IEICE TRANSACTIONS

on Information and Systems

DOI:10.1587/transinf.2023BCI0001

Publicized:2024/04/15

A PUBLICATION OF THE INFORMATION AND SYSTEMS SOCIETY
The Institute of Electronics, Information and Communication Engineers
Kikai-Shinko-Kaikan Bldg., 5-8, Shibakoen 3 chome, Minato-ku, TOKYO, 105-0011 JAPAN

INVITED PAPER *Special Section on Security and Privacy in Blockchain*

Permissionless Blockchain-Based Sybil-Resistant Self-Sovereign Identity Utilizing Attested Execution Secure Processors*

Koichi MORIYAMA[†], ***Member and*** **Akira OTSUKA**[††], ***Senior Member***

SUMMARY This article describes the idea of utilizing Attested Execution Secure Processors (AESPs) that fit into building a secure Self-Sovereign Identity (SSI) system satisfying Sybil-resistance under permissionless blockchains. Today's circumstances requiring people to be more online have encouraged us to address digital identity preserving privacy. There is a momentum of research addressing SSI, and many researchers approach blockchain technology as a foundation. SSI brings natural persons various benefits such as owning controls; on the other side, digital identity systems in the real world require Sybil-resistance to comply with Anti-Money-Laundering (AML) and other needs. The main idea in our proposal is to utilize AESPs for three reasons: first is the use of attested execution capability along with tamper-resistance, which is a strong assumption; second is powerfulness and flexibility, allowing various open-source programs to be executed within a secure enclave, and the third is that equipping hardware-assisted security in mobile devices has become a norm. Rafael Pass et al.'s formal abstraction of AESPs and the ideal functionality $\mathcal{G}_{\mathtt{att}}$ enable us to formulate how hardware-assisted security works for secure digital identity systems preserving privacy under permissionless blockchains mathematically. Our proposal of the AESP-based SSI architecture and system protocols, $\Pi^{\mathcal{G}_{\mathtt{att}}}$, demonstrates the advantages of building a proper SSI system that satisfies the Sybil-resistant requirement. The protocols may eliminate the online distributed committee assumed in other research, such as CanDID, because of assuming AESPs; thus, $\Pi^{\mathcal{G}_{\mathtt{att}}}$ allows not to rely on multi-party computation (MPC), bringing drastic flexibility and efficiency compared with the existing SSI systems.

key words: *Permissionless Blockchain, Decentralized Digital Identity, Sybil-Resistance, Self-Sovereign Identity, and Attested Execution Secure Processors*

1. Introduction

Inspiration of David Chaum's article in 1985 to avoid unexpected tracing by someone else like Big Brother[1] by utilizing pseudonyms, digital signatures, and card computers [2] have encouraged the authors to consider how to solve a conflicting problem of preserving privacy and to meet the Sybil-resistant requirement dealing with Anti-Money-Laundering (AML) and other needs in digital identity systems. The card computer expressed in 1985 was a dream written as a vision; however, it has become real, and secure processors are also becoming a norm in various mobile devices such as smartphones as a mandatory requirement today. Why don't we utilize such capability?

Self-Sovereign Identity (SSI) [3] is the momentum in academia and the tech industry. Christoper Allen expressed the history of digital identity and the expectation of SSI in his blog article in 2016 [4]. Since then, there have been many studies, researches, and implementations until now [5][6][7]. The terminology "Self-Sovereign" inspires many people to think about how SSI can protect privacy and resolve reliance on authorities that may control personal data. Those efforts are not limited to technology, government, and human beings. One of those researches addresses the relationship between SSI and GDPR [8], while there are already some initiatives on utilizing SSI in Europe [9].

Many pieces of research in this domain have addressed SSI systems architecture utilizing blockchain technology [10][11]. Some researchers discussed the necessity of blockchain; however, they still recognize that blockchain technology is a good foundation [12]. All the well-known existing implementations utilize blockchains, such as uPort on Ethereum[2] [13] and Sovrin on the Sovrin ledger [14]. However, surprisingly – to the best of our knowledge, no study has addressed utilizing hardware-assisted security [15][16] implemented within mobile devices that people own for their daily lives. Several studies and implementations address mobile apps for SSI systems but focus on user experiences and do not address security feature perspectives [17][13].

This paper proposes a permissionless blockchain-based SSI systems architecture that utilizes the formal abstraction of Attested Execution Secure Processors (AESPs) [18] equipped with mobile devices.

Contributions

- Demonstrate the powerfulness of hardware-assisted security and the formal abstraction of AESPs that fit to build a secure SSI system satisfying the Sybil-resistance requirement.
- In concrete, propose the AESP-based SSI systems architecture and protocols, $\Pi^{\mathcal{G}_{\mathtt{att}}}$, with its construction, security properties such as Sybil-resistance, and a proof sketch of the security properties.
- Assuming AESPs and $\mathcal{G}_{\mathtt{att}}$, the AESP-based SSI system protocols $\Pi^{\mathcal{G}_{\mathtt{att}}}$ eliminates the online distributed committee of trusted nodes assumed in CanDID [19]. Thus, $\Pi^{\mathcal{G}_{\mathtt{att}}}$ allows not to rely on multi-party computation (MPC) that requires such a trusted party of nodes,

[†] The author is with the Institute of Information Security, Japan, and also belongs to NTT DOCOMO, INC.

[††] The author is with the Institute of Information Security, Japan.

[*] This paper was presented at IEEE Blockchain 2022 [1].

[1] See George Orwell's novel, *"Nineteen Eighty-Four(1984) – Big Brother Is Watching You,"* published in 1948.

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

and it brings more flexibility and efficiency than the existing systems.

2. Background and Related Work

2.1 Digital Identity and Decentralized Digital Identity

The importance of "digital identity" is rapidly increasing under the current circumstances, even after the pandemic. People would need to do much more things online than before. Researchers and influencers in the tech industry in this domain refer to Kim Cameron's blog article, "The Laws of Identity" [20]. Beyond the contribution, researchers and the industry have made significant efforts, including standardization bodies resolving many problems from various perspectives, such as identity proofing, authentication, and federation.

In digital identity approaches, managing claims and credentials is one of the essential elements of representing who I am or who you are. Identity is a set of attributes or claims by definitions, e.g., ISO/IEC 24760-1:2019/Amd 1:2023(en) [21], and a credential represents an identity for authentication. There are also many activities at standardization bodies such as W3C[1], OpenID Foundation[2], and FIDO Alliance[3]. The NIST's Digital Identity Guidelines [22] is a set of guidelines that address various perspectives through its digital identity model.

Digital identity management started from the Isolated User Identity (SILO) model and moved to the Federated User Identity (FED) model by the mathematical definition of Md Sadek Ferdous et al.'s work [23]. The (full) identity representing a natural person is a union of partial identities, each set of claims consisting of an attribute and value pair. They demonstrated that digital identity and identity management move from the most straightforward model toward federated models in a decentralized fashion.

In the tech industry, several initiatives are addressing decentralized digital identity. Microsoft has been driving an initiative[4] and announced ION[5] on behalf of the Decentralized Identity Foundation (DIF)[6] in May 2021. The approach utilizes W3C's DIDs [24], decentralized systems such as blockchains and ledgers, and DIF's standards.

2.2 Hardware-Assisted Security and Attested Execution Secure Processors (AESPs)

Hardware-assisted security may provide tamper-resistant features, and some of them support attested execution capability. Such hardware-assisted security has recently been becoming the norm for mobile devices.

Apple's iPhone implements Secure Enclave, a dedicated secure subsystem. It is isolated from the app execution environment on the main processor[7]. Android devices support KeyStore and other security-related functionality utilizing hardware-assisted implementations, e.g., TEE (Trusted Execution Environment), Arm's TrustZone in particular. Google recently announced the Android Ready SE program[8], which will be supporting hardware-backed security applets for various use cases such as digital keys and identity credentials. In addition, Microsoft recently announced Windows 11 with new hardware requirements in which TPM (Trusted Platform Module) 2.0 is mandated[9]. There are also other design choices among implementations in the industry, such as Global Platform-supported Secure Elements[10] and Intel's SGX[11], in addition to TrustZone and TPM 2.0.

Among numerous implementations and research addressing hardware-assisted security, including how to realize [15] and how to utilize [25], Rafael Pass et al. uniquely addressed hardware-assisted security, secure processors, in a formal fashion [18]. In their words, trusted hardware is commonly believed to provide a powerful abstraction for building secure systems. They approached to formalize the attested execution abstraction and retrieved the formal modeling of a broad class of Attested Execution Secure Processors (AESPs) from the common belief. The authors are very encouraged to utilize Rafael Pass et al.'s formal abstraction of AESPs to formulate and demonstrate our proposed scheme in this paper.

