchain Based ID As a Service. *IEEE Access* **2018**, *6*, 2274–2278. [CrossRef]
16. Commission, E. *A New Era for Data Protection in the EU*; European Commission: Brussels, Belgium, 2018. [CrossRef]
17. Nabi, A.G. Comparative Study on Identity Management Methods Using Blockchain. Ph.D. Thesis, University of Zurich, Zurich, Switzerland, 2017.
18. Dunphy, P.; Petitcolas, F.A.P. A First Look at Identity Management Schemes on the Blockchain. *IEEE Secur. Priv.* **2018**, *16*, 20–29. [CrossRef]
19. Preuveneers, D.; Rimmer, V.; Tsingenopoulos, I.; Spooren, J.; Joosen, W.; Ilie-Zudor, E. Chained Anomaly Detection Models for Federated Learning: An Intrusion Detection Case Study. *Appl. Sci.* **2018**, *8*, 2663. [CrossRef]
20. Yang, J.; Onik, M.M.H.; Lee, N.Y.; Ahmed, M.; Kim, C.S. Proof-of-Familiarity: A Privacy-Preserved Blockchain Scheme for Collaborative Medical Decision-Making. *Appl. Sci.* **2019**, *9*, 1370. [CrossRef]
21. Khezr, S.; Moniruzzaman, M.; Yassine, A.; Benlamri, R. Blockchain Technology in Healthcare: A Comprehensive Review and Directions for Future Research. *Appl. Sci.* **2019**, *9*, 1736. [CrossRef]
22. *The Inevitable Rise of Self-Sovereign Identity*; Technical Report; Sovrin Foundation: Provo, UT, USA, 2018.
23. Davos-Klosters. *Digital Identity: On the Threshold of a Digital Identity Revolution*; World Economic Forum: Geneva, Switzerland, 2018.

24. Cheetham, M. *Digital Identity: An Introduction*; Technical Report; Piran Partners: London, UK, 2013.
25. Ferdous, M.S.; Poet, R. A comparative analysis of Identity Management Systems. In Proceedings of the 2012 International Conference on High Performance Computing Simulation (HPCS), Madrid, Spain, 2–6 July 2012; pp. 454–461. [CrossRef]
26. Borrows, M.; Harwich, E.; Heselwood, L. *The Future of Public Service Identity: Blockchain*; Reform Research Trust: London, UK, 2017.
27. *A Protocol and Token for Self-Sovereign Identity and Decentralized Trust*; Technical Report; Sovrin Foundation: Provo, UT, USA, 2018.
28. Thomson, L.L. Critical Issues in Identity Management—Challenges for Homeland Security. *Jurimetrics* **2007**, *47*, 335–356.
29. Dhamija, R.; Dusseault, L. The Seven Flaws of Identity Management: Usability and Security Challenges. *IEEE Secur. Priv.* **2008**, *6*, 24–29. [CrossRef]
30. Miculan, M.; Urban, C. Formal analysis of Facebook Connect Single Sign-On authentication protocol. In Proceedings of the Student Research Forum of 37th International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM 2011), Nový Smokovec, Slovakia, 22–28 January 2011; pp. 99–116.
31. *Snapshot of Technical Standards for Digital Identity Systems Technical Standards for Digital Identity*; World Bank Group: Washington, DC, USA, 2017.
32. Sayeed, S.; Marco-Gisbert, H. On the Effectiveness of Blockchain against Cryptocurrency Attacks. In Proceedings of the UBICOMM 2018, the Twelfth International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies, Athens, Greece, 18–22 November 2018.
33. Wood, D. Ethereum: A Secure Decentralised Generalised Transaction Ledger. 2014. Available online: https://gavwood.com/paper.pdf (accessed on 21 March 2019).
34. Triantafyllidis, N.P. Developing an Ethereum Blockchain Application. Ph.D. Thesis, University of Amsterdam, Amsterdam, The Netherlands, 2016.
35. Sayeed, S.; Marco-Gisbert, H. Assessing Blockchain Consensus and Security Mechanisms against the 51% Attack. *Appl. Sci.* **2019**, *9*, 1788. [CrossRef]
36. *Blockchain's Scaling Problem, Explained*; Cointelegraph: New York, NY, USA, 2018.
37. *The Case for Ethereum Scalability*; 2018. Available Online: https://medium.com/connext/the-case-for-ethereum-scalability-d2a8035f880f (accessed on 21 April 2019).
38. Lundkvist, C.; Heck, R.; Torstensson, J.; Mitton, Z.; Sena, M. *UPort: A Platform for Self-Sovereign Identity*; The BlockchainHub: North York, ON, Canada, 2016.
39. Alpár, G.; Hoepman, J.; Siljee, J. The Identity Crisis. Security, Privacy and Usability Issues in Identity Management. *arXiv* **2011**, arXiv:1101.0427.
40. Kalodner, H.A.; Carlsten, M.; Ellenbogen, P.; Bonneau, J.; Narayanan, A. *An Empirical Study of Namecoin and Lessons for Decentralized Namespace Design*; WEIS: Sunbury, PA, USA, 2015.
41. Blockverify. Available online: https://gust.com/companies/blockverify (accessed on 3 March 2019).
42. Cambridge Blockchain. Available online: https://www.cambridge-blockchain.com/ (accessed on 7 March 2019).
43. Nizamuddin, N.; Hasan, H.; Salah, K. *IPFS-Blockchain-Based Authenticity of Online Publications*; Springer International Publishing: Basel, Switzerland, 2018; pp. 199–212. [CrossRef]
44. *uPort: A Platform for Self-Sovereign Identity*; Blockchain Lab: London, UK, 2016.
45. What Is Sovrin? 2018. Available online: https://sovrin.org/faq/what-is-sovrin-2/ (accessed on 9 March 2019).
46. *Travel Identity of the Future—White Paper*; Technical Report; ShoCard: Cupertino, CA, USA, 2016.