TrustVault

A privacy-first data wallet for the European Blockchain Services Infrastructure

Sharif Jacobino
Committee: Dr.ir. J.A. Pouwelse (supervisor), Dr. M.M. de Weerdt, Dr. M.A. Zuñiga
EEMCS, Delft University of Technology, The Netherlands
31-08-2022

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Wednesday August 31, 2022 at 1:00 PM.

TrustVault: A privacy-first data wallet for the European Blockchain Services Infrastructure

Sharif Jacobino and Johan Pouwelse
Distributed Systems, Delft University of Technology
August 2022

***Abstract*—The European Union is on course for introducing a European Digital Identity that will be available to all EU citizens and businesses. This will have a huge impact on how citizens and businesses interact online. Big Tech companies currently dictate how digital identities are used. As a result they have amassed huge amounts of private user data. Movements like Self-Sovereign Identity aim to give users control over their online identity. TrustVault is the first data wallet that gives users back control not only of their identity, but all their data. TrustVault allows users to store all their data on their smartphone and control who they share it with. The user has fine-grained access control based on verifiable user attributes. EBSI connects TrustVault to the European Self-Sovereign Identity Framework allowing users to use Verifiable Credentials from public and private institutions in their access control policies. The system is serverless and has no Trusted Third Parties. TrustVault replaces the for-profit infrastructure of Big Tech with a public and transparent platform for innovation.**

I. Introduction

Internet users today have very little control over where and how their data is stored and used online. Big Tech companies store gigabytes of data about you, and know exactly which online services you use [1]. User data is an extremely valuable asset and is the main source of income for such companies. Billions of people rely on Big Tech monopolies to store their data and voluntarily give up control and ownership over that data. Much of this data is deeply personal and valuable, such as intimate photos of our friends and family. Public and policy trust in Big Tech has been breaking down in recent years (also called the ”techlash”) following major scandals, rampant misinformation campaigns, and a perceived consolidation of power [2]. Nearly five decades after the invention of public key cryptography, we still lack a good solution for people to manage their digital identity and efficiently share encrypted data directly with each other, certainly at a massive scale. There are various movements aiming at halting the power of Big Tech and giving back control to the users. These movements are powered by technologies like blockchain and Self-Sovereign Identity (SSI) which promise to improve the way we interact with online services and with each other. Distributed computing has progressed to a point where a truly distributed identity system, where trust is diffused and not under control of any entity is possible.

Self-Sovereign Identity, sometimes referred to as ”The Internet’s missing identity layer” is an attempt at satisfying the following requirements for a digital identity [3]:

- Security: protecting identity information from unintentional disclosure.
- Control: the identity owner determines who can access their data and under what circumstances
- Portability: user identity must not be tied to a single service or provider

These properties are what makes SSI a tool that will inevitably shift power away from centralised organisations and towards the people.

The European Union (EU) is not unaware of these movements and is ramping up its efforts for bringing transformation into the digital sphere with projects such as Europe’s Digital Decade [4]. In September 2020, the president of the EU declared that a European Digital Identity will be made available to all EU citizen and they all will be able to have a digital wallet [5].

> *”Every time an App or website asks us to create a new digital identity or to easily log on via a big platform, we have no idea what happens to our data in reality.”* Ursula von der Leyen, President of the European Commission

One of the goals for the EU is to improve the way citizens, businesses and public administrations share information and trust each other, and simplify verification processes for cross-border services using blockchain technology [6]. Its proposed solution to reduce our reliance on Big Tech is the European Blockchain Services Infrastructure (EBSI). As at May 2022, there was €57 million in funding for large scale trials [7]. EBSI uses Self-Sovereign Identity to reduce the time and cost of verifying the authenticity of documents and information shared on the EBSI network. EU citizens will be able to download a wallet from the app store and interact with EBSI [8]. Wide-scale adoption will have a significant impact on the digital lives of EU citizens.

While EBSI and SSI in general can make users sovereign over their identity, non-identity data still remains on the servers of centralised applications, not able to be used within other application. If you have had enough of Facebook, migrating your photos to another photo sharing app would be a huge undertaking. It would also be near impossible to completely control who has access to your data on a remote server.

This work aims to solve these problems by developing a data wallet with advanced data sharing capabilities that leverages SSI to provide users with true sovereignty over their data. The contribution of this work is TrustVault: A privacy-first data wallet deployed on the TrustChain Super App. TrustVault consists of a secure data vault and an EBSI conformant digital wallet. The data vault stores the users data locally and provides fine-grained access control (AC) for the stored files. The digital wallet holds Verifiable Credentials (VCs) obtained from the EBSI network and presents these credentials to peers using TrustVault. VCs contain attribute claims that function as access tokens to other users' data vaults. Using VCs as a basis for Attribute-Based Access Control (ABAC) for personal data storage is a novel concept that extends the notion of self-sovereignty over personal identity to personal data. This base implementation lets you browse trough photos of your peers and demonstrates TrustVault's ability to be used for zero-server applications. Users connect directly to your TrustVault and their credentials are automatically matched against your predefined access policies. Users only see the photos that you allow them to see. Our openness-by-design ecosystem encourages permissionless innovation and competition. Anyone is able to develop new applications that can interact with data in your TrustVault. This work shows that multiple SSI frameworks can come together in a way that can increase trust in data sharing applications with societal relevance. Users can switch between SSI frameworks depending on their trust and security needs in terms of data sharing. Integrating with EBSI allows users that credentials with the highest level of trust like digital passports to protect their own data. This works also evaluates IPv8 and TrustChain's ability to be the backing infrastructure of such a data sharing system.

Image of digital passport transfer

II. Problem description

The goal of this study is to design a system that gives users sovereignty not just over their identity, but also over their data. In other words, can we extend the security, control and portability properties of SSI from identity to data in general? The system has to be a part of the critical societal infrastructure being developed by the EU to reduce reliance on Big Tech. Web applications that see a lot of user data are a prime targets for hackers [9]. The reward for disrupting important services and/or stealing confidential data is huge. A lot of effort goes into securing centralised applications with frequent penetration testing, better software development method, hardening techniques like encryption and so on. Yet, even if user data is encrypted, a lot of information can be inferred from the large amount of metadata collected by web applications with statistical analysis, possibly breaching user privacy. Dispersing data throughout a network lowers the risk of large scale data breaches and makes the system more fault tolerant. As long as your data is on a remote server, it is not truly under your full control. Soft access control is hard to enforce if parties can be malicious [10]. Hard access control (enforced with cryptography) is either not very flexible when using public-key cryptography or introduces Trusted Third Parties (TTP) in the case of most Attribute-Based Encryption (ABE) schemes [11]. Most importantly, even systems that offer fined-grained AC without TTPs like distributed ABE schemes do not prevent censorship [12] by centralised applications. Data portability is a personal right established in the General Data Protection Regulation (GDPR) [13]. This is in direct conflict with the desire of companies to retain users and their data. Data is often tightly coupled to the application, complicating transitioning data between services. Regulations and public pressure is forcing companies to adopt or support standard formats. Data still has to be exported from the one application and imported into the other. This step can be simplified or even eliminated.

