IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received September 6, 2021, accepted September 24, 2021, date of publication September 28, 2021,
date of current version October 15, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3116095

ATIB: Design and Evaluation of an Architecture for Brokered Self-Sovereign Identity Integration and Trust-Enhancing Attribute Aggregation for Service Provider

ANDREAS GRÜNER, ALEXANDER MÜHLE, AND CHRISTOPH MEINEL
Hasso Plattner Institute (HPI), University of Potsdam, 14482 Potsdam, Germany

Corresponding author: Andreas Grüner (andreas.gruener@hpi.uni-potsdam.de)

ABSTRACT Identity management is a principle component of securing online services. In the advancement of traditional identity management patterns, the identity provider remained a Trusted Third Party (TTP). The service provider and the user need to trust a particular identity provider for correct attributes amongst other demands. This paradigm changed with the invention of blockchain-based Self-Sovereign Identity (SSI) solutions that primarily focus on the users. SSI reduces the functional scope of the identity provider to an attribute provider while enabling attribute aggregation. Besides that, the development of new protocols, disregarding established protocols and a significantly fragmented landscape of SSI solutions pose considerable challenges for an adoption by service providers. We propose an Attribute Trust-enhancing Identity Broker (ATIB) to leverage the potential of SSI for trust-enhancing attribute aggregation. Furthermore, ATIB abstracts from a dedicated SSI solution and offers standard protocols. Therefore, it facilitates the adoption by service providers. Despite the brokered integration approach, we show that ATIB provides a high security posture. Additionally, ATIB does not compromise the ten foundational SSI principles for the users.

INDEX TERMS Attribute aggregation, attribute assurance, digital identity, identity broker, self-sovereign identity, trust model.

I. INTRODUCTION

Online services require identity management to provide personalized functionality for their users. Nowadays, online services, e.g. social networks, online banking, pervade enormous parts of everyday life. Therefore, users interact with identity management systems in general by accessing an online service. The Identity Provider (IdP) is the central component of an identity management system to provide, for instance, enrollment, authentication and authorization functions [1].

In the ongoing development of traditional identity management patterns from isolated to centralized, and to the federated scheme, the IdP remained a TTP [2]. Within the isolated pattern, the IdP was specific to a service or a Service Provider (SP). A centralized IdP may serve several services or SPs [3]. Additionally, the IdP might be an actor outside the organizational trust boundaries of a SP. In the federated scheme, a number of IdPs, dedicated to distinct organizations, are associated and the organizations mutually accept their identities [3].

In all traditional identity management models, the user needs to trust the IdP with regards to different criteria [2]. A user holds a credential, such as a password or a private key, to control its digital identity. The IdP stores verification information to check the credential upon authentication. The user trusts the IdP to keep its verification information secure. Furthermore, the IdP ensures the privacy of the user's information and the activity of the corresponding identity. Usage statistics about digital identities at SPs is valuable data, for instance, to track actions and unveil the consumption of specific services.

Furthermore, the IdP has the obligation to provide properly verified attributes of an identity [4]. In case attributes are wrong, the service provisioning will fail. Thus, the SP or the user is exposed to a negative impact. In addition to that, the IdP is a profitable target for attackers. The higher the

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

number of enrolled users at an IdP, the more information or credentials can be illegitimately retrieved by an attacker. An adversary has also a high interest in personal data in form of the user's attributes. Besides the risk of an intrusion, the IdP itself may exert illegitimate control about an identity. Unauthorized actions can reach from denying access to the digital identity to contradicting behaviour against contractual agreements. The IdP could arbitrarily deny service to specific users or SPs. In conclusion, the IdP is a powerful TTP that implies several drawbacks for the user and the SP.

The new Self-Sovereign Identity (SSI) paradigm tries to address these disadvantages to bring the user back in control of their digital identity. With the additional rise of blockchain technology [5], the SSI pattern gets a viable implementation option. A blockchain network provides a decentralized execution platform that does not require trust between the peers. A decentralized IdP can be implemented that is no longer a TTP. Thus, the user has full control abouts its identity. This development reduces the role of the former TTP IdP to a sole Attribute Provider (AP) [4]. Furthermore, trust requirements in this pattern are significantly different compared to the traditional schemes [4].

As the emergent SSI pattern attracts eminent interest, a multitude of projects that implement blockchain-based SSI solutions emerged [6]. However, these solutions focus on the user as indicated by the SSI paradigm. Nonetheless, the projects largely disregard the requirements of the SP. Thus, mutual adoption is impeded. Each SSI solution offers a dedicated integration library instead of implementing established protocols. A SP would need to integrate to each SSI solution by changing its applications. Additionally, an identity of a SSI solution is only practically usable if attributes are available. Within the SSI pattern, the SP must be enabled to easily issue verifiable claims [7] to different solutions.

To foster the general adoption of SSI solutions, requirements on the side of the SP demand attention. Our contribution, presented in this paper, comprises the design and evaluation of an architecture for brokered SSI integration and trust-enhancing attribute aggregation. This Attribute Trust-enhancing Identity Broker (ATIB) abstracts from a single SSI solution. Thereby, ATIB offers a generic integration with established identity and access management protocols for applications. The integration enables authentication and the issuance of attributes as SSI-related verifiable claims.

Furthermore, ATIB enables the configuration of dedicated trust modules to define the acceptance of attributes. A trust module implements a trust model. The trust model determines the trustworthiness in a single AP or in combinations of APs. An attribute is considered trustworthy based on the AP or combinations of APs that have issued the respective characteristics. By using ATIB, the principles of the SSI paradigm are not affected for the users.

The remainder of the paper is structured as follows. After introducing the topic in Section I, we provide an overview of related work in Section II and present further background on SSI principles, blockchain, trust requirements and protocols in Section III. Subsequently, we elaborate on main SSI challenges for SPs in Section IV. In Section V, requirements for our architecture are outlined. We depict the architecture of our main contribution, ATIB, in Section VI and describe the implementation in Section VII. Furthermore, we evaluate ATIB in Section VIII. Afterwards, we discuss our work in Section IX. Finally, we conclude the paper in Section X and provide perspectives on future work in Section XI.

II. RELATED WORK

Related research work exists in the domain of attribute aggregation and brokered integration of SSI solutions.

Attribute aggregation patterns focus on the combination of different properties from distinct APs [8]. The main rationale behind the aggregation is that a single AP cannot deliver all required attributes. Therefore, several providers are needed. Ferdous and Poet [9] provide a classification scheme to categorize different types of attribute aggregation models. The main distinctive feature is the location where the aggregation occurs: at the SP, at the IdP or at the side of the user. Chadwick *et al.* [10] outlines the concept of a Linking Service that is under control of the user. The Linking Service has access to the credentials of all identities of a user. Upon authentication at a service, the Linking Service authenticates at all IdPs, retrieves the required attributes and provides them together to the SP. The origin of the attribute is retained.

Furthermore, Chadwick and Inman [11] have implemented a Trusted Attribute Aggregation Service (TAAS). TAAS is an additional trusted third party that controls the communication flow between the user, the SP and the IdPs. Subsequent research has concentrated on hybrid models [12], privacy promoting techniques and other dedicated usage scenarios [13], [14] [15]. In contrast to our work, different attributes from different APs are solely combined. ATIB enables the combination of the same attribute from distinct APs to increase trust in the property's correctness.

The research area of the mediated integration of SSI solutions targets the brokered usage of SSI with applications. A broker avoids the direct integration and enables the easy change of an SSI solutions or the transparent usage of multiple solutions. Hyperledger (HL) Aries [16] is a blockchain client for identity management that is specifically built for HL Indy [17]. HL Indy is a dedicated set of blockchains for identity management. HL Aries provides an high-level interface for applications to interact with identity management functions. HL Aries has the vision to support a variety of SSI solutions. However, it is currently focused on HL Indy, based on its origins as a client component of HL Indy. Furthermore, traditional identity and access management protocols are not supported.

Besides HL Aries, the Universal Resolver [18] is another brokered integration concept. An identity in the SSI paradigm can be addressed by its Decentralized Identifier (DID) [19]. Within the DID, the applied SSI solution is encoded. The Universal Resolver gets as input a DID and resolves it to a DID document [19]. The document can contain keys, protocols and further endpoints. A drawback is the manual input of the DID by the user. Moreover, the authentication process requires several steps to take place before the actual authentication can occur. In contrast to our work, there is no option to apply trust models for specifying trustworthiness in certain APs.

III. BACKGROUND

In the following subsections, we describe the background on SSI principles, blockchain-based implementations and trust requirements in the domain of SSI. Furthermore, we briefly introduce popular identity and access management protocols.

A. SSI PRINCIPLES

The SSI paradigm focuses primarily on the user including the objective of bringing the control of the digital identity and its data back to the user. Allen [20] coined the SSI paradigm by ten principles. The principles are categorized in Table 1.

TABLE 1. Allen's SSI principles categorized by the sovrin foundation [21].

Security	Controllability	Portability
Protection	Existence	Interoperability
Persistance	Control	Transparency
Minimization	Consent	Access

- **Existence:** The identity reflects a human user. The user is able to access digital services with support of the identity.
- **Control:** The user exerts the definite control about its digital identity and attributes. This characteristic differentiates SSI from traditional models where the ultimate control resides with the IdP (cf. Section I).
- **Access:** The user is always able to access the associated data of the identity. Especially, the user is fully aware of associated verifiable claims.
- **Transparency:** Applications that support the user to manage its identity must be transparent in composition and management.
- **Persistence:** The identity of a user should be enduring, and lasts as long as the user wishes it.
- **Portability:** The user should be able to transfer its identity from one provider to another. There should be no lock-in to a single TTP.
- **Interoperability:** The identity of a user should be as practicable as possible. This implies a widespread usage at many SPs.
- **Consent:** Usage of the identity and unveiling of attributes must only be allowed with the consent of the user.
- **Minimalization:** During the usage of the identity, especially when disclosing attributes, only a minimum amount of data must be disclosed to third parties. The principle of data economy should be adhered to.
- **Protection:** The axiom of protection implies the precedence of user rights. In case of a conflict between the identity holder and the network, the decision should be in favor of the identity holder.

In our opinion, the usage of SSI solutions at the side of the SP should not compromise these principles. Thus, the SSI paradigm is not undermined.

B. BLOCKCHAIN-BASED SSI

In 2008, Nakamoto [22] published the foundations of Bitcoin. Bitcoin is a decentralized digital cash scheme that applies blockchain technology. Thus, it solves the double-spend problem [23] in a decentralized manner instead of relying on a TTP. The advancement of the emergent blockchain technology leads to the development of a generic decentralized execution platform without a need for a TTP. Ethereum [24] is an example of a public unpermissioned blockchain whereas HL Fabric [25] is a private and permissioned blockchain.

In the traditional identity management patterns, the IdP is a strong TTP with drawbacks (cf. Section I). In particular, a central IdP is a single point of failure and control. Blockchain technology enables the implementation of a decentralized IdP that realizes the SSI paradigm. A blockchain uses private/ public-key cryptography to authenticate participants and sign messages that are sent to the network. Furthermore, the private key can be applied as the user's credential for controlling the identity. The blockchain network is able to verify the relation to the associated public key transparently. Thus, a self-authenticating scheme is established [26].