2.3 Self-Sovereign Identity (SSI)

Christopher Allen published his blog article entitled "The Path to Self-Sovereign Identity" in 2016 [4]. It presented the evolution of digital identity from Phase 1: Centralized Identity, Phase 2: Federated Identity, Phase 3: User-Centric Identity through Phase 4: Self-Sovereign Identity, followed by his definition of SSI with the ten principles:

1. **Existence**. *Users must have an independent existence.*
2. **Control**. *Users must control their identities.*
3. **Access**. *Users must have access to their own data.*
4. **Transparency**. *Systems and algorithms must be transparent.*
5. **Persistence**. *Identities must be long-lived.*
6. **Portability**. *Information and services about identity must be transportable.*
7. **Interoperability**. *Identities should be as widely usable as possible.*
8. **Consent**. *Users must agree to the use of their identity.*

[1] https://www.w3.org

[2] https://openid.net/foundation/

[3] https://fidoalliance.org

[4] https://www.microsoft.com/en-us/security/business/identity-access-management/decentralized-identity-blockchain

[5] https://identity.foundation/ion/

[6] https://identity.foundation

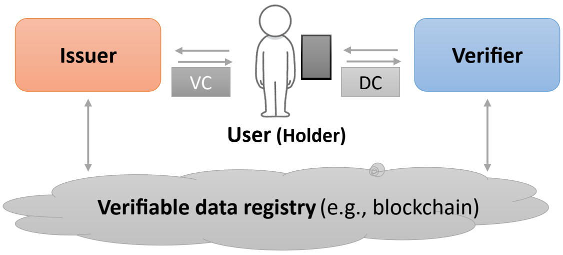
[7] https://support.apple.com/guide/security/secure-enclave-sec59b0b31ff/web

[8] https://developers.google.com/android/security/android-ready-se

[9] https://docs.microsoft.com/en-us/windows/security/information-protection/tpm/trusted-platform-module-overview

[10] https://globalplatform.org/resource-publication/introduction-to-secure-elements/

[11] https://www.intel.com/content/www/us/en/architecture-and-technology/software-guard-extensions.html



VC: Verifiable Credential
DC: Derived Credential (for presentation)

Fig. 1 Self-Sovereign Identity Systems in the Tech Industry

9. **Minimalization**. *Disclosure of claims must be minimized.*
10. **Protection**. *The rights of users must be protected.*

2.3.1 Extended Principles

Some research addressed whether the ten principles express all principles that may describe the essentials of SSI. Quinten Stokkink et al. proposed to add another principle **Provable** [11]. Md Sadek Ferdous et al. presented five taxonomies and 17 principles under the taxonomies of classes derived from the ten principles in his comprehensive survey [6]. Abylay Satybaldy et al. proposed to add **Usability** in their SSI evaluation framework, which also refers to the ten principles as a comprehensive spectrum of SSI requirements [26].

2.3.2 Building Blocks and Blockchain

Many pieces of research addressed how to build SSI systems; essential components [10], design patterns [27], and needs of, how to utilize, or if it requires blockchain technology [11][28][12][29][8][30]. Two of these research papers concluded that blockchain was not mandated [28][12]. However, they still recognize that blockchain technology is a good foundation to build an SSI system and indicated that some specific requirements would require further extra efforts to fill in gaps.

2.3.3 SSI Systems with Mobile Devices

There are several SSI implementations, such as uPort and Sovrin [29][14]. Also, some experimental research and prototypes in the tech industry support governmental agencies' interests [31][32]. Through such activities, including W3C's efforts on verifiable credentials and DIDs [33][24], there has been becoming a common structure and primary roles involved in exchanging verifiable credentials among an issuer, a holder (a natural person), and a verifier.

Fig. 1 illustrates such an SSI solution architecture in our interpretation. In the figure, VC stands for Verifiable Credential, and DC stands for Derived Credential. An issuer issues the holder a verifiable credential (VC). They may have a derived credential (DC) for presentation to minimize disclosure. A verifier may verify with the received derived credential per a request. Blockchain technology can be a verifiable data registry in the architecture.

We put a mobile phone next to the user in the figure because some SSI implementations provide a mobile app, such as a wallet app, for their use with the SSI systems. To the best of our knowledge, however, such mobile apps never play their roles in utilizing hardware-assisted security features of the mobile device. Kalman C. Torh et al. addressed using users' mobile devices as a digital identity for each of them [17]; however, they have not mentioned opportunities for hardware-backed attestations.

2.4 Sybil-Attack and Sybil-Resistance

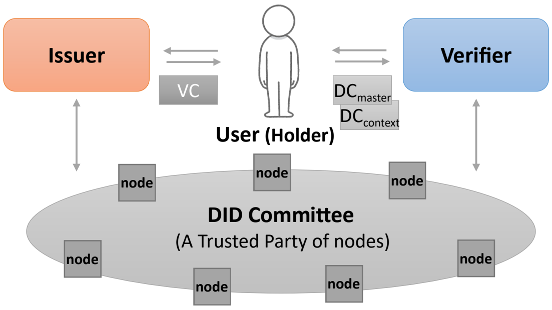
"Sybil" is a book by Flora Rheta Schreiber in 1973 and a pseudonym for Shirley Ardell Mason, who was in dissociative identity disorder with 16 multiple personalities in the book. John. R. Douceur, a researcher at Microsoft, suggested in 2002 to name a type of attack by creating a large number of pseudonymous identities and using them to gain a disproportionately large influence [34]. Sybil-resistance has become a critical requirement to deal with impersonation by preventing Sybil-attack.

3. Preliminaries

3.1 CanDID

It can do decentralized identity with legacy compatibility, Sybil-resistance, and accountability. Among numerous research addressing decentralized digital identity, Deepak Maram et al. identified remaining problems for building a decentralized identity system, legacy compatibility, Sybil-resistance, and accountability as entitled [19]. To solve the problems, they proposed system protocols with a trusted committee of nodes-based architecture.

Fig. 2 illustrates the overview of CanDID's approach. In the figure, VC is a verifiable credential, $\mathrm{DC}_{\mathrm{master}}$ means a master credential, and $\mathrm{DC}_{\mathrm{context}}$ means a context-based credential in their work, and a combination of two credential



VC: Verifiable Credential (pre-credentials in terms of CanDID)
DC: Derived Credential (for presentation)
$\mathrm{DC}_{\mathrm{master}}$: Master Derived Credential for Sybil-resistance
$\mathrm{DC}_{\mathrm{context}}$: Derived from the Master for various context

Fig. 2 CanDID – A State-of-the-Art Approach toward DID

types ($\mathsf{DC}_{\mathsf{master}}$ and $\mathsf{DC}_{\mathsf{context}}$) is designed for supporting Sybil-resistance. CanDID assumes a trusted party of nodes, each of which may trust the others, and it is called the DID Committee.

The CanDID system protocols provide three APIs for issuing credentials, `issuePreCred`(), `issueMasterCred`(), and `issueCtxCred`(). `issuePreCred`() issues VC from a legacy credential issuer. CanDID supports deduplication of identities that may ensure the existence of at most one pseudonym with a unique identifier such as Social Security Number (SSN) in the U.S. For this, the master credential, generated by `issueMasterCred`(), includes a special attribute `dedupOver` that is designed to avoid deduplicating their identity by adversaries. Then, the holder may create a various verifiable credential depending on the context, derived from the master credential by `issueCtxCred`(). This scheme enables the system to issue credentials uniquely per user and meets Sybil-resistance.

They utilize multi-party computation (MPC) to prevent committee members from learning unnecessarily private information. They also utilize SNARK proofs for registration-time screening and other various purposes of privacy-preserving. They successfully demonstrated that the committee-based architecture achieves its goals with some particular purpose MPC protocols for privacy-preserving deduplication and fuzzy matching for scanning sanction lists to avoid AML.

3.2 The Formal Modeling of AESPs: $\mathcal{G}_{\mathtt{att}}$

According to Rafael Pass et al.s' efforts [18], the attested execution abstraction enables the following:

- A platform equipped with an attested execution processor can send a program and inputs, denoted $(\mathtt{prog}, \mathtt{inp})$, to its local secure processor. The secure processor executes the program over the inputs and computes $\mathtt{outp} := \mathtt{prog}(\mathtt{inp})$. The secure processor then signs the tuple $(\mathtt{prog}, \mathtt{outp})$ with a secret signing key[1] to obtain a digital signature σ_M, which is commonly referred to as an "attestation," and this entire execution is referred to as an "attested execution."
- The program's execution is conducted in a sandboxed environment (an enclave, in other words), so a software adversary and/or a physical adversary cannot tamper with the execution or inspect data that lives inside the enclave. This is important for realizing privacy-preserving applications.