A system where users have true sovereignty over data has to have the following properties:

- Data storage has to be decentralised on devices under the control of the data owner.
- Access control has to be decentralised, fine-grained and resolutely enforced.
- Data has to be decoupled from applications.

Applications access user data at the discretion of the user. Certain applications require users to access data on another user's device. The requesting user has to satisfy the access policy (AP) set in place by the host user for the desired data. Secure AC requires that a user's authentication be verified before enforcement [14]. SSI solves this problem in a way that keeps users in control of their identity. Actually, SSI makes it possible to have any attribute of a user to be verifiable through VCs. APs can be defined in a fine-grained manner for arbitrary verifiable attributes.

EBSI can be the connecting piece to the societal infrastructure for identity. EU citizens will have credentials from public and private institutions such as drivers licence, diplomas and club membership in digital form. These can all be used to enable the automatic sharing of data between EU citizens on the basis these credentials. In section III we elaborate on concepts relevant to this work. In section IV the architecture and design of TrustVault is presented and in section V we discuss the implementation and evaluate the system. In section VI we go over related work and we end with a conclusion and future work proposals in section VII.

III. Background

A. Self-Sovereign Identity

SSI is a decentralised model of digital identity developed to address the shortcomings of the previous internet identity models [15]. With centralised identities, centralised institutions such as governments and banks issue credentials that allow citizens to interact with services and each other. On the internet you would establish an account with every website, service or application. In this model, all the data about you belongs to the issuing party, can't be reused, and is out of your control. The federated identity model introduces identity provider (IDP). IDPs allow you to have one account that can be used to interact with any service that supports that IDP. This is the mechanism behind the social login buttons (Login with Facebook) widely found on the internet today. Federated identity simplified managing accounts for every service to managing a few accounts at a few IDPs. All our identity data, and information about when or how we use our federated identities is now concentrated in these Tech Giants, raising a lot of privacy concerns.

The rise of blockchain technology inspired the decentralised identity model. This model is not based on accounts with centralised institutions or IDPs but on direct relationships between peers. No party controls or owns the relationship. Users are in full control of their identity data, how it shared and with whom. Peers establish private connections by securely exchanging public keys whereby blockchains serve as decentralised public key infrastructures. This model closest resembles how we manage our identities in the real world: with wallets containing credentials obtained from trusted parties which can shown to other parties to initiate an interaction. There are several deployed decentralised identity (DID) frameworks built on top of ledgers purpose-built for decentralised identity like Sovrin [16] (based on the Hyperledger Indy framework[17]) and ledgers repurposed for SSI, such as [18] (using TrustChain [19]) and Ethereum [20].

Verifiable Credentials are the building blocks of SSI. Much like physical credentials, VCs contain claims about your identity that some authority claims is true about you. You can then use this VC to convince others that trust said authority of the validity of these claims. The trust relationship between issuers, holders/provers, and verifiers is shown in figure 1. Issuers put digital signatures on credentials that are cryptographically verifiable. They are trusted to issue true credentials and to be authoritative on the attributes that they attest to. Verifiers request proofs about identity claims they need to be convinced of. They do not need to have any direct relationship with issuers. They just need to trust an issuer's ability to make correct assertions. Holders ultimately have the choice to respond to a request with a Verifiable Presentation (VP): a VC with a digital signature of the prover. Holders trust verifiers to keep their credentials confidential. The Verifiable Data Registry (VDR), where DIDs, public keys and schemas are registered, needs to be trusted by every party to be accurate and tamper-evident. That is why public ledgers are a good fit for the function of VDR. The holder's credentials and cryptographic keys are stored in a digital wallet. The wallet is trusted to securely store VCs. Digital agents wrap users' digital wallets and establish communication with other agents to exchange credentials.

B. European Blockchain Services Infrastructure

European Blockchain Services Infrastructure (EBSI) is a distributed network that runs a public blockchain to host public and private services that want to leverage the benefits of blockchain technology. Their objective is to offer secure and private cross-border public services among EU member states. The main services that EBSI aims to facilitate are:

1) Notarization: using the blockchain to make digital audit trails and automate compliance checks.
2) Diplomas: giving citizens control over their educational credentials and lowering the cost of verifying documents.
3) European Self-Sovereign-Identity Framework (ESSIF): serve as a verifiable registry and communication channel for an SSI framework across Europe.

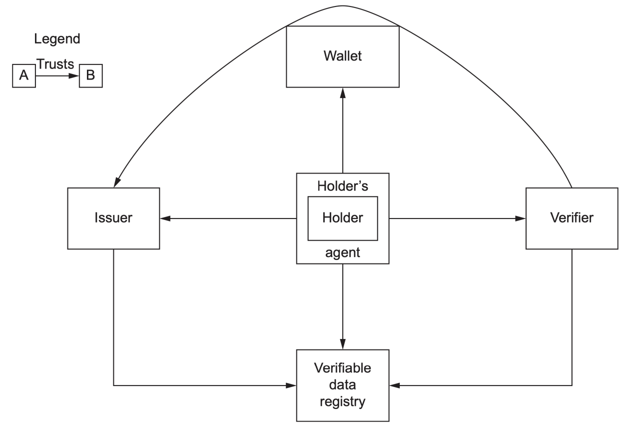

Fig. 1. VC trust model [15].

Relevant to this work is ESSIF, enabling the exchange of VCs on EBSI. This service encourages European citizens to adopt SSI to improve the identity verification process with public services and private companies across European borders. The EBSI blockchain serves as the VDR in the ESSIF framework, where public keys of users and trusted applications can be looked up.

The EBSI architecture consists of three layers: the Infrastructure layer, the Chain and Storage layer, and the Core Service layer. The Infrastructure layer contains the element required to set up an EBSI node and form a network. Every EU Member State is allowed to run nodes, distributing trust over all the members. The Chain and Storage layer contains the blockchain protocols and adds off-chain storage. This is where the smart contracts for the different verifiable registries such as the DID registry, the Trusted Issuers Registry (TIR) and the Trusted Schemas Registry (TSR) are defined. These elements are segregated to make it possible to interoperate with different blockchain network. The Core Service layer is the interface to the lower layers. It contains the API endpoints to interact with the verifiable registries and secondary services like the Notifications service.