Moreover, a decentralized identifier registry ensures the uniqueness of the identity's identifier across the namespace [26]. Therefore, a central authority is not required to guarantee individuality. An extended claim registry provides a verifiable proof of existence and revocation of an attribute [26]. A verifiable claim [27] represents an attribute. It consists of a claim and an attestation. The claim is the actual attribute. An attestation is the confirmation by an entity that the attribute is valid. The relation between these components are shown in Fig. 2. Notably, a claim can be attested by several attestation issuers.

Moreover, Fig. 1 shows an overview of the decentralized IdP, the actors and their interaction paths. The user is the owner of an identity and uses an identity wallet for interaction

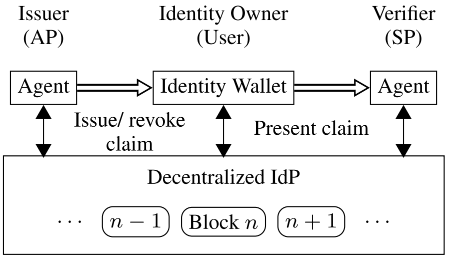


FIGURE 1. SSI actors and interaction.

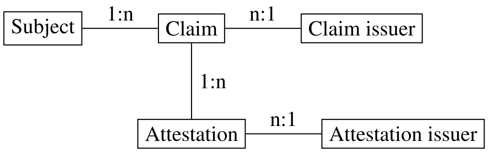


FIGURE 2. Relations of a verifiable claim.

with the decentralized IdP. The AP issues verifiable claims to the user and the SP verifies the presented claims. Both AP and SP communicate with the decentralized IdP by using an agent.

Implementations of blockchain-based SSI solutions are driven by a diverse range of projects. The SSI solutions uPort [28] and Jolocom [29] consist of smart contracts on the Ethereum blockchain. HL Indy builds a dedicated set of blockchains for identity management. Additionally, the private centralized implementations ShoCard [30] and Blockchain Helix [31] are only a few of the many implementations [6].

C. SSI TRUST DIFFERENCES

Trust is a social phenomenon that occurs among entities [32]. It plays a vital role in relationships that are analysed in various disciplines, including computer science. In our view, the most applicable denotation is Jøsang *et al.*'s [33] definition of decision trust. The notion delineates trust as *"the extent to which one party is willing to depend on something or somebody in a given situation with a feeling of relative security, even though negative consequences are possible."* The depending parties are the user, the SP and the IdP. Trust requirements are the dependencies between these actors. The demands encompass for instance credential management and attribute assurance.

The development of identity management paradigms, from isolated to centralized and federated, lead to a general increase of trust requirements between the user, the IdP and the SP [4]. In particular, the IdP requires the most trust from the other parties. By implication, the user and the SP are highly dependent on the IdP. Malfunctioning of the IdP will lead to an adverse impact for the user and SP. Blockchain-based SSI is not only centered around the needs of the user, but also decreases the trust requirements for the SP [4].

There are two major domains where trust reduction can occur [4]. First, trust in authentication and credential management is completely eliminated due to the transparent implementation of blockchain and the use of a self-authenticating scheme. Despite that, the trust demand in attribute management can be decreased by the usage of several APs. The combination of the same attribute from several APs by an aggregation model can lead to an overall increased trustworthiness in the attribute [34]. For instance, the SP might not accept an attribute from one medium trusted AP. However, the property might get accepted by the SP if two medium-trusted AP deliver it. The model ideally facilitates the trust-enhancing attribute aggregation because blockchain-based SSI decouples the identifier from the attributes of an identity.

D. PROTOCOLS

Identity and access management protocols have been established to ensure interoperability between IdPs and SPs and to foster the exchangeability of IdPs. Widespread protocols to facilitate authentication and authorization are OpenID Connect (OIDC) [35] and Security Assertion Markup Language version 2.0 (SAML2) [36].

OIDC outlines different authorization grants to obtain the ID token. During the authorization code flow, the user is redirected to the OIDC provider. After successful user authentication, the ID token is transmitted to the backend of the SP application. Within the implicit flow, the ID token is returned to the frontend of the application when redirecting the user back to the SP. This flow is intended for applications that do not have a backend. In the hybrid flow, the ID token is returned to both ends. User attributes are transmitted within the ID token direct after authentication. Additionally, user attributes can be requested at the token endpoint. For instance, OIDC is used as authentication protocol for social logins like Facebook[1] or Google.[2]

SAML2 provides a XML-based standard to securely exchange messages for authentication and authorization between the IdP and the SP. The authentication assertion reflects the result of the authentication process at the IdP. Properties of the user are transferred by the attribute assertion. OIDC and SAML2 do not specify a dedicated authentication method. Therefore, user authentication and attribute retrieval can happen via a SSI solution.

Besides these general protocols, specific SSI-related communication agreements are developed to provide standards within the SSI ecosystem. DIDAuth [37] defines the authentication flow with a SSI solution. DIDComm [38] outlines general communication between agents. However, these protocols are not yet mature enough and have a limited adoption compared to the adoption of OIDC and SAML2.

IV. SSI CHALLENGES FOR SP

As previously explained, the concept of SSI is aligned towards the user. The user should have the full control of its digital identity and related information. Based on the usage of the emerging SSI solutions, we illustrate four main challenges that SPs face.

A. MULTITUDE OF SSI SOLUTIONS

The SSI paradigm raised along with the hype about blockchain technology and related initial coin offerings. A countless number of solutions and approaches have been proposed by academia and industry [6]. These projects vary significantly in their approaches: usage of permissioned or unpermissioned blockchains, implementation as a smart contract or dedicated blockchain, restricted to a consortium or

[1] https://developers.facebook.com/docs/facebook-login/
[2] https://developers.google.com/identity/sign-in/web/sign-in

public to everybody. Additionally, a SP may directly integrate with the solution, or via a trusted steward as intermediary. Furthermore, software libraries to connect a SSI framework to applications differ in the applied programming language. A SP may need to spend a significant amount of effort to integrate to the majority of solutions or solely supports a minor proportion. Besides that, if users cannot login with SSI solutions at applications of the SPs, the adoption at the end of the user is also prevented. Overall, based on the fragmented SSI solution landscape, the development of the complete SSI ecosystem is impeded.

1) CHALLENGE

How can a SP be enabled to easily integrate with a plenitude of technologically different SSI solutions while not creating a considerable dependency on a single implementation?

B. DIVERGENT TRUST IN ATTRIBUTE PROVIDERS

In traditional identity management patterns attributes and the identifier of a digital identity are strictly connected and associated to a particular IdP. A SP may integrate one or more specific IdPs in their application portfolio and offer them to the user. Therefore, the user has a limited decision to enroll at these IdPs. In addition to that, the user may not use the service at all to refrain from enrollment. On the other hand, the SP is likely to offer an IdP that enables a large user base to consume its service. Thus, there is a strong mutual dependency that does not satisfy the needs of both sides. In the worst case, the SP and the user need to communicate with an IdP that is barely trusted by the two of them. A higher flexibility in the IdP and AP selection process supports the user and the SP in their trust preferences.

1) CHALLENGE

How can the SP and the user be enabled to integrate respectively register with their preferred IdPs or APs while not forcing a consistent choice?

C. EXISTING APPLICATION LANDSCAPE

A SP may use various applications to implement their business processes and offer service to their clients. Large organizations have accumulated a complex existing application landscape by undergoing several cycles of change in technology generations. Adapting this landscape to new requirements or vastly transforming highly depending components demands significant investment. The integration of SSI solutions as new modules into an established landscape of applications should demand less effort as possible. Thus, SP adoption is facilitated. In general, the use of standard protocols enables independence from specific solutions

1) CHALLENGE

How can a SP integrate SSI solutions without significantly re-engineering existing applications?

D. ATTRIBUTES BASED ON VERIFIABLE CLAIMS

The IdP retrieves user information during the enrollment process and makes it subsequently available to the SP as attributes of the user's digital identity. Verification of attributes is required for usage scenarios with high demands on correctness, for instance, in high risk scenarios. In the SSI ecosystem, attributes of an identity are verifiable claims. These verifiable claims can be issued by APs. However, further entities, including SPs, are able to issue verifiable claims based on the availability of proved information about the user. A SP may issue claims about memberships within subscription models. Having available data does not mean being in the position to technically issue verifiable claims. Furthermore, if a user newly creates an identity within a SSI solution, there are no attributes. The user is able to create self-attested claims. However, these claims are hardly trusted by any SP and only usable in non-risk scenarios. A self-attested claim regarding a firstname can be used in a welcome message of an web application without any risk for the SP. To overcome the chicken-and-egg problem when creating a new identity and to foster verifiable claim issuance, a dedicated facility that is highly integrated in identity management processes of the SP is required.

1) CHALLENGE

How can the SP be enabled to easily issue verifiable claims to the user?

V. REQUIREMENTS

Based on the previously described challenges and the background in the domain of SSI, we formulate the following Requirements (R). Our ATIB architecture should fulfill these requirements to support the SP for SSI adoption.

- **R1 Authentication:** We need to support authentication with SSI solutions for SP applications. A user can login with their favorite SSI solution and its identity wallet.
- **R2 Authorization:** After authentication, the application should be able to make authorization decisions based on attributes of the user. The user is able to provide its verifiable claims to the application.
- **R3 Verifiable Claim Issuance:** The SP should be enabled to easily issue verifiable claims to the user's SSI solution.
- **R4 SSI Independence:** Authentication, authorization and verifiable claim issuance functionalities should be independent from the SSI solution of the user. The SP does not need to integrate each single SSI solution for the mentioned purposes.
- **R5 Flexible Attribute Trust:** The user and the SP should be enabled to individually decide on their trusted APs without being forced to make a congruent choice.
- **R6 Application Technology Autonomy:** The implementation of our architecture should be as independent as possible of the technology stack of SP applications. This provides the maximum benefit within a heterogeneous application landscape.
- **R7 Non-impairment of SSI Principles:** Our architecture should not have a negative impact on the SSI principles for the user. The architecture should foster