The ideal functionality $\mathcal{G}_{\mathtt{att}}$ captures the core abstraction that a broad class of AESPs intends to provide. $\mathcal{G}_{\mathtt{att}}$ is parametrized with a signature scheme Σ and also a registry `reg` that is meant to capture all the platforms equipped with an AESP. The registry `reg` is treated as a static registry for simplicity in the research. $\mathcal{G}_{\mathtt{att}}$ consists of the initialization function to generate a key pair of the manufacturer public key $\mathtt{pk}_M$ and secret key $\mathtt{sk}_M$, public query interface $\mathsf{getpk}()$, and stateful enclave operations of $\mathsf{Install}()$ and $\mathsf{Resume}()$, which realize the anonymous attestation capability. A platform $\mathcal{P}$ that is in the registry `reg` may invoke those enclave operations[2].

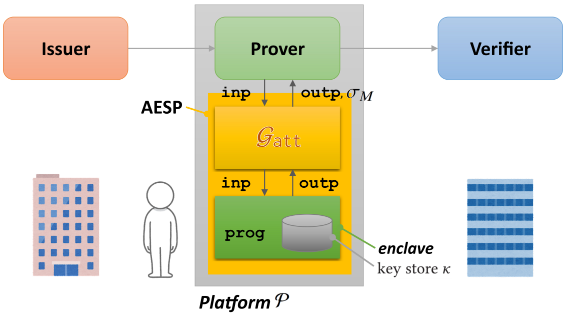


Fig. 3 An AESP and $\mathcal{G}_{\mathtt{att}}$ within a Platform $\mathcal{P}$

- *initialization:* $\Sigma.\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathtt{pk}_M, \mathtt{sk}_M)$.
- *public query interface:* $\mathsf{getpk}()$ from some $\mathcal{P}$: send $\mathtt{pk}_M$ to $\mathcal{P}$.
- *local interface – install an enclave:* $\mathsf{Install}()$ from some $\mathcal{P} \in \mathtt{reg}$: installing a new enclave with a program `prog`, henceforth referred to as the enclave program. Once installed, $\mathcal{G}_{\mathtt{att}}$ generates a fresh enclave identifier *eid* and returns it to $\mathcal{P}$.
- *local interface – resume an enclave:* $\mathsf{Resume}()$ from $\mathcal{P} \in \mathtt{reg}$: resuming the execution of an existing enclave with inputs `inp`. Once resumed, $\mathcal{G}_{\mathtt{att}}$ executes the `prog` (specified by *eid*) over the inputs `inp`, and obtains an output `outp`. $\mathcal{G}_{\mathtt{att}}$ would then sign the `prog` together with `outp` as well as additional metadata, and return both `outp` and the resulting anonymous attestation σ_M to $\mathcal{P}$.

Fig. 3 illustrates the relationship between an AESP and $\mathcal{G}_{\mathtt{att}}$ within a platform $\mathcal{P}$, which we may assume a mobile device for a prover, along with an issuer and a verifier. Once a program `prog` is installed and executed within an enclave, $\mathcal{G}_{\mathtt{att}}$ returns both `outp` and the anonymous attestation σ_M to $\mathcal{P}$. Since a platform $\mathcal{P}$ builts an AESP within it, we assume the AESP communicates with the platform $\mathcal{P}$ that is in `reg`.

4. Architecture and Protocols Proposal Overview

We propose architecture and system protocols to build a flexible, efficient, and secure SSI system by utilizing the formal abstraction of AESPs along with permissionless blockchain technology. Also, we would like to propose a design and construction for realizing a secure SSI to support Sybil-resistance based on the AESP-based SSI architecture.

[1] For brevity, we follow Rafael Pass et al.'s assumption where every instance of AESP shares the same signing key, so that no adversary can trace the originator of σ_M. See section 9.2 for detailed discussion.

[2] The notation here is modified from their original paper to adjust the following descriptions in this paper, but the meaning is equivalent.

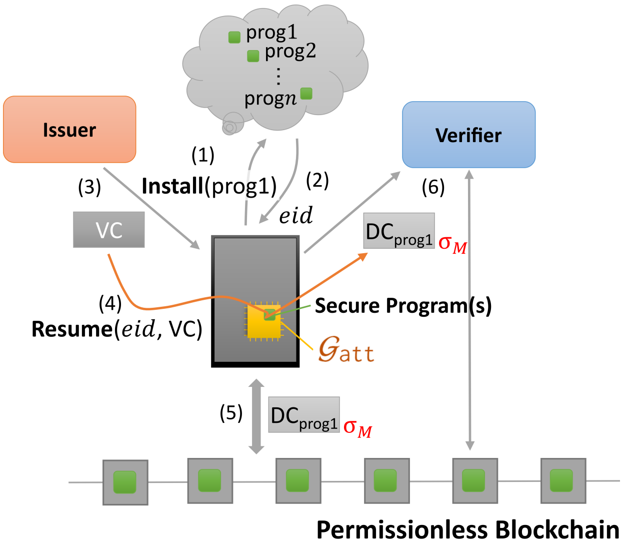


Fig. 4 Overview of the Proposed Architecture and the AESP Enclave Operations of Install() and Resume()

4.1 Architecture

Fig. 4 illustrates an overview of the architecture and how the basic AESP enclave operations of Install() and Resume() are integrated into the proposed architecture.

Overview of the main ideas are below:

- A person may have a mobile device equipped with an AESP, complying with the proposed AESP-based SSI architecture, which needs to be set up to make their device a Self-Sovereign Identity holder. This setup operation includes a device key pair $(\mathsf{pk}_M, \mathsf{sk}_M)$ generation.
- The person may install programs, $\mathtt{prog}_{1,\ldots,n}$, by Install() for enabling the device SSI-operations capable such as creating derived credentials ((1) in Fig. 4). For example, a prog is designed and implemented for minimizing disclosure of their original, verifiable credentials, less than 18 years old in particular. Once installed, *eid* is assigned for identifying the program prog to be executed by Resume() ((2) in Fig. 4).
- The person may ask authorities (Issuer) such as a governmental agency, a university, or other service providers to issue a verifiable credential consisting of claims and proof π for each claim ((3) in Fig. 4).
- Once the installed program is executed by Resume() ((4) in Fig. 4), and the AESP digitally signs an output outp to prove that the program has been executed on the specific AESP, and signed signature is attached with the output as a proof, σ_M. Before generating a derived credential, they should be allowed to produce a pairwise pseudonym for each entity E, one of which is Verifier; thus, their identity is to be represented with a key pair $(\mathsf{pk}_U^E, \mathsf{sk}_U^E)$. For simplicity, we will describe such a key pair like $(\mathsf{pk}_U, \mathsf{sk}_U)$ in this paper.

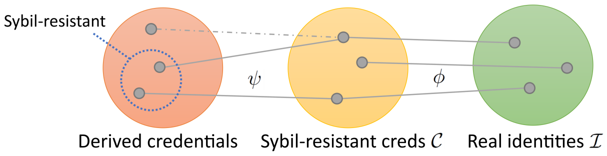


Fig. 5 Sybil-resistant derived credentials and the *identification map* ψ : $\mathtt{cred} \to C$

- Such verifiable credentials or derived credentials signed by the AESP with each proof are registered to a permissionless blockchain system as a repository ((5) in Fig. 4).
- Verifiers (Verifier) may utilize the signed credentials with a corresponding proof for each credential to verify if the person is requesting to subscribe and use services provided by the verifiers ((6) in Fig. 4).

In this proposal, the owner of a mobile device equipped with an AESP is the person who may represent their Self-Sovereign Identity. Because of utilizing AESPs, computation for preserving privacy can securely be executed within a device. In addition, verifiers may identify if the holder is the same person since the proof is attested by the holder's device equipped with an AESP. It means that MPC requiring a committee of trusted parties is not required, and permissionless blockchain can efficiently be utilized for openness.

4.2 Derived Credentials

Because of various needs, the proposed SSI architecture allows people to create programs for issuing derived credentials to meet different requirements. For example, some service providers need to verify if customers are not younger than 18 years old but do not need to know their birthdays. Some agencies need to verify if applicants are formally registered as residents in the city but do not need any other claims. For infinite varieties of needs to utilize derived credentials for presentation, which allows minimizing disclosure, and the programmable architecture enables users to choose appropriate prog for their needs. Those programs for the proposed SSI architecture must be public and open source for anyone to verify.