C. Attribute-Based Access Control

Attribute-Based Access Control (ABAC) is an access control model that controls access to objects by evaluating rules against attributes of entities [21]. This allows for fine-grained AC because of the large set of possible combinations of attributes that can feed into an AC decision and consequently large set of possible rules for policies, only limited by the richness of the available set of attributes. ABAC makes it possible to define AC policies without prior knowledge of who will need access and there is no list that needs to be modified in order to accommodate new users. AC decisions are purely based on the presented set of attributes. An important requirement for ABAC is that attribute values are correctly associated with the subject.

IV. System Architecture and Design

In this section we discuss how the different internal and external components come together to form the TrustVault architecture. We then go into how we integrated Verifiable Credentials into the access control mechanism to achieve fine-grained access control. We then discuss the design for a tamper-proof access log. Finally we explain the security measures taken to protect data in TrustVault.

A. Architecture

TrustVault is a mobile agent consisting of two parts: a secure data vault and a digital wallet. A software agent is a computer program that can act on behalf of an individual autonomously[1]. TrustVault autonomously enforces the users APs for the data vault and manages the credentials in the digital wallet. The data vault (DV) uses IPv8 networking protocol for peer-to-peer (P2P) data sharing. IPv8 is a fully decentralised architecture for private and authenticated communication [22]. Peers communicate directly with each other, without the need of servers, protecting their privacy. The protocol is built around communities that represent distinct services. Communities provide the ability for peer discovery and define service-specific messages that can be exchanged between peers. The DV has it's own community that implements the data vault protocol. The DV protocol is based on 5 messages: *accessibleFilesRequest*, *accessibleFilesResponse*, *fileRequest*, *fileResponse* and *fileRequestFailed*. IPv8 abstracts away physical addresses and allows peers to be identified by their public keys. Connection between peers are maintained when IP addresses change, even behind NAT boxes and firewalls by using a UDP hole-punching technique. The user is able to select a peer to interact with from a list of all the peers in the DV community.

The DV functions as a personal file server to the DV community. The latest smartphones have storage capacities rivaling laptops. We are also used to having a large amount of personal files, mostly photos, on our smartphones. Mobile internet speeds are also approaching landline internet speed[2], especially with the rollout of 5G. The data vault stores and organises data in a closed-off directory on the phone's file system. The digital wallet also stores VCs and key material in a closed-off directory and interacts with the EBSI Core Services Layer via HTTP requests. This architecture assumes

[1]https://www.britannica.com/technology/software-agent

[2]https://www.statista.com/statistics/689876/average-mobile-speeds-download-and-upload-in-western-europe/

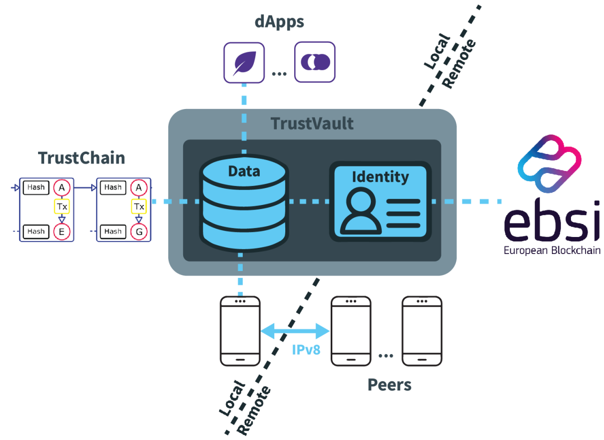

Fig. 2. TrustVault Architecture.

EBSI and its infrastructure, including the Ethereum network has negligible downtime.

TrustVault's open architecture encourages the development of new applications that that can read and write data to the user's DV. Different applications can provide different ways of interacting with data in users' vault. This makes for a more competitive ecosystem as user data is completely portable between applications.

B. Access control

Files and folders, including the root folder, have an associated meta-data file that includes the file or folder's local AP $\pi(f)$. To access a file, the file's global policy $\Pi(f)$, meaning every policy along the file's root path, must be satisfied. With $P(f)$ denoting the parent folder of f, $P(root) = \emptyset$ and $\Pi(\emptyset) = \emptyset$, global policies follow this recursive definition: $\Pi(f) = \pi(f) \wedge \Pi(P(f))$. Practically this means that policies are inherited from parent folders. An effective way of setting APs is to have minimal or no restrictions on the root folder and have increasingly specific and restrictive policies for sub-folders.

An AP is a binary boolean expression tree and the leaves are attribute rule expressions that are evaluated at access time. Attribute rules are triplets in the form of $(attribute, operator, value)$. An example policy would be $(age \geq 18 \wedge (university = TU\ Delft \vee issuer = me))$. To satisfy this policy, the prover has to present a VC that asserts that their age is over 18, e.g. a government ID, and either a proof-of-enrolment from the TU Delft or a VC issued by the verifier (owner of the TrustVault). The VC is first verified by the wallet and then evaluated against the global AP. The $age \geq 18$ rule can be satisfied with a predicate proof. A predicate proof proves a boolean statement about a value without having to reveal the value. The user interface lets the user add or remove nodes to the policy tree. Additionally, the user can define read+write APs or separate read and write policies for more granular control.

In a protocol run, the prover is referred to as the requester and the verifier as the host. The requester first makes an *acces-*

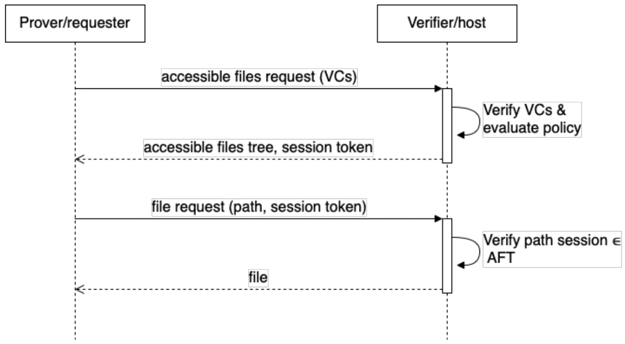

Fig. 3. File exchange between TrustVaults.