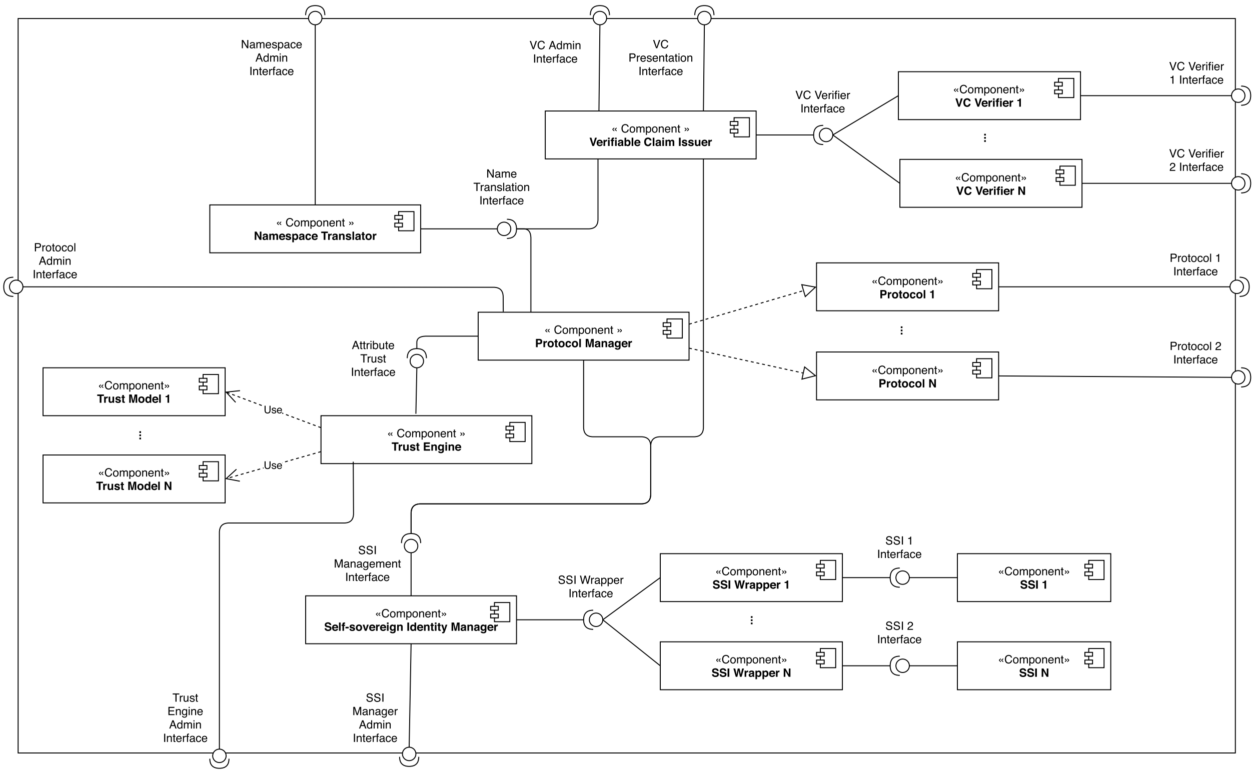

Namespace Admin Interface
VC Admin Interface
VC Presentation Interface
VC Verifier Interface
«Component» VC Verifier 1
VC Verifier 1 Interface
« Component » Verifiable Claim Issuer
«Component» VC Verifier N
VC Verifier 2 Interface
Name Translation Interface
« Component » Namespace Translator
Protocol Admin Interface
«Component» Protocol 1
Protocol 1 Interface
« Component » Protocol Manager
Attribute Trust Interface
«Component» Protocol N
Protocol 2 Interface
«Component» Trust Model 1
Use
« Component » Trust Engine
«Component» Trust Model N
Use
SSI Management Interface
SSI Wrapper Interface
«Component» SSI Wrapper 1
SSI 1 Interface
«Component» SSI 1
«Component» Self-sovereign Identity Manager
«Component» SSI Wrapper N
SSI 2 Interface
«Component» SSI N
Trust Engine Admin Interface
SSI Manager Admin Interface

FIGURE 3. Component view.

SP adoption by keeping the user in control about its identity and attribute data.

- **R8 Security:** ATIB should adhere to security best practices for instance to exchange identity and attribute assertion securely with the application and to request this information securely from a SSI solution. A brokered integration of a SSI solution should be at least as secure as a direct integration.

VI. ARCHITECTURE

In this section, we describe the architecture of ATIB and highlight deployment options. Additionally, we demonstrate the fulfilment of the requirements and the solution of the SSI challenges for the SP.

A. ATIB ARCHITECTURE

ATIB is comprised of several components and interfaces. The interfaces connect components within ATIB and enable the communication with the surrounding environment. In the following subsections, we outline the overall concept, the components and internal as well as external interfaces. Fig. 3. shows an overview of ATIB.

1) GENERAL CONCEPT

We introduced the initial concept for ATIB, Attribute Trust-enhancing Identity Broker, in [39]. ATIB is technically an IdP that provides traditional identity and access management protocols as communication endpoints for web applications. However, behind ATIB there is no user store with attributes and credential information for authentication and identity profile delivery. When an application requests user authentication at ATIB, the authentication is redirected to the respective SSI solution. The user logs in with its SSI identity. Subsequently, the identifier is conveyed to the application. Furthermore, requested user attributes are routed back to the application in the protocol flow. The user attributes are derived from the verifiable claims.

2) COMPONENTS

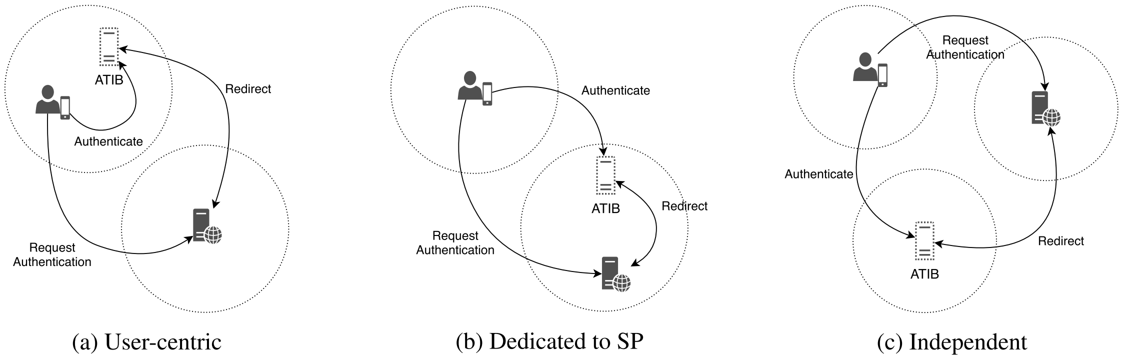
ATIB is comprised of the components Namespace Translator, Trust Engine, Protocol Manager, Self-Sovereign Identity Manager and the Verifiable Claim Issuer.

a: NAMESPACE TRANSLATOR

The Namespace Translator conveys the identifiers of claims between different namespaces. Usually, the same attribute is differently identified between distinct protocols, technologies or domains. This component achieves interoperability for claim names. The internal Name Translation Interface is used by other components to retrieve the correct value for their processing context.

b: TRUST ENGINE

The Trust Engine is the component to evaluate trust in the verifiable claims of an identity. The Trust Engine makes use of



(a) User-centric (b) Dedicated to SP (c) Independent

FIGURE 4. **Deployment patterns.**

one or more different trust models to evaluate the claims and make a decision. A trust model defines the trustworthiness of APs and combinations of them. By receiving attestations of a claim, the trust engine returns the claim as attribute if the evaluation was successful. The used trust model reflects the subjective opinion of the ATIB host about the trustworthiness of APs. It basically defines the trust anchors for the usage of ATIB. The Protocol Manager uses the functionality of the Trust Engine via the Attribute Trust Interface.

c: PROTOCOL MANAGER

The Protocol Manager is the core component of ATIB. This element can implement various standard identity and access management protocols for interaction with other applications. Furthermore, it manages the ATIB internal processing with the support of the other components. The Protocol Manager calls the Namespace Translator, the Self-Sovereign Identity Manager and the Trust Engine to drive user authentication. The Protocol Manager retrieves the required attributes from the SSI solution as verifiable claims. After trust evaluation, the characteristics are used in the protocol flow as user attributes.

d: SELF-SOVEREIGN IDENTITY MANAGER

The Self-Sovereign Identity Manager controls the communication towards the various SSI solutions by using a generic SSI Wrapper Interface. For each SSI solution a wrapper around the corresponding application programming interface exists. This interface exists to abstract from SSI specific libraries. All wrappers implement the wrapper interface that is called by the manager itself. The SSI Wrapper Interface supports the following functions.

- **Create Identity:** This function creates a new digital identity in a SSI solution. For each supported SSI solution, ATIB requires an individual identity to issue claims and to serve as an endpoint for the communication. Additionally, the identity is shown for consent when requesting the authentication or verifiable claims.
- **Create Challenge:** The Create Challenge function generates an authentication challenge for the SSI solution. The user can respond to the authentication challenge with its identity wallet. The authentication challenge may already comprise required attributes for the application.
- **Verify Challenge:** When ATIB receives a message of the user in response to the authentication challenge. This function verifies the message, for instance the signatures of the sender.
- **Request Verifiable Claim:** In case the authentication challenge does not contain the required attributes, the function Request Verifiable Claim enables requesting user attributes and validation information.
- **Verify Verifiable Claim:** ATIB verifies verifiable claims that are received as response. This function enables the verification with the SSI network.
- **Create Verifiable Claim:** The Create Verifiable Claim function enables ATIB to create a verifiable claim for a specific attribute with a certain value.

e: VERIFIABLE CLAIM ISSUER

The Verifiable Claim Issuer component manages the issuance of verifiable claims to the user and its SSI solution. Furthermore, the element executes verification processes to check the required data for the property of the user.

3) EXTERNAL INTERFACES

ATIB provides several interfaces for administration and communication with the surrounding environment. In the following subsections, we describe the Admin, VC Issuer Presentation, VC Verifier and Protocol Interfaces.

a: ADMIN INTERFACES

The various administration interfaces, as shown in Fig. 3, enable the configuration of ATIB and its components. For instance, the management of the supported SSI solutions or the available VC Issuers are possible.

b: VC ISSUER PRESENTATION INTERFACE

The presentation interface visualizes the verification process for and the retrieval of verifiable claims. The user requires a possibility to obtain the claim after verification. Additionally, the user must be guided through the verification process and may provide data for it.

TABLE 2. **Advantages and disadvantages of deployment patterns.**

Pattern	Advantages	Disadvantages
User-centric	Alignment to SSI principles	Trust model may not satisfy service requirements SP integration to all users is impractical
Dedicated to SP	Trust model satisfies service requirements	
Independent	Minimal effort for user and SP	Additional TTP established Reduced flexibility of AP selection

c: VC VERIFIER INTERFACES

The VC Verifier Interfaces provide communication means with surrounding systems to retrieve data or query systems for data verification. Data verification is required to build a foundation for the issuance of verifiable claims.

d: PROTOCOL INTERFACES

The various Protocol Interfaces provide the communication endpoint for applications for identity and access management. It is the interface to provide identity and attribute assertions to be used at the applications and make authorization decisions.

B. DEPLOYMENT PATTERNS

ATIB can be deployed in three major options with different impact on the organizational trust boundaries that separates the user and the SP. In the following subsection, we describe the patterns User-centric, Dedicated to SP and Independent. Fig. 4 shows the graphical representations. The dashed circles in the diagrams outline the trust boundaries of the actors.

1) USER CENTRIC

The user can deploy an instance of ATIB themselves. This instance runs within the trust boundary of the user and outside the trust domain of the SP. Therefore, it is specific to the user. The implemented trust model in ATIB reflects the opinion of the user with regard to the trustworthiness of applied APs.