Derived Credentials for Sybil-Resistance

Unlike CanDID, the AESP-based SSI architecture does not assume generating the master credential, an interim credential designed to support the deduplication of identities for satisfying Sybil-resistance. An AESP is a unique entity capable of secure computation within a local processor. The equipped AESP may embed an encrypted link for derived credentials with a natural person by their key pair $(\mathsf{pk}_U, \mathsf{sk}_U)$. Fig. 5 illustrates the relationship among real identities $\mathcal{I}$, Symbil-resistant credentials C, and derived credentials some of which are Sybil-resistant.

Programs requiring to manage credentials that meet the Sybil-resistance requirement should implement a function of injective *identification map* ψ between Sybil-resistant derived credentials and C. The function ψ ensures the existence of at most one Sybil-resistant derived credential associated with a Sybil-resistant credential in C, and a derived credential with a dashed line in Fig. 5 is not Sybil-resistant. AESP may install and execute programs capable of treating ψ securely. The following section and Fig. 7 will describe the protocol for creating Sybil-resistant credentials.

5. Protocols in Detail

The proposed SSI architecture defines and provides some primitive protocols as described in Fig. 6 and Fig. 7. In our scheme, we assume EUF-CMA (Existential Unforgeability under Chosen Message Attack) signature scheme Σ and IND-CCA (Indistinguishability under Chosen Ciphertext Attack) encryption scheme $\{\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec}\}$. Further, we assume all AESP-equipped devices share $\mathtt{pk}_M$ and $\mathtt{sk}_M$ as determined in the Rafael Pass et al.'s works [18].

Definition 1 (A mobile device equipped with $\mathcal{G}_{\mathtt{att}}$)**.** *The ideal functionality of Attested Execution Secure Processors (AESPs) is denoted by $\mathcal{G}_{\mathtt{att}}$, and let us assume that every natural person's mobile device who needs SSI is equipped with $\mathcal{G}_{\mathtt{att}}$.*

Definition 2 (A secure SSI system protocols)**.** *A set of protocols Π is said to be secure Self-Sovereign Identity (SSI) system protocols if and only if it satisfies Sybil-resistance, Unforgebility, Privacy - credential-issuance and verification, and Unlinkability.*

Theorem 3 (The AESP-based secure SSI system protocols)**.** *Assuming that natural persons own their mobile devices equipped with $\mathcal{G}_{\mathtt{att}}$ and standard computational assumptions, a set of protocols $\Pi^{\mathcal{G}_{\mathtt{att}}}$ shown in Fig. 6 realizes a secure Self-Sovereign Identity (SSI) system protocols.*

The AESP-based SSI architecture and its protocol $\Pi^{\mathcal{G}_{\mathtt{att}}}$ includes the enclave operations of $\mathcal{G}_{\mathtt{att}}$, such as $\mathsf{Install}()$ and $\mathsf{Resume}()$ as well as for set-up, in addition to Π specific primitives such as for issuing and verifying credentials. As described in Section 4.2, the AESP-based SSI architecture and primitive protocols can be extended to adopt various requirements, including Sybil-resistance.

6. Security Analysis and Attacker Models

With respect to the CanDID's contributions [19], we will follow how CanDID demonstrates their protocols of decentralized identity systems are designed as securely as much as possible. In particular, they define CanDID API; in some of their definitions, adversaries have unlimited access to the entire CanDID API, which they model for conciseness as an oracle O^*. Also, in their security definitions, the adversaries may have access to an external account oracle $O^*_{\mathtt{ext}}$ that models the legacy providers called by CanDID.

AESP-based secure SSI system protocols – $\Pi^{\mathcal{G}_{\mathtt{att}}}$

The protocol $\Pi^{\mathcal{G}_{\mathtt{att}}}$ consists of two classes of primitive functions; one is a class that incorporates $\mathcal{G}_{\mathtt{att}}$, the ideal abstraction of AESPs. The other class is a set of primitive functions for SSI working with AESPs. $\Pi^{\mathcal{G}_{\mathtt{att}}}$ provides flexibility by allowing a natural person to choose and install programs $\mathtt{prog}$ for various needs.

The core abstraction of AESPs – the ideal functionality $\mathcal{G}_{\mathtt{att}}$:

- $\mathsf{Setup}(1^\lambda) \to (\mathtt{pk}_M, \mathtt{sk}_M)$.

  1: $\mathcal{G}_{\mathtt{att}}.\mathsf{KeyGen}(1^\lambda)$; // for generating a key pair.
  2: $\mathcal{G}_{\mathtt{att}}.\mathsf{getpk}()$. // for receiving the key pair from some platform $\mathcal{P}$, and sends $\mathtt{pk}_M$ to $\mathcal{P}$.

- $\mathsf{Install}(\mathtt{prog}) \to eid$. // install a program to enclave.

  1: $\mathcal{G}_{\mathtt{att}}$ asserts if $\mathcal{P}$ is honest;
  2: $\mathcal{G}_{\mathtt{att}}$ generates a nonce $eid \in \{0,1\}^\lambda$, stores the program $\mathtt{prog}$, and sends eid to $\mathcal{P}$.

- $\mathsf{Resume}(eid, \mathtt{inp}) \to (\mathtt{outp}, \sigma_M)$.

  1: $\mathcal{G}_{\mathtt{att}}$ checks if the program $\mathtt{prog}$ associated eid exists, abort if not found;
  2: $\mathcal{G}_{\mathtt{att}}$ executes $\mathtt{prog}$ and generates output $\mathtt{outp}$;
  3: $\mathcal{G}_{\mathtt{att}}$ generates a signature σ_M by $\Sigma.\mathsf{Sig}_{\mathtt{sk}_M}(eid, \mathtt{prog}, \mathtt{outp})$, and sends $(\mathtt{outp}, \sigma_M)$ to $\mathcal{P}$.

$\Pi^{\mathcal{G}_{\mathtt{att}}}$ secure SSI-featured basic functions accessing $\mathcal{G}_{\mathtt{att}}$:

- $\mathsf{KeyGen}(1^\lambda) \to (\mathtt{pk}^E_U, \mathtt{sk}^E_U)$.

  1: An AESP generates a user's key pair $(\mathtt{pk}^E_U, \mathtt{sk}^E_U)$, a pseudonym for each Entity. For simplicity, we omit E in the following descriptions.

- $\mathsf{IssueCred}(\mathtt{sk}_U, \mathtt{pk}_U, Stmt) \to \mathtt{cred}$.

  1: An AESP requests a legacy authority to issue their verifiable credential consisting of claims regarding $Stmt$;
  2: An AESP retrieves a verifiable credential from the authority and treats $\{\mathtt{pk}_U, (\mathtt{claim}_i)_{i=1,\ldots,n}, \pi\}$ as $\mathtt{cred}$, where π is a proof for a set of the claims by the authority.

- $\mathsf{IssueDCred}\left(\mathtt{sk}_M, \mathtt{sk}_U, \mathtt{pk}^{\mathtt{new}}_U, \mathtt{ctx}, \mathtt{cred}\right) \to \mathtt{derivedCred}$. This function, $\mathsf{IssueDCred}()$, is a program $\mathtt{prog}$, which can be vary for different context $\mathtt{ctx}$. To install and execute $\mathtt{prog}$,

  1: $\mathcal{G}_{\mathtt{att}}.\mathsf{Install}(\mathtt{prog}) \to eid$; // only once for install.
  2: $\mathcal{G}_{\mathtt{att}}.\mathsf{Resume}(eid, \mathtt{inp}) \to (\mathtt{outp}, \sigma_M)$.

  Inputs $\mathtt{inp}$ of $\mathtt{ctx}$ and $\mathtt{cred}$ are depend on various context specified by $\mathtt{ctx}$, outputs $\mathtt{outp}$ are $(\mathtt{claim}_j)_{j=1,\ldots,m}$ as a part of $\mathtt{derivedCred}$, where σ_M is $\Sigma.\mathsf{Sig}_{\mathtt{sk}_M}(eid, \mathtt{prog}, \mathtt{outp})$, and $\mathtt{prog}$ is an open-source program satisfying the following transformation:

  $$\mathtt{prog} : \{\mathtt{cred}_k\}_{k=1,\ldots,l} \mapsto \{\mathtt{claim}_j\}_{j=1,\ldots,m}$$

  For creating a Sybil-resistant credential, the program $\mathtt{prog}$ should satisfy the construction defined in Fig. 7.

- $\mathsf{VerifyCred}(\mathtt{sk}_U, \mathtt{cred}) \to \{\mathtt{true}, \mathtt{false}\}$.