sibleFilesRequest. An *accessibleFilesRequest* must included a set of VCs. The host returns a *accessibleFilesResponse* containing a directory sub-tree of the file paths with global APs satisfied by the provided VCs along with a randomly generated session token. The directory sub-tree is used to dynamically recreate a copy of the host's DV on the requesting device. The actual files are retrieved on demand (*fileRequest*) on the requesting device to prevent retrieving files that are not needed. Retrieved files are cached to avoid having to fetch the same files multiple times, without storing them permanently on the device. *fileRequests* include the provided session token so that whole credentials don't need to be sent and verified with each file request. Session tokens are cached on the host, mapped to the corresponding sub-tree, with an expiry time that is extended with each new request. A request with an expired session token fails and the requester is notified with a *fileRequestFailed* message to make a new *accessible files* request. The cached session token to sub-tree mapping ensures that no files are served that aren't covered by the original VCs. This interaction is depicted in figure 3. Session token JWT containing AFT. No caching on host side

C. Verifiable Credentials

Upon first launch of TrustVault, an EBSI DID and an Elliptic Curve (EC) key pair is created and registered with EBSI. Verifiers are then able to lookup the user's public key in EBSI's DID Registry. Subsequently the user can obtain VCs from trusted issuers on the network. These could private and public entities all over the EU, making the set of attributes for which AP rules can defined very diverse.

When requesting access to a peer's vault, the prover selects a set of VCs to assemble into a VP. A VP lets the prover authenticated itself as the holder of the enclosed VCs. This mechanism lets the prover keep control over their identity by enabling the prover to decide which credentials to disclose. Coincidentally, the verifier is given confidence that their ABAC policies ensure that only users that authentically posses the required attributes can access their data.

D. Self-issued credentials

VC meta-data contains data not related to the identity of the credential subject such as the issuer and issuance date for which AP rules can also be made. We make use of this feature to create a new set of APs based on Self-issued Credentials (SICs). SICs serve a similar but more expressive function than follow/friend requests in traditional social networks. The issuer can add extra attributes to a SIC to make the context of the social connection more specific. This is particularly useful when you want to control access based on claims that won't be asserted by a trusted issuer. For example, you have some photos you took with some people you met on holiday in Italy. You can issue a credential to them that asserts that you have met on holiday giving them access only to the photo's in your vault with the corresponding AP. It is possible to model complex social connections in this manner, making TrustVault a well-suited data store for decentralised social applications. EBSI only allows VCs to be issued by trusted issuers. These are parties that have undergone a separate verification and/or accreditation process to be registered on the Trusted Issuers Registry (TIR). However, SICs are only intended to be presented back to the issuer. SICs can therefore be exchanged directly between peers, bypassing EBSI. The issuer can verify a SIC without having to consult the TIR. Besides EBSI VCs, TrustVault also support TrustChain IDentity (TCID) which inherently supports SICs. In TCID each agent has their own list of Trusted Issuers eliminating the need to consult an external registry altogether.

E. Tamper-proof access log

Access control is completely automated without intervention of the TrustVault owner. This makes it impossible to keep track of who has been given access to which files. This is remediated by recording *accessibleFilesRequest* on TrustChain for each session. The owner sends a transaction to the requester with a bloom filter containing the accessible files from the request. A bloom filter is a randomized data structure for representing a set of elements that supports membership queries with no false negative and a small false positive probability [23]. This forms a timestamped, tamper-proof record of the files made accessible to the requester. TrustChain transactions have to be signed by both the sender and the recipient. The requester's approval of the record is thus made irrefutable. This also guarantees the integrity of the record. In case of an audit or dispute, this record can be referenced and the bloom filter can queried to proof with high probability that a specific file was offered to a specific user.

F. Data protection

As a data wallet for EU citizens, it is crucial that personal data and the user's right to privacy is protected in line with the GDPR. An essential measure is to have data in the system be encrypted at rest and in transit. When the TrustVault is inactive, all files are encrypted with AES in Counter mode. Counter mode is great for encrypting/decrypting large amounts of data because blocks can be processed in parallel. This includes VCs stored in the wallet. A password is required to "unlock" the TrustVault and "lock" it again when closing the app. The encryption key is derived from the password using PBKDF2. When transmitting files, IPv8's end-to-end encryption is used. Data packets are asymmetrically encrypted for the recipient and signed for confidentiality, integrity and authenticity of transferred files.

V. Implementation and Evaluation

This section describes the implementation process of TrustVault and the digital wallet for EBSI specifically. We then evaluate the system's privacy protection and security. We explain our experimental setup and provide some insight on the system's performance.

A. Implementation

TrustVault is made for Android and is implemented entirely in Kotlin[3]. It is part of the TrustChain Super App, the collection of decentralised apps running on IPv8 and the TrustChain ledger. The codebase includes a fork of walt.id SSI kit. The open source code for SSI kit is also written in Kotlin. However it is developed as a command line tool and does not run on Android out of the box. Changes needed to make it compatible with Android include modifications to IO operations with the file system and replacing HTTP and crypto libraries not available on Android.

Structure of code base. LOC.

Before settling on developing TrustVault, work was done on the Super App's messaging app, implementing features like contact sharing to familiarize IPv8. Making an intuitive user interface (UI) to edit access policies on a small screen device is a challenge. The current UI does not reflect the tree-like structure of an AP. Instead, the linear layout enforces a linear evaluation of APs. A policy $(A \circ B \circ C \circ D)$ would be evaluated as $(A \circ (B \circ (C \circ D)))$. The shape of policy trees is thus limited to be consistent with what the user expects from the UI. The UI for TrustVault is a file browser interface that lets you explore the photos in your or a peer's data vault and set APs for your own photos. Figures 4, 5, 6 and 7 show screenshots of the browser interface for local files and peer browsing, and the access control management interfaces.

TrustVault is designed to be a secure data wallet for EBSI users. The process of getting TrustVault EBSI conformant has not been straight forward and is still ongoing. The first prototypes were built using the early versions of the TypeScript cef-ebsi packages to interact with EBSI v1 [24] as part of the EBSI Early Adopters programme[4]. In v1, all operations were API calls to test endpoints. In v2, critical operations including creating, signing, and verifying credentials were moved from the endpoints to libraries running on the user's device. At this point, there were 3 documentation sources for implementing an EBSI that were out of sync in several places and there was no official library for Android, meaning that there was

[3]https://github.com/Tribler/trustchain-superapp

[4]This work was facilitated and sponsored by The National Office for Identity Data (RvIG) of the Dutch government.

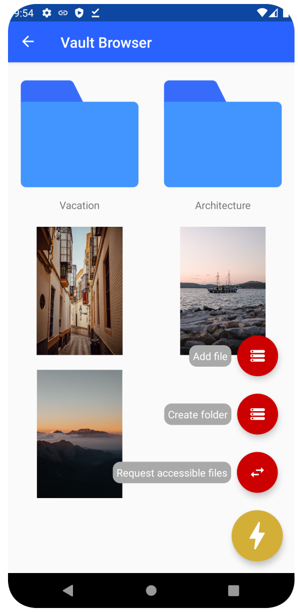

Fig. 4. Data vault browser.