It is favorable for the user and aligned to the SSI paradigm. However, the user already holds only verifiable claims from trusted APs. Thus, the user can already enforce its own vision about trusted APs and does not require ATIB in its own trust boundary.

Furthermore, a user-specific trust model unlikely satisfies the trust requirements for the services that are offered by a SP. Therefore, the SP may not trust the ATIB interaction and rejects integration of any kind. Additionally, an integration to a countless number of users by a SP, even if it is only a configuration, may not be practical.

2) DEDICATED TO SP

A SP can host an instance of ATIB. This instance can be integrated in all services for authentication and attribute-based authorization. Thus, ATIB runs in the organizational trust boundary of the SP. The implemented trust model reflects the opinion of the SP towards various APs. The acceptance of attributes are aligned to the criticality of the services that are connected to ATIB. The user can directly interact with ATIB for authentication and authorization. The process is successful in case that the user supplies verifiable claims from APs that match the trust model of the SP.

3) INDEPENDENT

Besides the previously described options, an independent entity can host an instance of ATIB. The host can be seen as an IdP that is a TTP for SPs and users. The IdP creates a new organizational trust boundary in addition to the user and SP. Furthermore, the IdP implements an attribute trust model with an independent opinion. The AP selection of the IdP reduces the flexibility to choose APs for the user and SP. The independent model easily provides functions, but counteracts the achievement of the SSI paradigm. The decentralized IdP is reinstated as TTP.

4) CONCLUSION

We presented three deployment patterns for ATIB in the previous subsections. An overview about advantages and disadvantages is shown in Table 2. The user-centric deployment options does not seem to be a feasible approach. Every user would require its own ATIB instance that demands integration to SPs. However, the SPs may not trust the respective AP selection. Besides that, the independent deployment pattern introduces an additional TTP. This counteracts the SSI paradigm that eliminates the IdP.

In our opinion, the approach that implements a dedicated ATIB instance in the organizational trust boundary of the SP is most applicable. In the following sections, we assume this deployment pattern.

C. REQUIREMENT COVERAGE

Having defined the architecture of ATIB, we can show the fulfilment of requirements **R1** to **R6**. Authentication (**R1**) and authorization (**R2**) are supported by the Protocol Manager component. Depending on the implemented identity and access management protocols, applications can delegate authentication and authorization to ATIB. ATIB utilizes connected SSI solutions to fulfil these functions. Required attributes of the user are requested from the SSI solution. Furthermore, the Protocol Manager achieves compliance to requirement **R6**, application technology autonomy, as well. The usage of standard communication protocols for identity and access management decouples from the actual application technology and enables cross-technology integration.

The Verifiable Claim Issuer component addresses requirement **R3** to easily issue claims to the user after required data validation has been executed. Complying to requirement **R4**, the stated functionality must be independent from a specific SSI solution. In general, this independence is achieved by the component-based architecture of ATIB. Specifically, the Self-Sovereign Identity Manager abstracts from the used SSI solution. In addition to that, only a very lightweight wrapper is a SSI solution specific element. However, the wrapper implements towards the manager component a standard interface to keep the SSI specific implementation to a minimum. Flexible attribute trust, as described in requirement **R5**, is realized by the Trust Engine with the use of flexible trust models. The trust model enables a variable composition of trusted APs or groups of APs for a specific property. As a result, the user has more freedom in chosing APs for its characteristics.

Coverage of requirements about non-violation of the SSI principles **R7** and the overall security **R8** are described in Section VIII.

VII. IMPLEMENTATION

In this section, we outline the implementation of ATIB to demonstrate the feasibility of the concept. We start with an overview of the general implementation in the first subsection and subsequently provide the covered functionality of each component.

A. TECHNICAL ARCHITECTURE

Fig. 5 depicts the technical architecture of our ATIB implementation. We use a virtual machine based on the operating system Ubuntu 18.04 as ATIB server that hosts all related components. The core ATIB application is implemented in the Python programming language by using the Tornado [40] web application framework. Configuration information for ATIB is stored locally in configuration files. Additional information is persisted in a PostgreSQL database that is also hosted on the virtual server. For end user communication, especially to issue verifiable claims, the ATIB User Interface exists. It is connected to the ATIB core application based on the web service paradigm. Furthermore, SSI wrapper for interaction with the Self-Sovereign Identity Manager component are implemented as web service, too. Wrapper exists for uPort, Jolocom and HL Aries.

Despite that, ATIB interacts with a mail server and a directory service for verification of user attributes. The protocols OIDC and SAML2 are offered for authentication and authorization to surrounding web applications.

B. IMPLEMENTED COMPONENTS

In Section 3, we have described the components of ATIB in general. Within this subsection, we present the implemented functionality of each component.

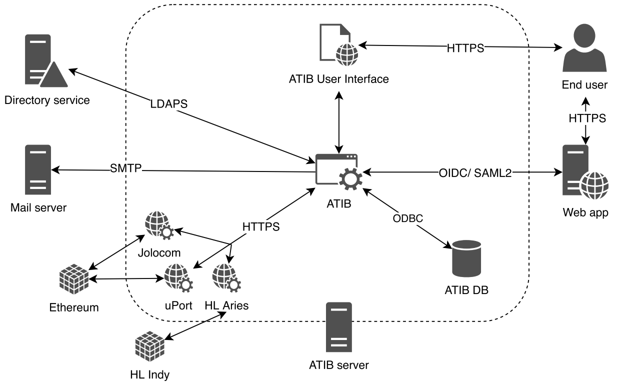


FIGURE 5. Technical architecture.

1) NAMESPACE TRANSLATOR

Our ATIB implementation is able to translate the identifiers for email address, name, first name and last name. An overview of the different denotations is shown in Table 3. uPort as well as OIDC labels the email address as email whereas Jolocom uses the term ProofOfEmailCredential. In contrast, HL Indy and SAML2 does not specify any predefined claim names. However, organizations and other entities can publish schema definitions and derive thereof credential definitions. Within the schema, the claim name can be chosen by the creator. Name translation is required to facilitate more complex trust modelling of APs and the issuance of verifiable claims independently from the SSI solution.

TABLE 3. Verifiable claim names in distinct domains.

Claim	uPort	Jolocom	OIDC
Email	email	ProofOfEmailCredential	email
Name	name	ProofOfNameCredential	name
Firstname	firstname	ProofOfFirstnameCredential	given_name
Lastname	lastname	ProofOfLastnameCredential	family_name

2) TRUST ENGINE

We have implemented two trust modules in the Trust Engine for determining the trustworthiness of verifiable claims. The simple trust module reflects a trivial trust opinion towards APs. The module only accepts verifiable claims as trustworthy if they are issued by the identity of ATIB itself. ATIB owns an identity, in the form of a DID, for each SSI solution that is supported. The owned identity is used for issuing verifiable claims. The trust understanding is aligned to the isolated and centralized paradigm. The IdP itself verifies and issues the user's attributes that are later conveyed to the SP as attribute assertions.

Besides the simple trust module, we have implemented a generic trust-enhancing module based on [34]. Trust in an AP is specified by a probability value for correctness and validity of an attribute. Combined with a dependency factor that is specific to an AP, the overall probability $\mathcal{P}$ for an attribute is calculated. The dependency factor expresses the strength of the relationship between correctness and validity at a certain AP. By considering several APs for an attribute, the joint probability is computed. We use the joint probability as trust function and refer to it with Θ.

Furthermore, the trust module applies acceptance rules comprising each attribute to determine a threshold for considering the property as valid.

Definition 1 (Acceptance Rules): Let $\mathbb{S}$ bet a set of acceptance rules to decide at a threshold $t \in (0 \ldots 1)$ on the use of an attribute $a \in \mathbb{A}$ under n attestations of distinct providers $p_1 \ldots p_n \in \mathbb{P}$. An element $s_i \in \mathbb{S}$ is defined as follows.

$$s_i : \Theta(\mathcal{P}_{p_1}, \ldots, \mathcal{P}_{p_n}) \geq t_{a_i} \Rightarrow a_i \tag{1}$$

In general, if the overall probability for an attribute exceeds a threshold, the property is accepted from the trust engine. The threshold reflects a risk indicator for the SP. The higher the threshold is set, the higher the assurance that the attribute is correct and valid. In the ATIB database, the considered APs, their DIDs as reference, and the respective probability values as well as the dependency factor are stored as a configuration.

3) PROTOCOL MANAGER

The Protocol Manager of our ATIB implementation covers the OIDC and the SAML2 protocol. The OIDC implementation is based on the Python library pyoidc [41] and the SAML2 protocol relies on the pysaml2 [42] library. For OIDC the following endpoints are implemented.

- **Authorization Endpoint:** The relative url from ATIB is */oidc/authorization*. A SP web application that requires authentication redirects the user to this endpoint. The implemented authentication method "blockchain" controls the SSI-based authentication flow.
- **Token Endpoint:** The token endpoint is accessed via the relative url */oidc/token*. The web application can retrieve at this endpoint an access or ID token of the user. The ID token already contains attributes of the user.
- **UserInfo Endpoint:** The userinfo endpoint is available under */oidc/userinfo*. At this endpoint further user attributes can be retrieved. The Protocol Manager will retrieve the respective verifiable claims from the SSI solution or conveys attributes that have been retrieved during the authentication process.
- **Further Endpoints:** Additional default endpoints of the OIDC protocol exist for session termination or automated application registration.

For SAML2, the ATIB implementation covers the following endpoints:

- **Single-Sign On Service:** The service is reachable on the relative ATIB url */saml/sso*. All specified bindings of the standard are supported. A web application redirects the user to this service for authentication. ATIB executes the SSI authentication and provides a response including attributes back to the SP application.
- **Single-Logout Service:** The single logout service ends a user session upon request by a SP. This service is reachable under the relative ATIB url */saml/slo*.

4) SELF-SOVEREIGN IDENTITY MANAGER

The Self-Sovereign Identity Manager including the associated SSI Wrappers implement the usage of uPort, Jolocom and HL Aries. The wrapper services for the SSI solutions moderate the communication. Fig. 6 presents the ATIB generic wrapper function to create a new authentication challenge for uPort. The signature contains a callback address for the authentication response and the required verifiable claims. Additionally, the function distinguishes self-attested properties and attributes that are attested by other parties.

```
def createChallenge(self, callback,
 claims, claims_verified):
  result = self.executeWSCall(
      'createchallenge',
      appname=self.app,
      did=self.appid, privatekey=self.key,
      claims=json.dumps(claims),
      claims_verified=
          json.dumps(claims_verified),
      callback=callback)
  try:
    return json.loads(result)['jwt']
  except Exception as e:
    log.exception("uPort WS Call failed")
```

FIGURE 6. Generic wrapper create challenge call.