  Two-party protocol between U and V with common input $\mathtt{pk}_M$. User U inputs $\mathtt{sk}_U$ and $\mathtt{cred}$, verifying party V authenticates U if U knows $\mathtt{sk}_U$ whose public key $\mathtt{pk}_U$ is on $\mathtt{cred}$ as follows,

  1: User U sends $(\mathtt{cred}, \sigma)$ to V where $\sigma = \mathsf{Sig}_{\mathtt{sk}_U}(c)$;
  2: Verifying party V checks if

$$\mathcal{V}_{\mathtt{pk}_M}(\mathtt{cred.body}, \mathtt{cred}.\sigma_M) = \mathtt{true} \wedge \mathcal{V}_{\mathtt{pk}_U}(c, \sigma) = \mathtt{true}.$$

Fig. 6 The Construction of $\Pi^{\mathcal{G}_{\mathtt{att}}}$, AESP-based SSI System Protocols

The Construction for creating Sybil-resistant credentials in $\Pi^{\mathcal{G}_{\mathtt{att}}}$

For creating a Sybil-resistant derived credential, the program $\mathtt{prog}$ should satisfy the following construction: the program $\mathtt{prog}$ treats $(\mathtt{pk}_U, \widehat{\psi})$ as inputs $\mathtt{inp}$, where $\widehat{\psi}$ is Sybil-resistant pseudonymizer to transform verifiable credentials to a set of claims satisfying the injective *identification map*

$$\psi : \mathtt{cred} \rightarrow C$$

in encrypted form. We require at least one verifiable credential, say $\mathtt{cred}_k$, which is a Sybil-resistant credential. We embed encrypted links using IND-CCA encryption algorithm $\mathcal{E}$:

$$\widehat{\psi} = \mathcal{E}.\mathrm{Enc}_{\mathtt{pk}_\mathrm{M}}(\mathtt{cred}_k)$$

The program $\mathtt{prog}$ decrypts $\widehat{\psi}$ to get ψ and checks if $\psi(\mathtt{cred}_k) \in C$.

The generated derived credential consists of $\mathtt{pk}_U$, $\widehat{\psi}$ as $\mathtt{prog}$, claims transformed by the Sybil-resistant pseudonymizer, together with the attestation signature σ_M from $\mathcal{G}_{\mathtt{att}}$ as follows:

$$\mathtt{derivedCred} \leftarrow (\mathtt{pk}_U, \widehat{\psi}, \{\mathtt{claim}_j\}_{j=1,\ldots,m}, \sigma_M)$$

To generate and treat derived credentials that are Sybil-resistant, we need to satisfy both **Definition 5** and the definitions for privacy at the same time. These requirements contradict each other. However, only the AESP can decrypt and verify links between the derived and Sybil-resistant credentials. Thus, we require all derived credentials to embed an encrypted link to one of those Sybil-resistant credentials in encrypted form.

Fig. 7 The Construction of the program for creating Sybil-resistant credentials in $\Pi^{\mathcal{G}_{\mathtt{att}}}$

We will reuse the same oracle models[1] for our AESP-based SSI system protocols $\Pi^{\mathcal{G}_{\mathtt{att}}}$. In our attacker models, we assume that all issuers and AESPs are honest; however, a holder (a natural person) who can be recognized as a prover or a verifier could be malicious. When a holder is malicious, the holder may attack their AESP to issue a wrong derived credential as an adversary $\mathcal{A}$ (*the malicious prover model*). Conversely, when a verifier is malicious, the verifier may violate holders' privacy (*the malicious verifier model*).

7. Security Properties

The set of protocols $\Pi^{\mathcal{G}_{\mathtt{att}}}$ aims to satisfy the following security properties, for each of which adversary may access and try to corrupt, Sybil-resistance, Unforgebility, Privacy - credential-issuance and verification, and Unlinkability.

7.1 Sybil-Resistance

An adversary cannot obtain Sybil-resistant credentials, which we define below:

Definition 4 (Sybil-resistant credential)**.** *Let $\mathcal{I}$ be a set of real identities and C be a set of credentials. The credentials C is said to be Sybil-resistant credentials if and only if there exists a bijective map $\phi : C \rightarrow \mathcal{I}$.*

In the real world, a national PKI system, e.g., JPKI (described in Section 9.1), is an example of authorities that can provide a unique identifier for creating Sybil-resistant credentials. A master credential in CanDID corresponds to a Sybil-resistant credential. We assume a single system of Sybil-resistant credentials for brevity in this paper.

Definition 5 (Existence)**.** *Suppose C be a set of all Sybil-resistant credentials. A derived credential $\mathtt{cred}$ is said to be Sybil-resistant with respect to C if and only if, for any PPT (Probabilistic Polynomial-Time) adversary $\mathcal{A}$ and security parameter λ, there exists an identification map $\psi : \mathtt{cred} \rightarrow C$ only with negligible error probability, namely:*

$$\Pr\left[\psi(\mathtt{cred}) \in C \;\middle|\; \begin{array}{c} \mathtt{pk}_M, \mathtt{sk}_M \leftarrow \mathsf{KeyGen}(1^\lambda); \\ \mathtt{cred} \leftarrow \mathcal{A}^{O^*, O^*_{\mathtt{ext}}}(\mathtt{pk}_M); \\ \mathcal{V}_{\mathtt{pk}_M}(\mathtt{cred.body}, \mathtt{cred.}\sigma) = \mathtt{true} \end{array}\right] \geq 1 - \mathrm{negl}(\lambda)$$

Informally, this definition captures the infeasibility of an adversary to obtain a derived credential $\mathtt{cred}$ that is not in the set of all Sybil-resistant credentials such that $\psi(\mathtt{cred}) \in C$ as far as $\mathtt{cred}$ bears a valid attestation signature, namely, $\mathcal{V}_{\mathtt{pk}_M}(\mathtt{cred.body}, \mathtt{cred.}\sigma) = \mathtt{true}$. Here, the *identification map* ψ is defined over all elements in derived credentials such that $\psi(\mathtt{cred}) \in C$. Thus, ψ uniquely 'identifies' the holders' real identity from anonymous derived credentials. In our scheme, we assume the map ψ is accessed only by the AESP internally. Thus, the link between derived credentials and the Sybil-resistant credentials is hidden; it supports preserving privacy.

This game resides on *the malicious prover model* in our attacker models. An adversary $\mathcal{A}$ attacks the holder to create potentially a wrong derived credential under the assumption that $\mathcal{V}_{\mathtt{pk}_M}$ is honest.

7.2 Unforgeability

An adversary cannot forge the credentials of honest users or otherwise impersonate them.

Definition 6 (Unforgeability)**.** *Let $\mathtt{chals}$ denote a set of all challenges and their responses produced by $\mathcal{A}$ in oracle access with O^* and a special oracle $O^*_{\mathtt{sk}_U}$ that allows calling any $\Pi^{\mathcal{G}_{\mathtt{att}}}$ functions with the user key parameter set to $\mathtt{sk}_U$. The protocol $\Pi^{\mathcal{G}_{\mathtt{att}}}$ offers unforgeability if, for any stateful PPT adversary $\mathcal{A}$,*

[1] Following the conventions in CanDID [19], O^* has the same functions(APIs) as Fig. 6 but acts honestly as an ideal functionality. $O^*_{\mathtt{ext}}$ has its internal state L, where L is a set of tuples of the form (id, a, v) where id is an user identifier, a an attribute, and v the corresponding value. $O^*_{\mathtt{ext}}$ has the following functions with initial state $L = \emptyset$:

1. $\mathsf{update}(id, a, v')$: if $\exists(id, a, v) \in L$, replace it with (id, a, v').
2. $\mathsf{delete}(id)$: Remove all $(id, \_, \_)$ from L if exist.
3. $\mathsf{getProof}(id, a) \rightarrow v, \pi$: If $\exists(id, a, v) \in L$, return v with a proof π, or $\perp$ otherwise.
4. $\mathsf{getOwnershipProof}(id) \rightarrow \pi$: If $\exists(id, \_) \in L$, return a proof of account ownership, or $\perp$ otherwise.

$$\Pr\left[\begin{array}{c|c} \mathsf{VerifyCred}(sk_U,\mathtt{cred}) & \mathtt{pk}_M,\mathtt{sk}_M\leftarrow\mathsf{KeyGen}(1^\lambda); \\ =\mathtt{true} & \mathtt{pk}_U,\mathtt{sk}_U\leftarrow\mathsf{KeyGen}(1^\lambda); \\ & \mathtt{cred}\leftarrow\mathcal{A}^{\mathcal{O}^*,\mathcal{O}^*_{\mathtt{sk}_U},\mathcal{O}^*_{\mathtt{ext}}}(\mathtt{sk}_M,\mathtt{pk}_U) \\ & \textit{s.t. }\mathtt{cred.body}\notin\mathtt{chals}; \end{array}\right] \le \mathrm{negl}(\lambda)$$

The definition captures that it must be infeasible for an adversary to impersonate users, i.e., forge signatures with users' keys. This game also resides on *the malicious prover model* where an adversary $\mathcal{A}$ attacks the holder to potentially create a wrong credential under the assumption that the verifier is honest.