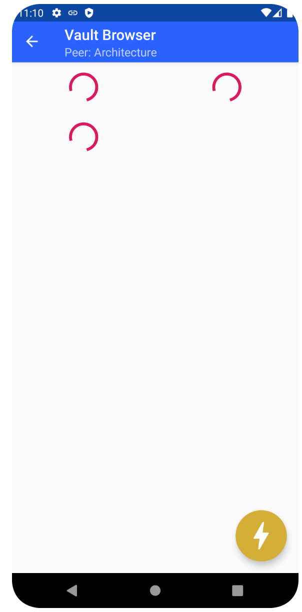

Fig. 5. Loading images from a peer.

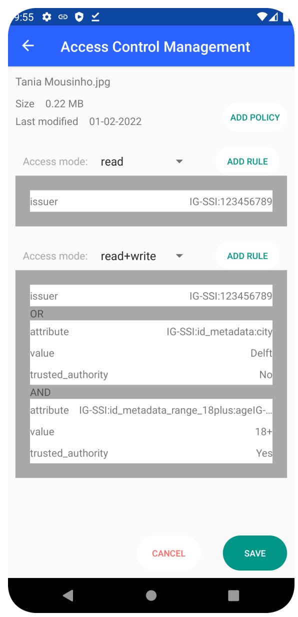

Fig. 6. Access Control Management.

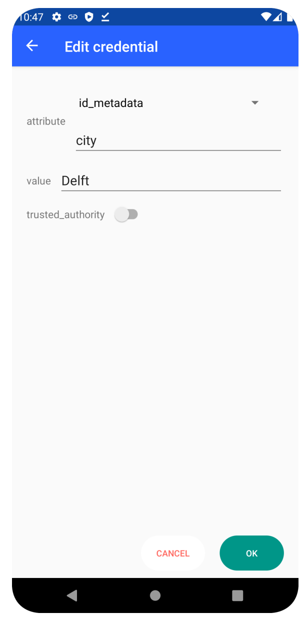

Fig. 7. Edit policy credential.

a lot of trial-and-error to get the API connection working[5][6]. As some wallets started passing the conformance tests, EBSI started publishing test reports that included correctly formulated HTTP requests for the different APIs. We were able to use some of these, including, onboarding, authentication and authorisation requests to validate our own implementation up to that point. The open-sourced walt.id SSI kit was chosen to

[5]Discrepancies in documentation and trial-and-error: https://github.com/Tribler/tribler/issues/6023#issuecomment-908087676

[6]Contact with EBSI support about down time and errors: https://github.com/Tribler/tribler/issues/6023#issuecomment-1104821838

generate and verify credentials as it is more feature complete in that area.

When initializing TrustVault, the user needs to complete the EBSI onboarding process which entails scanning a QR-code on the onboarding page to get an authentication token that is used to get permanent authorisation. In subsequent sessions, the authorisation token is exchanged for a short-term session token that needs to be included in every API request.

B. Privacy

Privacy in TrustVault can be analysed from the perspective of the TrustVault owner and from the requesting party. One of the main goals of this work is to give users control over their data and thus over their privacy. The first step is to enable users to self-host their data, stopping data-hungry companies from running machine learning algorithms over user data and learning users' behavioral patterns. This has the added benefit of disrupting Big Tech from monetizing user data. Giving the user fine-grained access control allows the user to have specific disclosure policies at the desired granularity level, down to file level. This comes with great responsibility, as there is the opportunity making mistakes when defining access policies and disclose data to unintended parties. The challenge is to make user experience simple and intuitive to minimise the chances for mistakes. Hosts are encouraged to exercise data minimisation: the practice of requesting only the minimum amount of information necessary for an operation. In this context it means not having policies that require provers to reveal an unnecessary amount of (personal) information. The requesting party meanwhile has full control over its identity. Selective disclosurability allows the requesting party to only present information it is comfortable disclosing.

Identification by static public keys does present the possibility of learning information over time. The host can keep record of every time a certain public key wants to access data, which arguably is a sensible thing for the host. However, the host is able to link different access requests over time while collecting the credentials presented at each request, possibly accumulating a more revealing, or even identifiable set of attributes of the requesting party. Entities are able to have multiple DIDs for different contexts in SSI. This reduces the linkability of credentials to an entity. However, users still have one public key by which they are identified in an IPv8 communities, voiding the benefits of having multiple DIDs in EBSI. The Python implementation of IPv8[7] has Network-Level Anonymity which mitigate credential linkability and correlation attacks. The Kotlin implementation[8] however does not have this feature. TrustChain does not support private/anonymous transactions. By logging access request on-chain, interactions between parties become publicly visible. Anyone can keep track of when and how often one

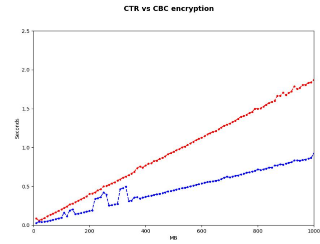

Fig. 8. AES encryption CTR vs CBC mode.

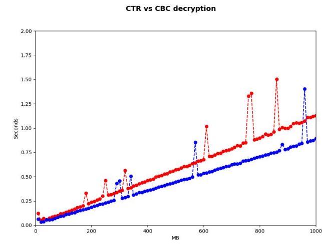

Fig. 9. AES decryption CTR vs CBC mode.

public key requests access from another public key, potentially leaking information.

C. Security

The security of TrustVault depends on the security of the data vault and the digital wallet. The data vault's main task is keeping data confidential. The Android internal file storage protects files from being accessed from outside the Super App [25]. This offers the first layer of protection. Additionally, the data vault is encrypted using AES when the application is not in use. When opened, a password is required to decrypt the data vault. This prevents unauthorised access even if someone gets physical access to the device and launches the application. Data is also protected in transit with IPv8. Packets are encrypted with the public key of the recipient and signed for authenticity and integrity.

TrustVault inherits the VC trust model. EBSI can be trusted to be tamper-evident in fulfilling the role of VDR by virtue of using public blockchain. It is less convincing in meeting the requirement of accuracy because there is a layer between users and the blockchain where read and write operations could be corrupted. The likeliest way an attacker could get access to data not intended to be disclosed is by getting a false VC from a malicious, compromised or incompetent issuer. Issuers that have a reputation to protect are incentivised to be honest. EBSI tries to facilitate this by having an accreditation process for issuers on the TIR. Ultimately the verifier decides who to trust. TCID gives more control to the user in this are by having a personal Trusted Issuer registry. Credentials can also be revoked in TCID resulting in better credential accuracy.