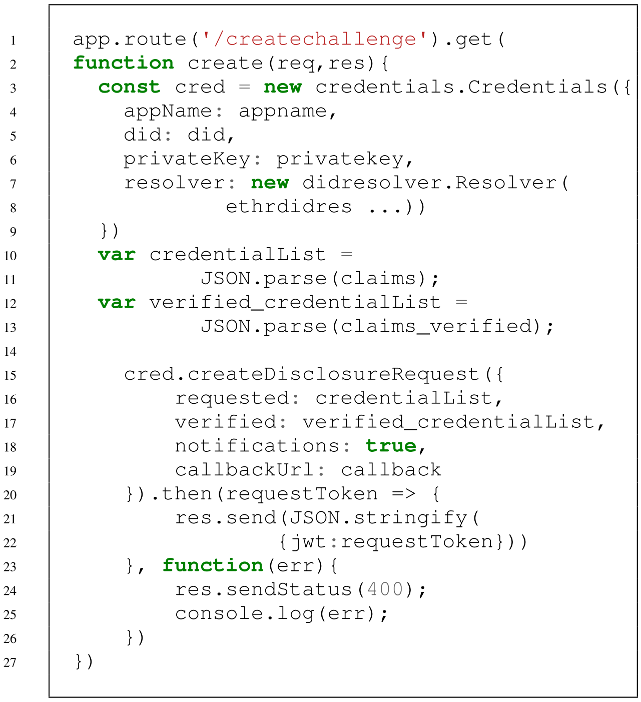
The wrapper for uPort and Jolocom are implemented in Node.js based on the respective libraries. As uPort and Jolocom is based on Ethereum, we connect to the test environment Rinkeby [43]. We implemented the wrapper for HL Aries in Python and used the cloud agent [44] for interacting. The HL Aries cloud agent is connected to the Verifiable Organization Network [45], a test network initiated by the Government of British Columbia.

Fig. 7 presents the function to create a new authentication challenge for uPort. The function requires an application name, a DID and the associated private key to create credential object (line 3-9). Additionally, the required claims are wrapped correspondingly in a claim or verified claim list (line 10-13). Subsequently, a disclosure request is created. The result is returned to the Self-Sovereign Identity Manager component.

5) VERIFIABLE CLAIM ISSUER

The Verifiable Claim Issuer component executes data verification procedures and publishes afterwards an attested claim to the user. For each type of verifiable claim several input and verification data sources exist. We have implemented the verification of the email address and the name of a user.

For providing a verifiable claim of an email address, the user needs to authenticate at ATIB. Subsequently, the user manually enters the email address that he/she claims to own



```
app.route('/createchallenge').get(
function create(req,res){
  const cred = new credentials.Credentials({
    appName: appname,
    did: did,
    privateKey: privatekey,
    resolver: new didresolver.Resolver(
            ethrdidres ...))
  })
  var credentialList =
            JSON.parse(claims);
  var verified_credentialList =
            JSON.parse(claims_verified);

    cred.createDisclosureRequest({
        requested: credentialList,
        verified: verified_credentialList,
        notifications: true,
        callbackUrl: callback
    }).then(requestToken => {
        res.send(JSON.stringify(
                {jwt:requestToken}))
    }, function(err){
        res.sendStatus(400);
        console.log(err);
    })
})
```

FIGURE 7. uPort wrapper create challenge function.

and starts the process. ATIB sends a verification email to the specified address. The email contains a link to ATIB with a large random number. If the user really owns the email address, he/she can login to his mailbox and click on the link. If the link is successfully opened, ATIB receives the result that the email address is actually under the control of the user. Afterwards, the user can retrieve the verifiable claim for the applied SSI solution. The claim is issued by the identity of ATIB.

A directory service is a common authentication solutions at companies. Additionally, it stores further user information. Besides the email verification, we developed a module that connects to a directory service. We use this module to issue a verifiable claim about the name of a user. The user provides its distinguished name and the password. The verification module executes a bind against the directory service to determine validity. Additionally, ATIB searches for the value of the displayname attribute, that belongs to the InetOrgPerson class, for the provided user name. Subsequently, the user can retrieve the corresponding claim.

VIII. EVALUATION

We evaluate the architecture concept and the implementation of ATIB in several ways. First, we demonstrate authentication and the attribute retrieval workflow with the sample application tele-TASK[3] of our institute. Subsequently, we show the transformation of attribute requirements for other applications into the trust-enhancing aggregation model of ATIB. Moreover, we present ATIB performance measurements. Furthermore, we evaluate the adherence to SSI principles and the overall security of ATIB. Finally, we outline usage statistics of the publicly available ATIB instance from the years 2019 and 2020.

A. AUTHENTICATION WORKFLOW

We have chosen tele-TASK to demonstrate an end-to-end authentication workflow with a sample application. Tele-TASK is a web application built at our institute that enables users to watch recorded video sessions of lectures and other events. The permission model of tele-TASK is simple and separates users into two categories: internals and externals. An internal user is associated with our institute and can watch all recorded videos. External users can solely watch a restricted set of sessions.

The differentiation criterion is the email address of the user. In case the email address belongs to the domain of our institute, the user is internal. Otherwise, the user is external. Thus, the authorization model is attribute-based. Table 4 provides an overview of the attributes, considered providers and the acceptance rules. ATIB refers to its identity. The provider *anonym* references any attestation issuer. The threshold of the acceptance rule is 1 due to the risk of the attribute for access control.

TABLE 4. Trust model characteristics for tele-TASK.

Attributes	Providers	Acceptance Rules
$\mathbb{A} = \{email\}$	$\mathbb{P} = \{ATIB, anonym\}$	$\mathbb{S} = \{\Theta \geq 1 \Rightarrow email\}$

Furthermore, tele-TASK is build on top of a web framework that already includes easy configuration for the OIDC protocol to authenticate and authorize a user. Therefore, the simple authorization model and the prepared OIDC connectivity makes tele-TASK an ideal candidate for integration with ATIB.

As preparation for a successful authentication process, several requirements need to be fulfilled. Users must have installed the uPort identity wallet on their smartphone and creates a new digital identity. We demonstrate this sample flow with uPort. However, the usage of Jolocom or an identity wallet for HL Aries/ Indy is comparable. Subsequently, a verifiable claim for the email address needs to be obtained by using ATIB. ATIB considers this issued claim as trustworthy. Besides that, the OIDC protocol implementation of ATIB is configured to recognize tele-TASK as integrated application. At ATIB, a client identifier and a client secret as well as redirect urls are configured.

After we complete the prerequisites, the actual authentication process can start. Fig. 8 outlines an overview of the exemplary process. First, the user opens the tele-TASK web application and proceeds to login (1). As a selectable authentication method, SSI via ATIB is offered. The user selects ATIB as the preferred method. Subsequently, tele-TASK redirects the user to ATIB (2). The relative url of the

[3] https://www.tele-task.de

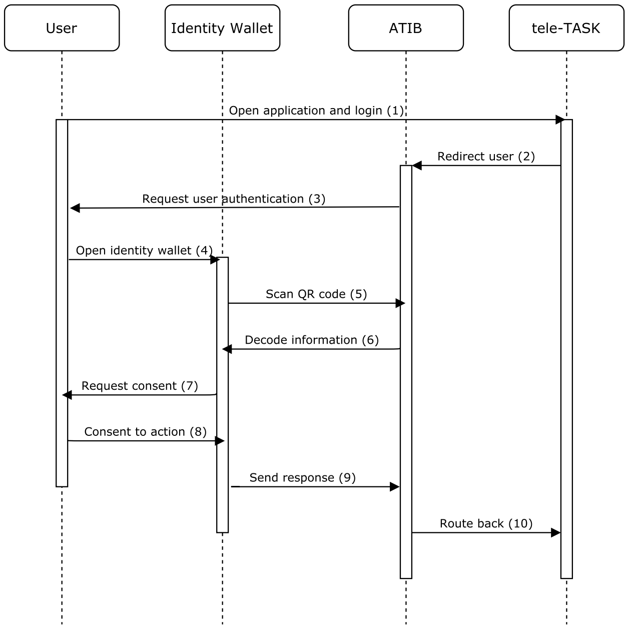
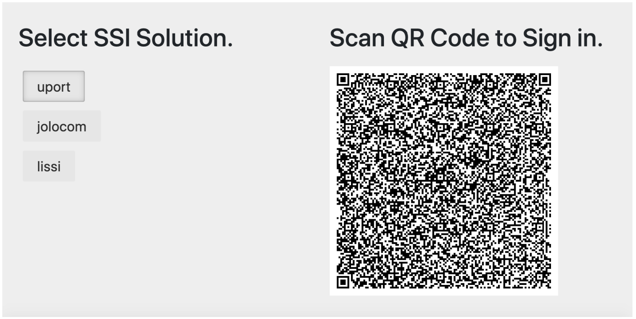


FIGURE 8. Authentication flow.



Select SSI Solution.

Scan QR Code to Sign in.

FIGURE 9. ATIB authentication challenge selection.

```
https://atib.local/oidc/authorization?
scope=openid+email&
redirect_uri=https%3A%2F%tele-task.local&
client_id=v5Zd7isg8932ghjk&
response_type=code
```

FIGURE 10. Redirect URI.

OIDC implementation with the blockchain authentication method is addressed.

Additionally, the redirection call contains the parameters *scope* with the value *openid* to indicate the respective flow. Additionally, the attribute *email* is requested. Furthermore, the redirection url and the client identifier is transmitted. The redirection url is required for validation when the user is routed back to tele-TASK after finishing the process. Fig. 10 presents the forwarding url to ATIB.

ATIB requests authentication from the user (3) by presenting the challenge of uPort as QR code [46]. uPort is preselected as SSI solution. However, the user can change to Jolocom or HL Aries/ Indy (see Fig. 9). The QR code is comprised of a JWT [47] that is signed by the identity of ATIB. Additionally, within the JWT the email address is encoded as requested attribute. In contrast, for HL Aries/ Indy the authentication challenge establishes only a connection between the identity wallet and ATIB. In a second communication step, ATIB retrieves required attributes.

The user opens the uPort identity wallet (4) and scans the QR code from ATIB (5). After scanning the code, the uPort mobile app decodes it (6) and verifies the signature of the JWT. Furthermore, the app identifies the requested attributes and asks the user for consent (7) to transmit the associated verifiable claims to the identity of ATIB. As the email address is requested, the previously obtained verifiable claim of the email address is conveyed to ATIB upon the user's consent (8).

The identity wallet creates a new JWT that contains the verifiable claims and transfers it to the callback address of ATIB. The callback address was included in the authentication challenge. ATIB processes the received JWT and verifies the signature. Additionally, the received verifiable claim of the email address is validated to ensure that it has not expired and has not been revoked. Subsequently, the trustworthiness of the claim is evaluated by the trust engine. As the claim has been issued by ATIB itself, it is considered as trusted. Thereafter, ATIB creates the OIDC ID token containing the DID of the user's identity and the email address as attribute. Additionally, the used SSI solution is provided as further information.

In conclusion, ATIB routes the user back to tele-TASK (10). At the same time, the ID token is also sent. Tele-TASK parses the ID token and acquires the email address attribute of the user. At this point, the authentication process is successfully completed and tele-TASK can authorize the user based on its attribute. The ID token is shown in Fig. 11.

```
{
  "email": "max.mustermann@test.com",
  "ssi": "uport",
  "sub": "001c67fc2e3f91b...77f95ff77546"
}
```

FIGURE 11. ID token.