7.3 Privacy - Credential-Issuance

It is infeasible for an adversary to learn users' attributes from observing the derived credential-issuance protocol.

Definition 7 (Credential issuance privacy)**.** *The protocol $\Pi^{\mathcal{G}_{\mathtt{att}}}$ offers derived credential issue privacy if, for any stateful PPT adversary $\mathcal{A}$,*

$$\left|\Pr\left[b=b' \,\middle|\, \begin{array}{c} \mathtt{pk}_M,\mathtt{sk}_M\leftarrow\mathsf{KeyGen}(1^\lambda); \\ \mathtt{pk}_U,\mathtt{sk}_U,\{c^0_1,\ldots,c^0_l\},\{c^1_1,\ldots,c^1_l\}\leftarrow\mathcal{A}^{\mathcal{O}^*,\mathcal{O}^*_{\mathtt{ext}}}(\mathtt{pk}_M);^{\S} \\ \mathtt{cred}^0\leftarrow\mathsf{IssueDCred}(\mathtt{sk}_M,\mathtt{sk}_U,\mathtt{pk}_U,\{c^0_1,\ldots,c^0_l\},\mathtt{prog}), \\ \mathtt{cred}^1\leftarrow\mathsf{IssueDCred}(\mathtt{sk}_M,\mathtt{sk}_U,\mathtt{pk}_U,\{c^1_1,\ldots,c^1_l\},\mathtt{prog}) \\ \textit{where } \mathtt{cred}^0=(\mathtt{pk}_U,\{\mathtt{claim}^0_j\}_{j=1,\ldots,m},\widehat{\psi^0},\mathtt{prog},\sigma^0_M) \\ \textit{and } \mathtt{cred}^1=(\mathtt{pk}_U,\{\mathtt{claim}^1_j\}_{j=1,\ldots,m},\widehat{\psi^1},\mathtt{prog},\sigma^1_M); \\ \text{assert } \{\mathtt{claim}^0_j\}_{j=1,\ldots,m}=\{\mathtt{claim}^1_j\}_{j=1,\ldots,m} \textit{ as sets}; \\ b\leftarrow\$\{0,1\}; \\ b'\leftarrow\mathcal{A}^{\mathcal{O}^*,\mathcal{O}^*_{\mathtt{ext}}}(\mathtt{cred}^b) \end{array}\right] - \frac{1}{2}\right| \le \mathrm{negl}(\lambda)$$

[§] *where $\{\mathtt{cred}^0_k\}_{k=1,\ldots,l}$ is denoted $\{c^0_1,\ldots,c^0_l\}$ and $\{\mathtt{cred}^1_k\}_{k=1,\ldots,l}$ is denoted $\{c^1_1,\ldots,c^1_l\}$ as a set of claims for each $\{0,1\}$*

This game resides on *the malicious verifier model* in our attacker models. An adversary $\mathcal{A}$ tries to violate a holder's privacy by retrieving information from their credential, assuming that the holder is honest.

7.4 Privacy - Credential-Verification

An adversary can learn about a user no more than the information they explicitly present while using their credentials.

Definition 8 (Credential verification privacy)**.** *Given an open-source map* $\mathtt{prog}$ *that maps user data in verifiable credentials to derived credential claims, any PPT adversary $\mathcal{A}$ learns negligibly more about any given user than the output of* $\mathtt{prog}$.

7.5 Unlinkability

The entities administering the protocol $\Pi^{\mathcal{G}_{\mathtt{att}}}$ reliant programs cannot collude and link the respective transactions of any given user.

Definition 9 (Unlinkability across programs)**.** *The protocol $\Pi^{\mathcal{G}_{\mathtt{att}}}$ offers unlinkability if, for any stateful PPT adversary $\mathcal{A}$,*

$$\left|\Pr\left[b=b' \,\middle|\, \begin{array}{c} \mathtt{pk}_M,\mathtt{sk}_M\leftarrow\mathsf{KeyGen}(1^\lambda); \\ \mathtt{cred}^0,\mathtt{cred}^1,\mathtt{pk}_U,\mathtt{sk}_U,\mathtt{ctx}\leftarrow\mathcal{A}^{\mathcal{O}^*,\mathcal{O}^*_{\mathtt{ext}}}(\mathtt{pk}_M); \\ \text{assert } \mathcal{V}_{\mathtt{pk}_U}(\mathtt{cred}^b.\mathtt{body},\mathtt{cred}^b.\sigma)=\mathtt{true} \textit{ for } b=0,1; \\ b\leftarrow\$\{0,1\}; \\ \mathtt{cred}_{new}\leftarrow\mathsf{IssueDCred}(\mathtt{sk}_M,\mathtt{sk}_U,\mathtt{pk}_U,\mathtt{cred}^b); \\ b'\leftarrow\mathcal{A}^{\mathcal{O}^*,\mathcal{O}^*_{\mathtt{ext}}}(\mathtt{cred}_{new},\mathtt{ctx}) \end{array}\right] - \frac{1}{2}\right| \le \mathrm{negl}(\lambda)$$

This game also resides on *the malicious verifier model* in our attacker models. An adversary $\mathcal{A}$ tries to violate a holder's privacy by retrieving information from their new credential, assuming that the holder is honest.

8. A Proof Sketch of the Security Properties

Proof Sketch of Theorem 3. We prove that the protocol $\Pi^{\mathcal{G}_{\mathtt{att}}}$ defined in Fig. 6 and Fig. 7, which is a set of secure Self-Sovereign Identity (SSI) system protocols.

8.1 Sybil-Resistance

First, we prove $\Pi^{\mathcal{G}_{\mathtt{att}}}$ satisfies **Definition 5** for **Existence**. It is sufficient to prove that every derived credential $\mathtt{cred}$ has an *identification map* ψ such that $\psi(\mathtt{cred}) \in C$. In the protocol $\Pi^{\mathcal{G}_{\mathtt{att}}}$, every $\mathtt{cred}$ has the following form

$$(\mathtt{pk}_U, \widehat{\psi}, \{\mathtt{claim}_j\}_{j=1,\ldots,m}, \sigma_M)$$

where $\widehat{\psi}$ is a ciphertext of a verifiable and Sybil-resistant credential $\mathtt{cred}$ encrypted with the public key of $\mathcal{G}_{\mathtt{att}}$. Therefore, given

$$\mathcal{V}_{\mathtt{pk}_M}(\mathtt{cred.body}, \mathtt{cred}.\sigma_M) = \mathtt{true} \Rightarrow$$
$$\mathcal{V}_{\mathtt{pk}_M}\left(\mathtt{pk}_U, \widehat{\psi}, \{\mathtt{claim}_j\}_{j=1,\ldots,m}, \sigma_M\right) = \mathtt{true},$$

it implies that $\mathcal{G}_{\mathtt{att}}$ can decrypt $\widehat{\psi}$ to get $\mathtt{cred}$ as $\mathtt{cred} = \mathcal{E}.\mathrm{Dec}_{\mathtt{sk}_M}(\widehat{\psi})$ and verify the relation $\mathtt{cred} \in C$ unless the signature σ_M is forged. $\mathcal{E}$ denotes IND-CCA encryption scheme. The latter probability is negligible in λ given Σ is the EUF-CMA signature scheme.

8.2 Unforgeability

In $\Pi^{\mathcal{G}_{\mathtt{att}}}$ based SSI systems, users' key never leaves their device with an AESP. During the protocols, they use it only to sign challenges issued as part of $\mathsf{VerifyCred}()$. Thus, unforgeability of the $\Pi^{\mathcal{G}_{\mathtt{att}}}$ based SSI systems follows in a straightforward way.

Here, $\mathtt{cred}$ has the following form:

$$(\mathtt{pk}_U, \widehat{\psi}, \{\mathtt{claim}_j\}_{j=1,\ldots,m}, \sigma_M).$$

Queries to $\mathcal{O}^*$ and $\mathcal{O}^*_{\mathtt{sk}_U}$ must be a set of tuples

$$(\mathtt{pk}_U, \{\mathtt{cred}_k\}_{k=1,\ldots,l}, \mathtt{prog})$$

and the responses are $(\mathtt{pk}_U, \widehat{\psi}, \{\mathtt{claim}_j\}_{j=1,\ldots,m}, \sigma_M)$ where a set of claims $\{\mathtt{claim}_j\}_{j=1,\ldots,m}$ is the image of $\mathtt{prog}$ with inputs $\{\mathtt{cred}_k\}_{k=1,\ldots,l}$. Thus, $\mathtt{chals}$ contains all tuples appeared in the oracle access by $\mathcal{A}$ of the form $(\mathtt{pk}_U, \widehat{\psi}, \{\mathtt{claim}_j\}_{j=1,\dots,m})$. For creating new cred such that $\mathtt{cred.body} \notin \mathtt{chals}$, $\mathcal{A}$ must forge a signature $\mathtt{cred}.\sigma$ on the message tuple cred.body. Given the underlying EUF-CMA signature scheme Σ, this probability is bounded by $\mathrm{negl}(\lambda)$, which is negligible in the security parameter λ.