There are several threats to data availability. The first threat is the lack of redundancy. All the user's data is on a mobile device that can temporarily or permanently be out of service for a number of reasons. If the data vault is not backed up on a more reliable medium, the user is at risk of losing data if the device becomes permanently inaccessible. Limited battery capacity and fluctuating internet speed occasional drops in service level can be expected. Communication on the protocol level is more robust. IPv8 maintains network connectivity between peers even with changing IP addresses and firewall protection. While EBSI uses a distributed ledger, interaction with the ledger goes via the hosted Core Services layer. These hosted services can be a single point of failure that can disrupt VC verification.

[7]https://github.com/Tribler/py-ipv8

[8]https://github.com/Tribler/kotlin-ipv8

	# verifications	overhead
Session token	1	> 300B
TCID	n	$> n \times 1800$B
EBSI VC	$n+1$	> 200B $+ n \times 1300$B

TABLE I
NUMBER OF VERIFICATIONS AND DATA OVERHEAD PER REQUEST.

#	Phone	# requesters	Δ
1	WiFi	1	5
2	4G	1	5
3	WiFi	2	5
4	WiFi	2	2
5	WiFi	2	1

TABLE II
NUMBER OF VERIFICATIONS AND DATA OVERHEAD PER REQUEST.

The tamper-proof access log does not provide indisputable proof that a requester actually retrieved a file. The record only claims that the requester could access a set of files based on the provided VCs. A malicious host could add files to the bloom filter that are not actually accessible for the requester. This would be difficult for the requester to detect.

D. *Experimental analysis*

For TrustVault to be a viable solution for storing and sharing data, file transfer including the time it takes to verify access tokens needs to be as quick as possible. The factors that determine transfer time are Internet connection speed and latency, transfer protocol speed, and access token verification time which is in turn dependent on the available computing power. Internet connection speed sets the upper limit for achievable transfer speeds but will likely not be the bottleneck. The data transfer protocol used is a connection-less data transfer protocol based on the Trivial File Transfer Protocol [26]. The protocol works around the unreliability of User Datagram Protocol (UDP) to create protocol that can be used for P2P data transfer. An average transfer speed of 260kB/s is achieved over WiFi on the same network and 210kB/s over 4G. File size is capped at 250MB to avoid running out of memory when reconstructing packets. The different access token types incur distinct penalties on transfer time because of the different verification methods. Session tokens need to verified when presented which requires an EC signature verification. Depending on the size of the encoded *accessibleFilesTree*, a session token adds at least 370B to a request. TCID attestations also require an EC signature verification. However, if multiple attestations are included in a single request, each has to be separately verified. Each attestation also adds at best 1800B to a request. Similar to TCID, each EBSI credential in a VPs requires an EC signature verification but an extra verification is to authenticate the holder of the VCs. Whereas TCID has a local register of Trusted Issuer public keys, the DID of the issuer and the holder has to resolved with the EBSI DID Registry API. That's an additional overhead of two HTTP request per VP. EBSI access token adds around 200B for the VP metadata and 1300B at a minimum for each included VC to a request. The EBSI VCs in this experiment contain three attribute claims. We therefor chose to use three TCID credentials, each attesting to one claim. These figures are summarised in table I.

E. *Experimental setup*

The experimental setup consists of one phone, a Samsung Galaxy S8 with 4GB RAM, 4x2.3 GHz + 4x1.7 GHz 8-core CPU running Android 9, and two MacBook Pro's running Google Pixel 4 emulators with Android 11. The Galaxy S8 serves as the host TrustVault and the emulators function as requesters. All verification and access control is thus done on the Galaxy S8. All the devices are connected to the same WiFi network but one MacBook has a VPN connection outside of the local network. We ran 5 experiments detailed in table II. Each experiment makes 50 requests for each access token type and a baseline with no access token. The time between requests Δ is changed to vary the amount of requests per second that the phone needs to handle. Transfer time is measured from the moment a request is sent to the moment a response is received. A timestamp is included in the request metadata that is returned in the response and the difference is taken from the timestamp at which the response is received. A single file of 220kB is returned in the response. The experiments are running with an automated Kotlin script that generate logs that are processed in Python.

F. *Experimental results*

The result of the single requester experiments are shown in figure 10. The results confirm a slower transfer speed when on 4G than on WiFi. The average transfer time increased 138%, 157%, 52% and 54% respectively for the 4 categories. This is a bigger decrease in speed from WiFi to 4G than that measured in [26]. Session token verification is on par with the baseline measurements. The extra signature verification only takes on average 130 ms. It takes about 180 ms to verify the TCID credentials but the bigger request size seems to have the biggest impact on the transfer time. On average TCID is about 1.5 seconds slower than baseline. EBSI VPs take the longest to verify at around 340 ms. Yet the overall transfer time is quicker than TCID. A smaller request size is more optimal. This same pattern holds with the transfer times over 4G.

The two-requester experiment results are shown in figure 11. With two devices sending requests at a 5 second interval, the baseline and session token runs mirror the single requester runs. TCID and EBSI VP take a small hit of about 1 second. As the interval is decreased to 2 and then 1 second, the phone is still able verify the access tokens nominally, yet the transfer times gradually increase to well over a minute in the case of TCID and EBSI VP (times above 20 seconds are filtered out to keep the plots legible).

G. *Performance evaluation*

The transfer when the device is not congested is in line with the results of [26]. In the baseline single requester case, the mean transfer time for a 220kB file is 780 milliseconds. The request would have negligible impact on the transfer time as it is just a couple of hundred bytes meaning the file response

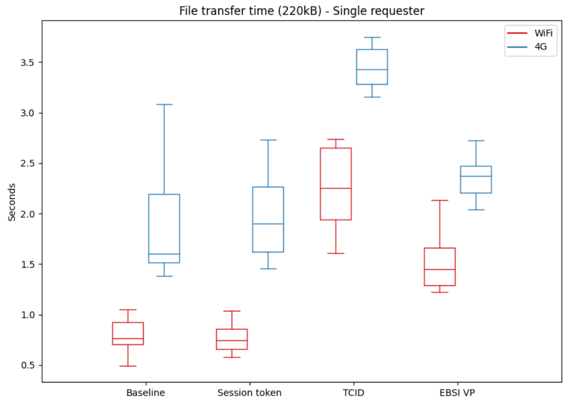

Fig. 10. File transfer Time (200kB) single requester.