B. TRANSFORMATION OF ATTRIBUTE REQUIREMENTS

To apply trust-enhancing attribute aggregation and integrate ATIB, the attribute requirements must be transformed to acceptance rules. We have shown the transformation for tele-TASK in the previous section. In the following paragraphs, we depict the acceptance rules for the ATIB User Interface and openHPI.[4]

1) ATIB USER INTERFACE

The ATIB User Interface provides the frontend for ATIB. The user can authenticate to verify and retrieve verifiable claims.

[4] openHPI.

The login process uses ATIB as identity provider facilitated by the OIDC protocol. Upon authentication, the ATIB User Interface requests the firstname, lastname or name as properties of the user. These attributes are only used for the welcome message. Potential wrong values have a very limited negative impact on the application. Thus, the acceptance threshold is set to 0. Therefore, self-attested claims are also forwarded. Table 5 provides an overview.

TABLE 5. Trust model characteristics for ATIB User Interface.

Attributes	Providers	Acceptance Rules
$\mathbb{A} = \{name, firstname, lastname\}$	$\mathbb{P} = \{ATIB, anonym\}$	$\mathbb{S} = \{\Theta \geq 0 \Rightarrow name, \Theta \geq 0 \Rightarrow firstname, \Theta \geq 0 \Rightarrow lastname\}$

2) openHPI

OpenHPI is an online learning platform that provides Massive Open Online Courses (MOOC). During the registration process the user enter its name and email address. The name is required for the issuance of participation certificates and the email address is used for communication with the user. The user has an own interest to provide the correct name. However, the email address is required for the application to function correctly. Thus, the threshold levels are on the one side set to 0 and 1. Thus, for the name a self-attested claim is sufficient. In contrast, for the email address an issued verifiable claim from trusted providers are required. Table 6 provides the overview.

TABLE 6. Trust model characteristics for OpenHPI.

Attributes	Providers	Acceptance Rules
$\mathbb{A} = \{email, name\}$	$\mathbb{P} = \{ATIB, anonym\}$	$\mathbb{S} = \{\Theta \geq 1 \Rightarrow email, \Theta \geq 0 \Rightarrow name\}$

C. PERFORMANCE

Our ATIB proof of concept implementation runs on a virtual machine with 1024 MB main memory and one CPU having 2.4 Ghz. clock rate. Nginx is used as reverse proxy to distribute the web requests to the ATIB application. In case several instances of the ATIB application are hosted, Nginx can also be used as load balancer.

On another virtual machine that is hosted in the same network, we run the tests with the support of the Locust [48] load testing framework. We conduct three test scenarios and determine the respective duration of the request:

1) **Open main page** Requesting the main page of ATIB. The home page is a simple web page. The duration for opening this page serves as baseline.
2) **Generate authentication challenge** Within this scenario the login page is requested. This creates the authentication challenge for a SSI solution.
3) **Perform authentication process** The test case comprises the complete authentication process by creating the authentication challenge and processing the respective response.

Each scenario is tested with a different number of concurrent user in Locust. Every test case is repeated several times to calculate an average duration and to remediate onetime effects.

The results are visualized in Fig. 12 as line charts. On the x-axis the number of concurrent user applied by the Locust framework are depicted. On the y-axis the execution time of one request is shown. The solid line in the chart reflects the first scenario. A constant request duration of about 10 milliseconds characterizes this simple request. The duration of requesting the login page is outlined as a dotted line in the chart. The execution time starts with about 1.4 seconds by one concurrent user and increases up to 33 seconds by using 40 concurrent user. Similarly, the duration for the complete authentication process lasts approximately 5 seconds with 1 concurrent user and raises to about 111 second per request for 40 concurrent user. Scenario 3 is shown as dashed line in the chart.

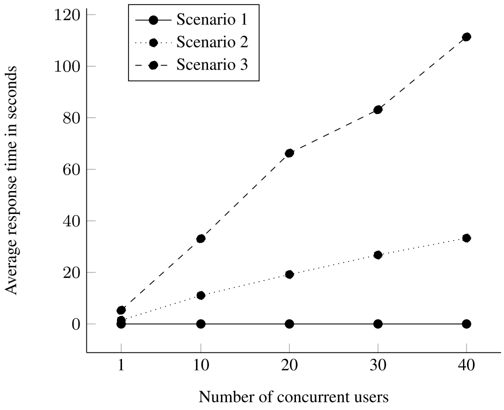


FIGURE 12. ATIB response times.

Overall, we can deduce that the increase of concurrent users significantly increases the execution times in scenario 2 and 3. Additionally, the duration of creating the authentication challenge and performing a complete authentication process takes substantially more time than solely delivering the main page to the user.

D. CONFORMANCE TO SSI PRINCIPLES

The SSI paradigm is grounded on the ten principles (cf. Section III-A) that are favourable for the user. We defined the non-impairment of these principles as requirement (**R7**) for ATIB. SP adoption must not undermine the user's privileges. The axioms **existence**, **persistence**, **portability** and **protection** are independent from ATIB and are primarily to be considered by the SSI solution itself. Therefore, no violation exists.

TABLE 7. Overview of attacks.

Security Objective	Attack	Adversary	Countermeasures
Integrity	Verifiable claim spoofing	User, external	Encrypted and signed data exchange; user authentication; access control
Integrity	Illegal service consumption	User, external	Encrypted and signed data exchange; user authentication; access control
Privacy	Retrieval of session information	External	Encrypted data exchange
Privacy	Retrieval of usage statistics	External	Encrypted data exchange
Availability	Service interruption	External	Increased scalability

Regarding the **control** principle, ATIB does not interfere with the user's control regarding its digital identity. The control is still with the user. ATIB routes the authentication to the SSI solution and the respective identity wallet.

Furthermore, the **access** criterion is also not violated. ATIB issues verifiable claims about the identity to the user's identity wallet after verification. The user can decide on the acceptance of the claim. Therefore, no additional claims are separately stored. During authentication, required claims are directly retrieved from the identity wallet.

In addition to that, the **interoperability** principle is actively supported by ATIB. ATIB enables an easier integration of SSI solutions by the SP. Therefore, more SPs may offer SSI for authentication. Likewise, the **consent** criterion is not impaired by ATIB. Claim disclosure requests during authentication or within a subsequent process are routed to the SSI solution and wallet. Thereby, the mechanism of the identity wallet for user consent is applied. There is no separate mechanism within ATIB that prevents or overrules the consent decision.

Moreover, ATIB adheres to the **minimalization** maxime. ATIB only requests attributes from the user that are demanded by the actual application for the authentication action. There are no additional properties inquired. The **transparency** principle requires an open functioning of ATIB without any hidden service. ATIB's functional layer is thin. Additionally, we provide the ATIB source code openly on Github.[5]

E. SECURITY

ATIB is a security relevant component for authentication and authorization at web applications. Therefore, security of ATIB itself is tremendously important. Within this section we conduct a security review by starting with an analysis of attacker types. Based on that, we provide an overview of attack vectors and use the attack tree methodology to closer explore illegal service consumption. Finally, we provide countermeasures that we have considered. In general, we integrated security into ATIB in the very beginning when posing the requirements, by designing the concept and implementing the application. Additionally, our review is focused on the attack surface that is introduced by ATIB. We do not review directly SSI related elements. This is subject to an analysis of the SSI solution itself.

1) ATTACKER TYPES

We differentiate between two types of attackers: internal and external. The internal category refers to participants using the ATIB functionality. In general, these are the user and the SP. The SP has no interest in attacking ATIB because ATIB is providing service for its applications. Additionally, ATIB is hosted in its organizational trust boundary. Despite that, the user may have interest in gaining extended privileges to the SP's domain.

Furthermore, the external class comprises attackers that are not related to the user or the SP. We see the threat from external attackers and the user as most prominent and evaluate it in the subsequent sections.

2) ATTACKS AND COUNTERMEASURES

We present an overview of the attacks in Table 7. Integrity, privacy and availability are security objectives for hosting and using the ATIB implementation. ATIB does not allow the illegitimate modification of managed data in particular verifiable claims. Privacy is significant for the user to keep application usage and claim values non-disclosed to the public. Furthermore, availability is fundamental to using the underlying service at all.

We clustered the attacks and countermeasures according to the security objective. Regarding integrity, we see verifiable claim and illegal service consumptions as the main attack vectors. Verifiable claim spoofing refers to the illegitimate retrieval of a verifiable claim that may contain a false value. Illegal service consumption means to use a SP application in an unauthorized way. To counteract these threats, we implemented encrypted and signed data exchange protocols. For instance, ATIB verifies that the signature of the verifiable claims have not been forged. Additionally, communication encryption is used to prevent interception.

Within the privacy domain, attacks to retrieve session information or application usage statistics by external attackers are the major threats. Likewise, we applied here communication encryption to prevent information leakage. Besides that, an attack to interrupt the service of ATIB targets the overall availability of SP services. We implement measures to increase scalability by running several instances of ATIB behind the Nginx proxy to distribute requests.

[5] https://github.com/agruener2000/ssixa-core

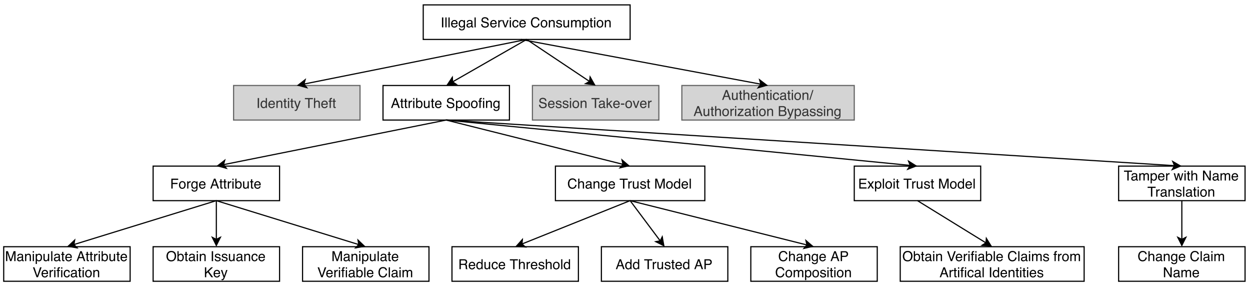

Illegal Service Consumption
Identity Theft
Attribute Spoofing
Session Take-over
Authentication/ Authorization Bypassing
Forge Attribute
Change Trust Model
Exploit Trust Model
Tamper with Name Translation
Manipulate Attribute Verification
Obtain Issuance Key
Manipulate Verifiable Claim
Reduce Threshold
Add Trusted AP
Change AP Composition
Obtain Verifiable Claims from Artifical Identities
Change Claim Name

FIGURE 13. **Illegal service consumption attack tree.**

3) ILLEGAL SERVICE CONSUMPTION

An attack with the objective of illegal service consumption has a serious impact on the SP. Based on our concentration on the SP, this attack objective is our major focus. From the SP's perspective, such an attack must be prevented to avoid technical or financial harm. We applied the attack tree methodology [49] to better understand this attack and the potentially required steps. Fig. 13 shows the attack tree for illegal service consumption. We focused on attack vectors originating from discrete ATIB functionality. Therefore, identity theft, session take-over and authentication/ authorization bypassing is marked with a grey box and is not further analyzed. Existing literature [50], [51] sufficiently focus on these topics.