8.3 Privacy - Credential-Issuance

In our privacy game for privacy - credential-issuance, the adversary chooses a pseudonym of the user who initiates each query and which providers are used but otherwise learns nothing else about users' identities or attributes during operations such as credentials issuance.

By **Definition 7**, the adversary chooses two identities of $\{0, 1\}$ and observes that derived credentials are created by executing $\mathsf{IssueDCred}()$ with inputs of claims for each identity and the program prog with the encrypted *identification map* $\widehat{\psi}$. The adversary tries to access and guess any attributes and/or values; however, they cannot guess from a derived credential selected randomly.

Let us explain the reason behind it more. Since two credentials, $\mathtt{cred}^0$ and $\mathtt{cred}^1$ only differ in $\widehat{\psi}^0$, $\widehat{\psi}^1$ and related signatures, σ_U^0 and σ_U^1. $\widehat{\psi}^0$ and $\widehat{\psi}^1$ are encrypted by the IND-CCA encryption algorithm $\mathcal{E}$. Probability to distinguish them is upper-bounded $\mathrm{negl}(\lambda)$. Therefore, we conclude that the adversary cannot win the game as it does not learn any information to distinguish the verifiable credentials.

8.4 Privacy - Credential-Verification

In our scheme, we assume that all privacy operations for issuing and treating credentials are executed within an AESP internally by prog, including $\mathsf{IssueDCred}()$ and $\mathsf{VerifyCred}()$. We also expect that only prog will be accepted by users and providers who reach the consensus. Such prog only leaks required privacy information described as a set of claims $\{\mathtt{claim}_j\}_{j=1,\dots,m}$. This process is expected to leak any more information as defined in **Definition 8**.

8.5 Unlinkability

As the same as the other privacy game for privacy - credential issuance, the adversary needs to try an input but randomly selected, and a credential $\mathtt{cred}^0$ or $\mathtt{cred}^1$ in this case as defined in **Definition 9**. It cannot guess any information to distinguish which provider from a credential selected randomly. Therefore, we conclude that the adversary cannot win the game of unlinkability in our scheme.

□

9. Applications, Limitations and Future Directions

This proposal has many opportunities and applications in the real world because of the rapid increase of smartphones and other mobile devices equipped with a tamper-resistant secure processor in the market. Permissionless blockchains in this proposal play a role in building SSI systems as a foundation. Like previous research and implementations, they work for verifiable data registries; however, the use is not limited to storing and retrieving verifiable and derived credentials as a registry. In addition, combining permissionless blockchains and AESPs may extend the usage.

For instance, secure programs for creating derived credentials by $\mathsf{IssueDCred}()$, which allows for a user to choose a program prog depending on different context ctx, can and should probably be registered and maintained on the permissionless blockchain. Also, derived credentials created by the user's device with an AESP may represent the person on permissionless blockchain ecosystems, preserving privacy. Because of the recent rapid growth, opportunities to utilize the main idea of this proposal to combine permissionless blockchains and AESPs are unlimited.

9.1 Applications

One of the well-known initiatives is mDL, mobile driver's license[1]. It must be helpful, but people would not always be happy to show their driver's license even though it allows them to choose to display only requested data such as name and age. The AESP-based SSI architecture and protocols will enable service providers to create programs that request only they need to verify; it encourages their users to contact them without hesitating to disclose unnecessary information. More importantly, people on the planet will gain Self-Sovereign Identity consisting of the ten principles, including existence and control.

My Number Individual Card and JPKI

The Japanese government has taken the initiative to support enabling JPKI, a part of My Number Individual Card capabilities, on smartphones. For this, they have utilized Global Platform-supported Secure Elements. The goals of the initiative and ideas of Self-Sovereign Identity are not identical; however, there are many analogies between them. Duplicated certificates and key pairs are securely stored in the device, and it may work for their identity proofing or verifying claims. The result of JPKI can also be used to create a Sybil-resistant credential. Future extensions of real-world identity-related initiatives toward Self-Sovereign Identity are very expected.

9.2 Limitations and Future Directions

We want to describe two problems that remain and will be addressed to solve in our future work.

Addressing Complexity in the Real World

We propose to incorporate Rafael Pass et al.'s contribution

[1] https://www.aamva.org/Mobile-Drivers-License/

regarding the formal abstraction of AESPs [18] to build an SSI system. Also, we have demonstrated our ideas of protocols, security properties, and a proof sketch of the security properties in an informal fashion; however, we made some assumptions for brevity, such as a single system of Sybil-resistant credentials. Further research is expected to address more complexity existing in the real world.

Potential Vulnerability of Hardware-Assisted Security

Some readers might be concerned about the vulnerability of tamper-resistant secure processors to compromise. We plan to address defining a threat model to cover such vulnerability. One of those is the globally shared key pair of $\mathtt{pk}_M$ and $\mathtt{sk}_M$, which is possibly an obvious target for compromise; however, we believe that previous research addressing anonymous attestation may resolve the concern. Ernie Brickell et al. proposed Direct Anonymous Attestation (DAA) scheme, as well as its enhancements known as Enhanced Privacy ID (EPID) to address the problem [35][36][37][38], firstly adopted onto TPM. Further, Christina Garman et al.'s contributions proposed decentralized direct anonymous attestation [39]. Taisei Takahashi et al. addressed the same problem possibly happening when double spending in their scenario and utilized Brickell et al.'s approach [40].

Practice – Reference Implementation

We have not implemented the proposed architecture and the system protocol in our work. Some existing SSI systems are already deployed utilizing permissionless blockchain, and we plan to design a prototype of the proposed architecture over the existing permissionless blockchain systems such as Ethereum 2.0[1].

The further detailed design will include *a.*) interface between issuers/verifiers and permissionless blockchain for a natural person who owns a mobile device equipped with an AESP to control their credentials, *b.*) open-source programs that will be installed and executed on the AESP, and *c.*) public verification capability that eliminates the strong assumption of having AESPs from verifiers.

10. Conclusion

We have demonstrated the powerfulness of hardware-assisted security and the formal abstraction of Attested Execution Secure Processors (AESPs) over permissionless blockchain technology. Based on those techniques, we proposed the AESP-based secure Self-Sovereign Identity (SSI) architecture and system protocols $\Pi^{\mathcal{G}_{\mathtt{att}}}$ along with security properties including Sybil-resistance and a proof sketch of the security properties.

Assuming AESPs and $\mathcal{G}_{\mathtt{att}}$, the AESP-based SSI system protocols $\Pi^{\mathcal{G}_{\mathtt{att}}}$ eliminates the online distributed committee of trusted nodes assumed in CanDID; thus, $\Pi^{\mathcal{G}_{\mathtt{att}}}$ allows not to rely on multi-party computation (MPC), and it brings drastic flexibility and efficiency when compared with the existing systems. In addition, we described applications, limitations, and our future directions in this work.

Acknowledgments

The authors thank all researchers and contributors for digital identity and mobile. Also, we very much appreciate all anonymous reviewers' feedback to improve a draft of this paper; their comments have encouraged us.

References

[1] K. Moriyama and A. Otsuka, "Permissionless Blockchain-Based Sybil-Resistant Self-Sovereign Identity Utilizing Attested Execution Secure Processors," IEEE Blockchain 2022, Espoo, Finland, pp.1–10, IEEE, Aug. 2022.

[2] D. Chaum, "Security Without Identification: Transaction Systems to Make Big Brother Obsolete," Communications of the ACM, vol.28, no.10, pp.1030–1044, 1985.

[3] A. Preukschat and D. Reed, Self-Sovereign Identity, Decentralized Digital Identity and Verifiable Credentials, Manning Publications Co., Shelter Island, NY, USA, May 2021.

[4] C. Allen, "The Path to Self-Sovereign Identity," April 2016.

[5] P. Dunphy and F. A. P. Petitcolas, "A First Look at Identity Management Schemes on the Blockchain," IEEE Security & Privacy, vol.16, no.4, pp.20–29, Jan. 2018.

[6] M.S. Ferdous, F. Chowdhury, and M. O. Alassafi, "In Search of Self-Sovereign Identity Leveraging Blockchain Technology," IEEE Access, vol.7, pp.103059–103079, Aug. 2019.