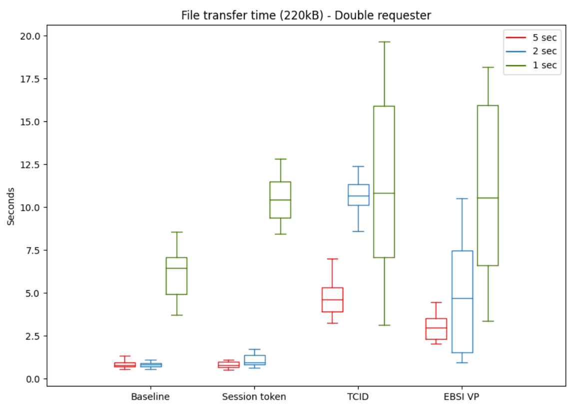

Fig. 11. File transfer Time (200kB) double requester.

is the dominant factor. Considering this, the transfer speed can be approximated at around 280kB/s. However, even light congestion causes the transfer time to to become impractically high in the order of tens of seconds for a single file. This is a baked in limitation of IPv8 using UDP.

On the other hand, the use of access tokens with each request does not diminish the usability of the system, especially when converting TCID and EBSI credentials into session tokens. The results show the time penalty incurred for the different types of access tokens on the retrieval of a single file. In practice multiple files are retrieved with one request, spreading the verification time penalty across multiple files.

At the moment it is fair to say that large scale data sharing is not achieved yet.

VI. Related Work

Solid is an open-source protocol that let's people store their data securely in decentralised data stores called Pods [27]. Pods are personal web servers that can store any kind of data as Linked Data. Linked Data is data with semantic links to other data recorded in its metadata such that computers can explore these links using semantic queries. Pod owners have granular control over who has access to the data. Solid uses the WebAccessControl system, which is based on access control lists with user identification by WebID, to grant and revoke access to any slice of data contained in a Pod to individuals, organizations, or applications. WebID is a protocol to allows persons, organizations or other types of agents to create their own unique identities and embed links to other people or objects using Resource Description Framework [28]. WebID makes it easy to make arbitrary claims about yourself but those claims are not trustworthy as they are not verifiable. At best they provide authentication by proving possesion over a private key. Access control rules can be made based on the agent's properties found in their profile document. Solid applications are client-side, mobile or web, that read data straight from users' pods. Users can switch from one application to the other because data is decoupled from applications by design.

TrustChain is a Sybil-resistant permissionless blockchain [19]. Transactions are signed by both parties and blocks are chained together to the previous block of both party. Each maintains their own chain that is tangled with the chains of parties they transacted with. There is no global chain containing every transaction over which consensus has to be made. Modifications or reordering of blocks on one chain can be detected on the chains of counterparties. This way consensus is achieved between participants of a transaction instead of on a global level.

TCID is an SSI system designed with performance and security at the networking layer in mind [29]. TCID provides the properties of Self-Sovereignty and Credibility, but crucially also Network-level Anonymity. Network-level Anonymity is achieved when source and destination addresses are obfuscated. Without this property it possible to carry out correlation attacks on credentials exchanges over time, undermining the data disclosure protections of SSI. TCID solves this problem by adding an anonymisation layer on top of the communication layer. The anonymisation layer routes identity-based messages through a multi-hop communication channel of randomly selected peers. Increasing the number of intermediaries improves anonymity but also increases latency. TCID supports credentials with zero-knowledge proofs (ZKPs), including ZKP range proofs.

[30] extends TCID with a distributed revocation mechanism. A gossip protocol is used to propagate revocations through a network. Accepting a revocation is at the verifier's discretion. Verifiers keep their own local registry of Trusted Issuers that inform decisions on both verification and revocation.

[31] proposes using an ABAC scheme based on DIDs, similar to the scheme proposed in this work. The system is used to control access to a platform. The platform is the only verifier in the system and there are 3 established issuers. There is actually no requirement for identity portability or protection from unintentional disclosure because the platform is the sole intended recipient of all credentials.

There are multiple research that looked into decentralizing

ABAC. [32] and [33] proposes using a blockchain for policy enforcement. The task of making policy decisions is handed over to smart contracts. APs and user attributes are stored on smart contracts. When access is requested to a resource, a request is made to a smart contract that based on the policies and attributes stored on chain. The decision is returned to the server enforcing the AP. This approach makes it possible to have the blockchain serve as a decentralised escrow for digital assets. An on-chain access log is automatically created recording the AP decision, removing the need to have a separate logging mechanism. A drawback to this approach is that updating policies is costly as that requires write operations on the blockchain. Every access request has to go through a smart contract, introducing some latency.

The SSI Kit by walt.id is a Self-Sovereign-Identity open source solution, primarily focused on the EBSI/ESSIF ecosystem[9]. It provides building blocks for key management, issuing, presenting and verifying credentials, and specific EBSI-related functions. Walt.id developed one of the earliest EBSI conformant wallets.

VII. Conclusion and Future Work

This work presents TrustVault, a system where users are sovereign over both their identity and their data. TrustVault is a unique first operational system that builds upon the upcoming European Digital Identity Wallet. Alternatives for Big Tech is an emerging topic of research. In our system, users are not reliant on Big Tech companies to authenticate themselves nor store and host their data. User data is stored in a data vault on a device under the control of the user. ABAC is used to achieve fine-grained access control to the data vault while leveraging the wealth of verifiable attributes available in a SSI context. We show that EU's EBSI initiative is a viable way to give control to the citizens of the EU by integrating it into our system. Compared to related work such as Solid, there is no infrastructure and system management burden for the user. The user does not need to understand the inner workings of data management. Our work presents the first data and identity wallet solution with fine-grained access control over both. It is possible to have a fairer and more competitive system than the for-profit infrastructure of Big Tech, that is public, transparent, and open source.

Future Work

TrustVault can be expanded to support other SSI networks like Sovrin and many built on Ethereum. This would open the door to even more types of credentials and attributes to include in access policies. TCID supports some ZKPs but there are currently more proof schemes in development like BBS+ signatures[10] that provide selective disclosure, signature blinding and private holder blinding. These schemes further improve user privacy. Network-Level anonymity, which is already implemented in Python, could be implemented in Kotlin as well. This would mitigate the correlation attacks possible in the system as is. Improving the UI to better reflect the structure of APs could allow the user to intuitively set up more complex and expressive policies. For critical data with high availability requirements, having a fallback device could be a great capability. Redundant devices could be deployed simultaneously for load balancing or simply as a back-up. Finally, applications can be developed that makes use of the TrustVault infrastructure to provide useful services to TrustChain Super App users.