In contrast, attribute spoofing is a significant new attack vector. The attributes of a user are essential for service consumption. Attribute spoofing approaches are categorized in attribute forging, changing or exploiting the trust model and tampering with the name translation. In the following paragraphs, we describe these vectors.

a: FORGE ATTRIBUTE

The attack vector *Forge Attribute* refers to approaches that target the counterfeiting of a single verifiable claim.

- **Manipulate Attribute Verification:** In general, the issuer of the verifiable claim runs a verification procedure to validate the attribute value. Within this attack vector, the verification process is exploited to obtain a claim that does not correspond with the reality. The detailed strategy depends on the type of the claim and the chosen verification procedure.
- **Obtain Issuance Key:** The issuer of a verifiable claim signs the document with a private key. Arbitrary claims can be issued independently of any verification process if an attacker acquires possession of the private key.
- **Manipulate Verifiable Claim:** A verifiable claim that is legitimately issued to an attacker can be manipulated. The manipulation can target the claim metadata or the attribute value. For instance, the claim value can be a cryptographic hash of a document that testifies authenticity. If the applied cryptographic hash function is vulnerable to attacks, the verifiable claim can be manipulated.

b: CHANGE TRUST MODEL

The modification of the ATIB trust model is a further attack vector. Manipulating the trust model leads to a change in the subjective trustworthiness of APs. An attacker would acquire illegitimate access to ATIB for this type of manipulation. Moreover, social engineering can be applied to the ATIB hosting entity for this attack vector as well.

- **Reduce Threshold:** An attribute threshold represents a barrier. If the computed trust score of the APs exceeds the threshold, the claim is accepted as trustworthy. The reduction of the threshold leads to the acceptance of claims that originates from lower trusted APs.
- **Add Trusted AP:** ATIB stores the trusted APs for the applied trust model. An attacker may add more trusted APs or even public keys of an own AP. As a result, ATIB accepts verifiable claims that are issued by these APs.
- **Change AP Composition:** Manipulating the trust composition from different APs interferes with the originally established trust model. Therefore, APs can be preferred that have weak verification procedures.

c: EXPLOIT TRUST MODEL

Besides the manipulation of the trust model, an existing trust model can be exploited without a change. This attack approach targets characteristics of the applied trust model.

- **Obtain Verifiable Claims from Artificial Identities:** A trust model may apply the composition of several unknown APs. For instance, if ten APs attest a certain claim, the claim is accepted as attribute. The attacker generates public keys as many as needed to create the amount of required attestations. Subsequently, the attacker presents the verifiable claims to ATIB during the authentication process.

d: TAMPER WITH NAME TRANSLATION

ATIB applies claim name translation to decode between different standards. Illegitimate intervention in the translation process is an attack approach.

- **Change Claim Name:** An attacker circumvent access controls to ATIB and replaces claim names. For instance, a claim name that has a high trust posture is replaced with the name of a claim with a low trust posture.

F. USAGE STATISTICS

We published a publicly available instance of ATIB in the Internet under the domain SSI 4 All[6] in March 2019. At the same time, we published our initial paper [39]. The instance offered authentication with uPort and Jolocom where uPort was the preselected login option. At this time, uPort was one of the most mature SSI solutions comprising a general available identity wallet.

Fig. 14 presents statistics for authentication challenge creation for the year 2019. Additionally, Fig. 15 shows the same scenario for the year 2020. As ATIB went live in March, the capture of statistics started at the same time. After June the application was not used anymore in 2019. Generally, the number of created challenges for uPort outweigh the usage of Jolocom. This may result from using uPort as default authentication solution. However, it can also indicate a higher popularity. In 2020, the overall usage and popularity of the web site significantly decreased.

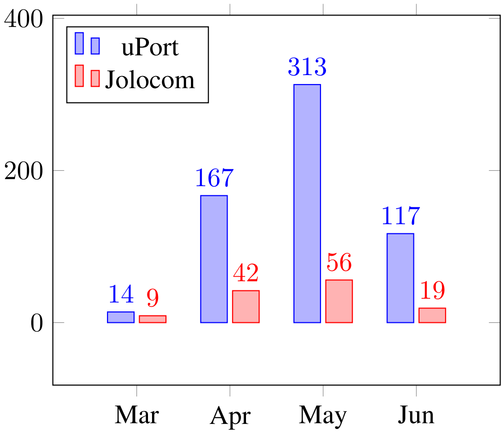


FIGURE 14. ATIB challenge creation statistics for year 2019.

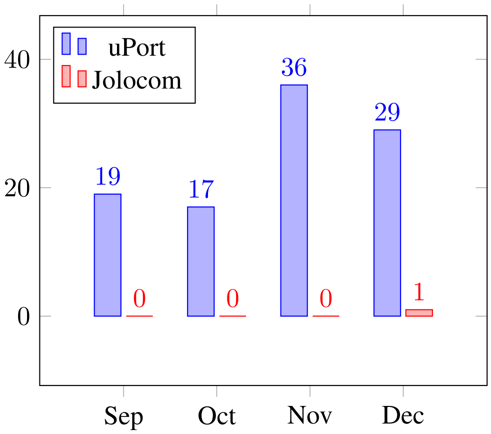


FIGURE 15. ATIB challenge creation statistics for year 2020.

Despite that, Fig. 16 outlines statistics for a complete authentication process at the ATIB User Interface.

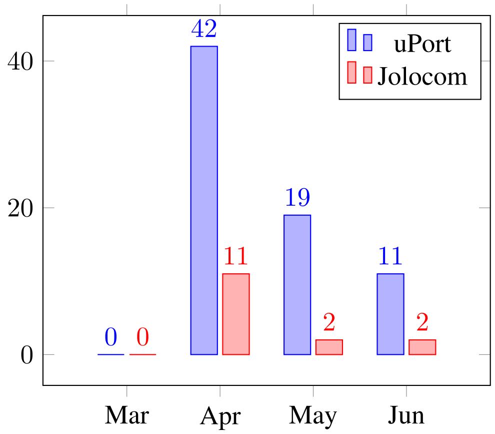


FIGURE 16. ATIB User Interface authentication statistics year 2019.

The highest number of authentication processes have been measured shortly after the go live. Subsequently, the numbers declined.

IX. DISCUSSION

ATIB enables the practical usage of SSI solutions for SPs. However, the SSI solutions themselves are in an early stage of development. Therefore, the SSI implementations have a high frequency of change leading to fast updates of libraries and integration patterns. Thus, the SSI wrapper for the specific solutions may require constant updates within short intervals. Nonetheless, updating the SSI wrappers of ATIB seems to be less effort compared to changing direct application integrations.

Besides that, a fundamental feature of ATIB is the trust-enhancing attribute aggregation of verifiable claims. To fully use this feature, ATIB requires information about the DIDs that are used by the trusted APs. Subsequently, the host of ATIB defines for each AP an opinion about trustworthiness. However, there is no central lookup dictionary to search for DIDs of an AP or any other organization. Furthermore, large organizations that traditionally provide trust may not already have DIDs in SSI networks due to the early development stage. Therefore, the applicability of the trust-enhanced attribute aggregation is currently limited.

X. CONCLUSION

In traditional identity management patterns, the IdP is a TTP with significant disadvantages for the user and the SP. The emerging blockchain-based SSI pattern significantly changes this situation by implementing a decentralized IdP and eventually decoupling the identifier from the attributes of an identity. The SSI pattern and the solutions are focused on the users and their needs, but disregard requirements on the side of the SP. Despite that, non-usage of established protocols and plentiful SSI solutions that strive for the favor of the user are challenges for SP adoption. To overcome these challenges and to make use of attributes that are decoupled from the identifier, we proposed ATIB, an Attribute Trust-Enhancing

[6] https://ssixa.de

Identity Broker for SPs. The component-based architecture abstracts from a single SSI solution, enables the issuance of verifiable claims and applies trust models for flexible trust decisions in attributes.

Furthermore, we implemented ATIB as a proof of concept with connection to uPort, Jolocom and HL Aries/ Indy. In the evaluation, we showed a complete authentication workflow with attributes used for authorization, measured performance metrics and reviewed the conformance for SSI principles. Ancillary, we showed the transformation of attribute requirements in acceptance rules and provided usage statistics of the publicly available ATIB instance. Finally, a security analysis showed attack patterns on integrity, privacy and availability and the respective countermeasures that we have taken to secure ATIB.

XI. FUTURE WORK

As outlined in Section IX, an open challenge is the secure determination of a DID for an organization or person and the retrieval thereof. Potential solutions might comprise approaches based on certificates, central registries or decentralized attestations. However, a central registry counteracts the decentralization principles of SSI. An approach that provides global availability of DIDs fosters a web of trust for trust-enhancing attribute aggregation and to reduce the dependency towards a minority of APs.

A myriad of SSI solutions exists. Thus, the development of a global applicable identity based on SSI requires interoperability. Besides that, the interoperability principle is also core to Allen's [20] SSI axioms. As ATIB's architecture is an interoperability approach for SSI solutions at the side of the service provider, further interoperability concepts should be researched and compared to the identity broker approach.

REFERENCES

[1] G. Williamson, D. Yip, I. Sharoni, and K. Spaulding, *Identity Management: A Primer*. Boise, ID, USA: MC Press, 2009.

[2] A. Jøsang, J. Fabre, B. Hay, J. Dalziel, and S. Pope, "Trust requirements in identity management," in *Proc. Australas. Workshop Grid Comput. e-Res. (ACWS)*, 2005, pp. 99–108.

[3] P. Windley, *Digital Identity*. Newton, MA, USA: O'Reilly Media, 2005.

[4] A. Grüner, A. Mühle, T. Gayvoronskaya, and C. Meinel, "A comparative analysis of trust requirements in decentralized identity management," in *Proc. 33rd Int. Conf. Adv. Inf. Netw. Appl. (AINA)*, vol. 926, 2019, pp. 200–213.

[5] A. Narayanan and J. Clark, "Bitcoin's academic pedigree," *Commun. ACM*, vol. 60, no. 12, pp. 36–45, 2017.

[6] M. Kuperberg, "Blockchain-based identity management: A survey from the enterprise and ecosystem perspective," *IEEE Trans. Eng. Manag.*, vol. 67, no. 4, pp. 1008–1027, Nov. 2019.