[7] N. Naik and P. Jenkins, "Self-Sovereign Identity Specifications: Govern Your Identity Through Your Digital Wallet using Blockchain Technology," 2020 8th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud), Oxford, UK, pp.90–95, IEEE, Aug. 2020.

[8] G. Kondova and J. Erbguth, "Self-sovereign identity on public blockchains and the GDPR," Proceedings of the 35th Annual ACM Symposium on Applied Computing, New York, NY, USA, pp.342–345, ACM, March 2020.

[9] C. Gomez Munoz, "eIDAS Supported Self-Sovereign Identity," tech. rep., European Commission, May 2019.

[10] A. Mühle, A. Grüner, T. Gayvoronskaya, and C. Meinel, "A Survey on Essential Components of a Self-Sovereign Identity," July 2018. arXiv:1807.06346.

[11] Q. Stokkink and J. Pouwelse, "Deployment of a Blockchain-Based Self-Sovereign Identity," 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), pp.1336–1342, June 2018.

[12] D. van Bokkem, R. Hageman, G. Koning, L. Nguyen, and N. Zarin, "Self-Sovereign Identity Solutions: The Necessity of Blockchain Technology," April 2019. arXiv:1904.12816.

[13] N. Naik and P. Jenkins, "uPort Open-Source Identity Management System: An Assessment of Self-Sovereign Identity and User-Centric Data Platform Built on Blockchain," 2020 IEEE International Symposium on Systems Engineering (ISSE), Vienna, Austria, pp.1–7, IEEE, Nov. 2020.

[14] P. J. Windley, "Sovrin: An Identity Metasystem for Self-Sovereign Identity," Frontiers in Blockchain, vol.4, July 2021.

[15] V. Costan, I. Lebedev, and S. Devadas, "Sanctum: Minimal Hardware Extensions for Strong Software Isolation," 25th USENIX Security Symposium (USENIX Security '16), pp.857–874, Aug. 2016.

[16] E. Shi, F. Zhang, R. Pass, S. Devadas, D. Song, and C. Liu, "Trusted

[1] `https://ethereum.org/en/eth2/`

Hardware: Life, the Composable Universe, and Everything (2015 12 15 4 Elaine Shi, et el.)," May 2017.

[17] K. C. Torh and A. Anderson-Priddy, "Self-Sovereign Digital Identity: A Paradigm Shift for Identity," IEEE Security & Privacy, vol.17, pp.17–27, May 2019.

[18] R. Pass, E. Shi, and F. Tramèr, "Formal Abstractions for Attested Execution Secure Processors," Advances in Cryptology – EUROCRYPT 2017, LNCS, vol.10210, Paris, France, pp.260–289, Springer, April 2017.

[19] D. Maram, H. Malvai, F. Zhang, N. Jean-Louis, A. Frolov, T. Kell, T. Lobban, C. Moy, A. Juels, and A. Miller, "CanDID: Can-Do Decentralized Identity with Legacy Compatibility, Sybil-Resistance, and Accountability," 2021 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, pp.1348–1366, IEEE, May 2021.

[20] K. Cameron, "The Laws of Identity," May 2005.

[21] ISO/IEC, "ISO/IEC 24760-1:2019/Amd 1:2023 (en), IT Security and Privacy — A framework for identity management — Part 1: Terminology and concepts," Jan. 2023.

[22] P. A. Grassi, M. E. Garcia, and J. L. Fenton, "NIST Special Publication 800-63-3 Digital Identity Guidelines," June 2017.

[23] M.S. Ferdous, G. Norman, A. Jøsang, and R. Poet, "Mathematical Modelling of Trust Issues in Federated Identity Management," IFIPTM 2015: Trust Management IX, pp.13–29, April 2015.

[24] W3C, "Decentralized Identifiers (DIDs) v1.0 (W3C Recommendation)," July 2022.

[25] N. Santos, H. Raj, S. Saroiu, and A. Wolman, "Using ARM trustzone to build a trusted language runtime for mobile applications," Proceedings of the 19th international conference on Architectural support for programming languages and operating systems, Salt Lake City, UT, USA, pp.67–80, ACM, Feb. 2014.

[26] A. Satybaldy, M. Nowostawski, and J. Ellingsen, "Self-Sovereign Identity Systems," 14th IFIP International Summer School on Privacy and Identity Management (Privacy and Identity), Windisch, Switzerland, pp.447–461, Aug. 2019.

[27] Y. Liu, Q. Lu, H. Paik, and X. Xu, "Design Patterns for Blockchain-based Self-Sovereign Identity," Proceedings of the European Conference on Pattern Languages of Programs 2020, Virtual Event Germany, pp.1–14, July 2020.

[28] A. Grüner, A. Mühle, and C. Meinel, "On the Relevance of Blockchain in Identity Management," July 2018. arXiv:1807.08136.

[29] A.E. Panait, R. F. Olimid, and A. Stefanescu, "Analysis of uPort Open, an Identity Management Blockchain-Based Solution," Trust, Privacy and Security in Digital Business, LNCS, vol.12395, Bratislava, Slovakia, pp.3–13, Springer, Sept. 2020.

[30] S. Mahula, E. Tan, and J. Crompvoets, "With blockchain or not? Opportunities and challenges of self-sovereign identity implementation in public administration: Lessons from the Belgian case," DG.O2021: The 22nd Annual International Conference on Digital Government Research, Omaha, NE, USA, pp.495–504, ACM, June 2021.

[31] D. Du Seuil, "European Self Sovereign identity framework," July 2019.

[32] A. Boysen, "Decentralized, Self-Sovereign, Consortium: The Future of Digital Identity in Canada," Boysen, vol.4, no.624258, p.8, April 2021.

[33] W3C, "Verifiable Credentials Data Model 1.1 (W3C Recommendation)," March 2022.

[34] John. R. Douceur, "The Sybil Attack," 1st International Workshop on Peer-to-Peer Systems (IPTPS 2002), LNCS, vol.2429, Cambridge, MA, USA, pp.251–260, Springer, March 2002.

[35] E. Brickell, J. Camenisch, and L. Chen, "Direct Anonymous Attestation," Proceedings of the 11th ACM Conference on Computer and Communications Security - CCS '04, Washington DC, USA, pp.132–145, ACM, Oct. 2004.

[36] E. Brickell and J. Li, "Enhanced Privacy ID: A Direct Anonymous Attestation Scheme with Enhanced Revocation Capabilities," Proceedings of the 2007 ACM workshop on Privacy in electronic society, Alexandria, VA, USA, pp.21–30, Oct. 2007.

[37] E. Brickell and J. Li, "Enhanced Privacy ID from Bilinear Pairing," March 2009. Cryptology ePrint Archive, Paper 2009/095.

[38] E. Brickell and J. Li, "Enhanced Privacy ID from Bilinear Pairing for Hardware Authentication and Attestation," 2010 IEEE Second International Conference on Social Computing, Minneapolis, MN, USA, pp.768–775, Aug. 2010.

[39] C. Garman, M. Green, and I. Miers, "Decentralized Anonymous Credentials," Oct. 2013.

[40] T. Takahashi, T. Higuchi, and A. Otsuka, "VeloCash: Anonymous Decentralized Probabilistic Micropayments With Transferability," IEEE Access, vol.10, pp.93701–93730, Sept. 2022.

Koichi Moriyama received B.E. degree in Electrical Engineering and M.S. degree in Computer Science from Keio University in 1992 and 1994, respectively. He stayed at College of Computing, Georgia Institute of Technology as a research scholar in 1999–2000, while he worked for Sony Corporation and Sony Ericsson Mobile Communications until March 2011. He is Chief Security Architect at NTT DOCOMO, INC. and is a Board of Directors at FIDO Alliance, Inc. and W3C, Inc. for both, today. He is now also a doctoral course student at Graduate School of Information Security, Institute of Information Security. He is a member of IEICE, ACM, and IEEE.

Akira Otsuka received B.E. and M.E. from Osaka University in 1989 and 1991 respectively, and Ph.D. degree from the University of Tokyo in 2002. From 2002, he was a Post Doctoral Fellow and a Cooperative Researcher at the University of Tokyo. From 2005, he was with National Institute of Advanced Industrial Science and Technology (AIST), serves as a Leader of Research Security Fundamentals during 2006 to 2010. During 2007 to 2014, he was a Visiting Professor at Research and Development Initiative, Chuo University. From 2017, he is a Professor at Graduate School of Information Security, Institute of Information Security. He was a Visiting Researcher, Financial Research Institute, Bank of Japan (2020-2021). He is a senior member of IEICE and IPSJ, and a member of JSAI, IEEE, IACR, and IFCA.