[9]https://github.com/walt-id/waltid-ssikit

[10]https://www.evernym.com/blog/bbs-verifiable-credentials/

References

[1] D. Curran. (2018) Are you ready? Here is all the data Facebook and Google have on you. [Online]. Available: https://www.theguardian.com/commentisfree/2018/mar/28/all-the-data-facebook-google-has-on-you-privacy

[2] K. Birch, D. Cochrane, and C. Ward, "Data as asset? the measurement, governance, and valuation of digital personal data by big tech," *Big Data & Society*, vol. 8, no. 1, p. 20539517211017308, 2021.

[3] A. Tobin and D. Reed, "The inevitable rise of self-sovereign identity," *The Sovrin Foundation*, vol. 29, no. 2016, p. 18, 2016.

[4] European Commission. (2021) Europe's Digital Decade. [Online]. Available: https://digital-strategy.ec.europa.eu/en/policies/europes-digital-decade

[5] ——. (2019) European Digital Identity. [Online]. Available: https://ec.europa.eu/info/strategy/priorities-2019-2024/europe-fit-digital-age/european-digital-identity_en

[6] ——. European Blockchain Services Infrastructure. [Online]. Available: https://ec.europa.eu/digital-building-blocks/wikis/display/EBSI/Home

[7] ——. (2022) EBSI Grants. [Online]. Available: https://ec.europa.eu/digital-building-blocks/wikis/display/EBSI/EBSI+Grants

[8] Thales Group. The EU Digital ID wallet is coming. [Online]. Available: https://www.thalesgroup.com/en/worldwide-digital-identity-and-security/government/magazine/eu-digital-id-wallet-coming-heres-what

[9] B. Musa Shuaibu, N. Md Norwawi, M. H. Selamat, and A. Al-Alwani, "Systematic review of web application security development model," *Artificial Intelligence Review*, vol. 43, no. 2, pp. 259–276, 2015.

[10] L. Pesonen, D. Eyers, and B. Jean, "Access control in decentralised publish/subscribe systems," *Journal of Networks*, vol. 2, 04 2007.

[11] S. Müller, S. Katzenbeisser, and C. Eckert, "Distributed attribute-based encryption," in *International Conference on Information Security and Cryptology*. Springer, 2008, pp. 20–36.

[12] J. Lee, B. Lee, J. Jung, H. Shim, and H. Kim, "Dq: Two approaches to measure the degree of decentralization of blockchain," *ICT Express*, vol. 7, no. 3, pp. 278–282, 2021.

[13] GDPR Right to data portability. [Online]. Available: https://gdpr-info.eu/art-20-gdpr/

[14] R. S. Sandhu and P. Samarati, “Access control: principle and practice,” *IEEE communications magazine*, vol. 32, no. 9, pp. 40–48, 1994.

[15] A. Preukschat and D. Reed, *Self-sovereign identity*. Manning Publications, 2021.

[16] Sovrin. [Online]. Available: https://sovrin.org

[17] Hyperledger. Hyperledger Indy. [Online]. Available: https://www.hyperledger.org/use/hyperledger-indy

[18] Q. Stokkink and J. Pouwelse, “Deployment of a blockchain-based self-sovereign identity,” in *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, 2018, pp. 1336–1342.

[19] P. Otte, M. de Vos, and J. Pouwelse, “Trustchain: A sybil-resistant scalable blockchain,” *Future Generation Computer Systems*, vol. 107, pp. 770–780, 2020.

[20] Ethereum decentralized identity. [Online]. Available: https://ethereum.org/en/decentralized-identity/

[21] V. C. Hu, D. R. Kuhn, D. F. Ferraiolo, and J. Voas, “Attribute-based access control,” *Computer*, vol. 48, no. 2, pp. 85–88, 2015.

[22] Tribler. (2021) Ipv8 documentation. [Online]. Available: https://py-ipv8.readthedocs.io/_/downloads/en/latest/pdf/

[23] L. Luo, D. Guo, R. T. B. Ma, O. Rottenstreich, and X. Luo, “Optimizing bloom filter: Challenges, solutions, and comparisons,” *IEEE Communications Surveys & Tutorials*, vol. 21, no. 2, pp. 1912–1949, 2019.

[24] European Commission. cef-ebsi packages. [Online]. Available: https://www.npmjs.com/search?q=cef-ebsi

[25] Android. Access app-specific files. [Online]. Available: https://developer.android.com/training/data-storage/app-specific

[26] J. Bambacht and J. Pouwelse, “Web3: A decentralized societal infrastructure for identity, trust, money, and data,” 2022. [Online]. Available: https://arxiv.org/abs/2203.00398

[27] E. Mansour, A. V. Sambra, S. Hawke, M. Zereba, S. Capadisli, A. Ghanem, A. Aboulnaga, and T. Berners-Lee, “A demonstration of the solid platform for social web applications,” in *Proceedings of the 25th International Conference Companion on World Wide Web*, ser. WWW ’16 Companion. Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee, 2016, p. 223–226. [Online]. Available: https://doi.org/10.1145/2872518.2890529

[28] P. Mainini and A. Laube-Rosenpflanzer, “Access control in linked data using webid,” *arXiv preprint arXiv:1611.03019*, 2016.

[29] Q. Stokkink, G. Ishmaev, D. Epema, and J. Pouwelse, “A truly self-sovereign identity system,” in *2021 IEEE 46th Conference on Local Computer Networks (LCN)*, 2021, pp. 1–8.

[30] R. Chotkan, J. Decouchant, and J. Pouwelse, “Distributed attestation revocation in self-sovereign identity,” 2022. [Online]. Available: https://arxiv.org/abs/2208.05339

[31] B. Kim, W. Shin, D.-Y. Hwang, and K.-H. Kim, “Attribute-based access control (abac) with decentralized identifier in the blockchain-based energy transaction platform,” in *2021 International Conference on Information Networking (ICOIN)*. IEEE, 2021, pp. 845–848.

[32] S. Rouhani, R. Belchior, R. S. Cruz, and R. Deters, “Distributed attribute-based access control system using permissioned blockchain,” *World Wide Web*, vol. 24, no. 5, pp. 1617–1644, 2021.

[33] Y. Zhu, Y. Qin, Z. Zhou, X. Song, G. Liu, and W. C.-C. Chu, “Digital asset management with distributed permission over blockchain and attribute-based access control,” in *2018 IEEE International Conference on Services Computing (SCC)*, 2018, pp. 193–200.