[7] M. Sporny, D. Longley, and D. Chadwick. (2019). *Verifiable Credentials Data Model 1.0. Expressing Verifiable Information on the Web*. Accessed: Apr. 15, 2021. [Online]. Available: https://www.w3.org/TR/vc-data-model/

[8] B. Hulsebosch, M. Wegdam, B. Zoetekouw, N. van Dijk, and R. P. van Wijnen. (2011). *Virtual Collaboration Attribute Management*. Accessed: Jul. 19, 2019. [Online]. Available: https://www.surf.nl/binaries/content/assets/surf/en/knowledgebase/2012/EDS+11-06+Attribute+Management+v1.0.pdf

[9] M. S. Ferdous and R. Poet, "Analysing attribute aggregation models in federated identity management," in *Proc. 6th Int. Conf. Secur. Inf. Netw. (SIN)*, 2013, pp. 181–188.

[10] D. W. Chadwick, G. Inman, and N. Klingenstein, "A conceptual model for attribute aggregation," *Future Gener. Comput. Syst.*, vol. 26, no. 7, pp. 1043–1052, Jul. 2010.

[11] D. W. Chadwick and G. Inman, "The trusted attribute aggregation service (TAAS)—Providing an attribute aggregation layer for federated identity management," in *Proc. Int. Conf. Availability, Rel. Secur. (ARES)*, Sep. 2013, pp. 285–290.

[12] M. S. Ferdous, F. Chowdhury, and R. Poet, "A hybrid model of attribute aggregation in federated identity management," in *Proc. 2nd Int. Workshop Enterprise Secur. (ES)*, 2017, pp. 120–154.

[13] K. Yamaji, T. Kataoka, M. Nakamura, T. Orawiwattanakul, and N. Sonehara, "Attribute aggregating system for shibboleth based access management federation," in *Proc. 10th IEEE/IPSJ Int. Symp. Appl. Internet*, Jul. 2010, pp. 281–284.

[14] N. Klingenstein, "Attribute aggregation and federated identity," in *Proc. Int. Symp. Appl. Internet Workshops*, Jan. 2007, p. 26.

[15] J. Gemmill, J.-P. Robinson, T. Scavo, and P. Bangalore, "Cross-domain authorization for federated virtual organizations using the myVocs collaboration environment," *Concurrency Comput., Pract. Exper.*, vol. 21, no. 4, pp. 509–532, Mar. 2009.

[16] (Accessed: Apr. 15, 2021). *The Linux Foundation*. Hyperledger Aries. [Online]. Available: https://www.hyperledger.org/projects/aries

[17] (Accessed: Apr. 15, 2021). *Hyperledger Indy*. [Online]. Available: https://www.hyperledger.org/projects/indy

[18] (Accessed: Apr. 15, 2021). *Universal Resolver*. [Online]. Available: https://github.com/decentralized-identity/universal-resolver

[19] D. Reed, M. Sporny, D. Longley, C. Allen, R. Grant, and M. Sabadello. (Accessed: Apr. 15, 2021). *Decentralized Identifiers (DIDS) V1.0. Core Data Model and Syntaxes*. [Online]. Available: https://www.w3.org/TR/did-core/

[20] C. Allen. (2016). *The Path to Self-Sovereign Identity*. Accessed: Apr. 15, 2021. [Online]. Available: http://www.lifewithalacrity.com/previous/2016/04/the-path-to-self-sovereign-identity.html

[21] A. Tobin and D. Reed. (2016). *The Inevitable Rise of Self-Sovereign Identity. The Sovrin Foundation*. Accessed: Apr. 15, 2021. [Online]. Available: https://sovrin.org/wp-content/uploads/2018/03/The-Inevitable-Rise-of-Self-Sovereign-Identity.pdf

[22] S. Nakamoto. (2008). *Bitcoin: A Peer-to-Peer Electronic Cash System*. Accessed: Apr. 15, 2021. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[23] G. O. Karame, E. Androulaki, M. Roeschlin, A. Gervais, and S. Čapkun, "Misbehavior in bitcoin: A study of double-spending and accountability," *ACM Trans. Inf. Syst. Secur.*, vol. 18, no. 1, pp. 1–32, Jun. 2015.

[24] G. Wood. *Ethereum: A Secure Decentralised Generalised Transaction Ledger*. Accessed: Apr. 15, 2021. [Online]. Available: https://pdfs.semanticscholar.org/ac15/ea808ef3b17ad754f91d3a00fedc8f 96b929.pdf

[25] (Accessed: Apr. 15, 2021). *The Linux Foundation*. Hyperledger Fabric. [Online]. Available: https://www.hyperledger.org/projects/fabric

[26] A. Mühle, A. Grüner, T. Gayvoronskaya, and C. Meinel, "A survey on essential components of a self-sovereign identity," *Comput. Sci. Rev.*, vol. 30, pp. 80–86, Nov. 2018.

[27] M. Sporny, D. Longley, and D. Chadwick. (Accessed: Apr. 15, 2021). *Verifiable Credentials Data Model 1.0. Expressing Verifiable Information on the Web*. [Online]. Available: https://www.w3.org/TR/vc-data-model/

[28] C. Lundkvist, R. Heck, J. Torstensson, Z. Mitton, and M. Sena. (2016). *Uport: A Platform for Self-Sovereign Identity*. Accessed: Apr. 15, 2021. [Online]. Available: http://blockchainlab.com/pdf/uPort_whitepaper_DRAFT20161020.pdf

[29] (Accessed: Apr. 15, 2021). *Jolocom*. [Online]. Available: https://jolocom.io

[30] (Accessed: Apr. 15, 2021). *Shocard*. [Online]. Available: https://shocard.com

[31] (Accessed: Apr. 15, 2021). *Blockchain Helix*. [Online]. Available: https://blockchain-helix.com

[32] J. Sabater and C. Sierra, "Review on computational trust and reputation models," *Artif. Intell. Rev.*, vol. 24, no. 1, pp. 33–60, Sep. 2005.

[33] A. Jøsang, R. Ismail, and C. Boyd, "A survey of trust and reputation systems for online service provision," *Decis. Support Syst.*, vol. 43, no. 2, pp. 618–644, 2007.

[34] A. Grüner, A. Mühle, and C. Meinel, "Using probabilistic attribute aggregation for increasing trust in attribute assurance," in *Proc. IEEE Symp. Ser. Comput. Intell. (SSCI)*, Dec. 2019, pp. 633–640.
[35] (Accessed: Apr. 15, 2021). *OpenID Foundation*. Openid Connect Core 1.0. [Online]. Available: https://openid.net/specs/openid-connect-core-1_0.html
[36] (Accessed: Apr. 15, 2021). *OASIS. Saml Version 2.0*. [Online]. Available: http://docs.oasis-open.org/security/saml/v2.0/sstc-saml-approved-errata-2.0.pdf
[37] M. Sabadello, K. D. Hartog, C. Lundkvist, C. Franz, A. Elias, A. Hughes, J. Jordan, and D. Zagidulin. (Accessed: Apr. 15, 2021). *Introduction to Did Auth*. [Online]. Available: https://github.com/WebOfTrustInfo/rwot6-santabarbara/blob/master/final-documents/did-auth.md
[38] (Accessed: Apr. 15, 2021). *Decentralized Identity Foundation*. Authentication Working Group. Didcomm. [Online]. Available: https://identity.foundation/working-groups/authentication.html
[39] A. Grüner, A. Mühle, and C. Meinel, "An integration architecture to enable service providers for self-sovereign identity," in *Proc. IEEE 18th Int. Symp. Netw. Comput. Appl. (NCA)*, Sep. 2019, pp. 1–5.
[40] (Accessed: Apr. 15, 2021). *Tornado*. [Online]. Available: https://www.tornadoweb.org/en/stable/
[41] (Accessed: Apr. 15, 2021). *Pyoidc*. [Online]. Available: https://pyoidc.readthedocs.io/en/latest/
[42] (Accessed: Apr. 15, 2021). *Pysaml2*. [Online]. Available: https://pypi.org/project/pysaml2/
[43] (Accessed: Apr. 15, 2021). *Rinkeby*. [Online]. Available: https://www.rinkeby.io
[44] (Accessed: Apr. 15, 2021). *Hyperledger Aries. Cloud Agent Python*. [Online]. Available: https://github.com/hyperledger/aries-cloudagent-python
[45] (Accessed: Apr. 15, 2021). *Verifiable Organization Network*. [Online]. Available: https://vonx.io
[46] International Standardization Organization. *Information Technology—Automatic Identification and Data Capture Techniques—Bar Code Symbology—QR Code*. Standard ISO/IEC 18004:2000, 2000. Accessed: Apr. 15, 2021. [Online]. Available: https://tools.ietf.org/html/rfc7159
[47] *Internet Engineering Task Force (IETF)*, document RFC 7519, Json Web Token (JWT), 2021. Accessed: Apr. 15, 2021. [Online]. Available: https://tools.ietf.org/html/rfc7519
[48] (Accessed: Apr. 15, 2021). *Locust. A Modern Load Testing Framework*. [Online]. Available: https://locust.io
[49] B. Schneier, "Attack trees," *Dr. Dobb's J. Softw. Tools*, vol. 24, no. 12, pp. 21–29, 1999.
[50] W. Burgers, R. Verdult, and M. van Eekelen, "Prevent session hijacking by binding the session to the cryptographic network credentials," in *Proc. Nordic Conf. Secure IT Syst. (NordSec)*, 2013, pp. 33–50.
[51] M. Dalton, C. Kozyrakis, and N. Zeldovich, "Nemesis: Preventing authentication & access control vulnerabilities in web applications," in *Proc. 18th Conf. USENIX Secur. Symp. (SSYM)*, 2009, pp. 267–282.

ANDREAS GRÜNER received the B.S. degree in public administration from the Federal University of Applied Administrative Science, Brühl, Germany, in 2008, and the B.S. and M.S. degrees in computer science from Humboldt University, Berlin, Germany, in 2012. He is currently pursuing the Ph.D. degree in IT systems engineering with a research focus on self-sovereign identity concentrating on related trust models and interoperability approaches.

ALEXANDER MÜHLE is currently pursuing the Ph.D. degree with the Chair for Internet Technologies and Systems, Hasso Plattner Institute. After graduating at TU Berlin, in 2018, with a focus on communication networks, his research at HPI has been on P2P network analysis and self-sovereign identity in the context of digital academic credentials.

CHRISTOPH MEINEL is the CEO and the Scientific Director of Hasso Plattner Institute for Digital Engineering gGmbH (HPI). He is also the Vice Dean of the Faculty of Digital Engineering, University of Potsdam. He is the Director of the HPI-Stanford Design Thinking Research Program, an Honorary Professor at TU Beijing, a Visiting Professor at Shanghai University, and a Concurrent Professor at Nanjing University. He holds the Chair of internet technologies and systems. He is engaged in the fields of cybersecurity and digital education. He has developed the MOOC platform openHPI.de and supervises numerous Ph.D. students. He is a Teacher at HPI School of Design Thinking, where he is also scientifically active in research. His earlier scientific work concentrated on efficient algorithms and complexity theory. He is the author or the coauthor of more than 25 books, anthologies, and numerous conference proceedings. He has had more than 550 (peer-reviewed) papers published in scientific journals and at international conferences and holds a number of international patents.

Prof. Meinel is a member of the National Academy of Science and Engineering (acatech). He is also a member of numerous scientific committees and supervisory boards.