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

Koichi Moriyama†
Institute of Information Security
Yokohama, Japan
moriyama@ai.iisec.ac.jp

Akira Otsuka
Institute of Information Security
Yokohama, Japan
otsuka@iisec.ac.jp

***Abstract*—The current circumstance that requires people more online has encouraged us to address digital identity preserving privacy. There is a momentum of research addressing Self-Sovereign Identity (SSI); many research approach blockchain technology as a foundation. SSI brings natural persons various benefits such as their 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. Our proposal in this paper is to build a secure SSI system by utilizing permissionless blockchain and Rafael Pass et al.'s contribution of the formal abstraction of Attested Execution Secure Processors (AESPs). Our proposal of the AESP-based SSI architecture and system protocols, $\Pi^{\mathcal{G}_{att}}$, demonstrates the powerfulness of hardware-assisted security and the formal abstraction of AESPs, fitting into building a proper SSI system that satisfies Sybil-resistance. Assuming AESPs and $\mathcal{G}_{att}$, the protocols may eliminate the online distributed committee assumed in other research such as CanDID; thus, $\Pi^{\mathcal{G}_{att}}$ allows not to rely on multi-party computation (MPC), and it brings drastic flexibility and efficiency compared with the existing SSI systems.**

***Index Terms*—Permissionless Blockchain, Decentralized Digital Identity, Sybil-Resistance, Self-Sovereign Identity, and Attested Execution Secure Processors**

I. Introduction

Increasing attention to digital identity is a natural trend in the current Covid-19 circumstances because people face rapid changes in our lives, requiring much more online than before, and identifying and verifying who I am has become fundamental and daily operations under the new normal. Self-Sovereign Identity (SSI) is the momentum in academia and the tech industry. Christoper Allen expressed the history of digital identity and the expectation to SSI well in his blog article in 2016 [1]. Since then, there have been many studies, researches, and implementations until now [2] [3] [4].

The terminology "Self-Sovereign" inspires many people to think about how SSI can protect the privacy and resolve reliance on authorities that may control personal data. Studies and research on SSI are not limited to technology but also government and human beings. One of those researches addresses the relationship between SSI and GDPR [5], while there are already some initiatives on utilizing SSI in Europe [6]. These trends remind the authors of David Chaum's approach in 1985 to avoid unexpected tracing by someone else like Big Brother[1] by utilizing pseudonyms, digital signatures, and card computers [7].

If we look at such many of those recent studies, researches, and implementations, they focus on or are utilizing distributed ledger and blockchain technology [8] [9]. Some researchers addressed the necessity of blockchain; however, they still recognize that blockchain technology is a good foundation [10]. All the well-known existing implementations actually utilize blockchains, such as uPort on Ethereum[2] [11], ShoCard[3] on Bitcoin [12], and Sovrin on the Sovrin ledger [13].

Many pieces of research in this domain have addressed SSI systems architecture. However, surprisingly – to the best of our knowledge, no study has addressed an opportunity to utilize hardware-assisted security [14] [15] implemented within mobile devices that people own for their daily lives. Several studies and implementations address mobile apps for SSI systems, but those mobile apps focus on user experiences and have never addressed security feature perspectives. The card computer expressed in 1985 was a kind of dream written as a vision; however, it has become real, and secure processors are also becoming a norm in such mobile devices as a mandatory requirement today. Why don't we utilize such capability for building SSI systems?

This paper proposes a permissionless blockchain-based SSI systems architecture that utilizes the formal abstraction of Attested Execution Secure Processors (AESPs) [16] equipped with mobile devices. Thus, SSI systems will gain more flexibility and efficiency than existing ones.

Contributions

– Demonstrate the powerfulness of hardware-assisted security and the formal abstraction of Attested Execution Secure Processors (AESPs) that fit to build a secure

†Also, belonging to NTT DOCOMO, INC.

[1]See George Orwell's novel, *"Nineteen Eighty-Four(1984) – Big Brother Is Watching You,"* published in 1948.
[2]https://ethereum.org
[3]https://www.shocard.com

 DOI 10.1109/Blockchain55522.2022.00012

Self-Sovereign Identity (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 [17]. Thus, $\Pi^{\mathcal{G}_{\mathtt{att}}}$ allows not to rely on multi-party computation (MPC) that requires such a trusted party of nodes, and it brings more flexibility and efficiency than the existing systems.

II. Background

A. Digital Identity

The importance of "digital identity" is rapidly increasing under the current circumstance. Many people must feel that many things have changed after the pandemic. People would need to do much more things online than before. Service providers, companies, and governmental organizations need to provide services for customers, employees, and citizens online to deal with the circumstance. Ideas and past efforts on digital identity are not only for the occasion today, but accumulated contributions have helped society survive the pandemic.

Researchers and influencers in the tech industry in this domain refer to Kim Cameron's blog article, "The Laws of Identity" [18]. 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(en) [19], and a credential represents an identity for use in authentication. There are also many activities at standardization bodies such as W3C[4], OpenID Foundation[5], and FIDO Alliance[6]. The NIST's Digital Identity Guidelines [20] 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 got moved to the Federated User Identity (FED) model by mathematical definition of Md Sadek Ferdous et al.'s work [21]. The (full) identity representing a natural person is a union of partial identities, each set of claims consisting of 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.

B. Decentralized Digital Identity

In the tech industry, several initiatives are addressing decentralized digital identity. Microsoft has been driving an initiative[7] and announced ION[8] on behalf of the Decentralized Identity Foundation (DIF)[9] in May 2021. The approach utilizes W3C's DIDs [22], decentralized systems such as blockchains and ledgers, and DIF's standards. There are also many pieces of research addressing decentralized identity in academia. The authors would like to focus on modern cryptographic approaches for preserving privacy because of our motivations. Thus, we will address Deepak Maram et al.'s work [17] in this section.

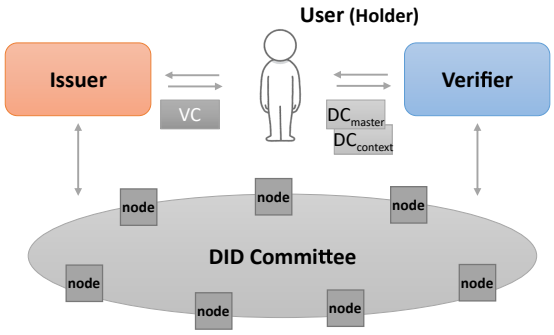

Fig. 1. CanDID – A State-of-the-Art Approach toward Decentralized Digital Identity

CanDID: It can do decentralized identity with legacy compatibility, Sybil-resistance, and accountability. They identify remaining problems for building a decentralized identity system, legacy compatibility, Sybil-resistance, and accountability as entitled. In order to solve the problems, they propose system protocols with a trusted committee of nodes-based architecture. Fig. 1 illustrates the overview of CanDID's approach. In the figure, VC stands for a Verifiable Credential, and DC stands for a Derived Credential, $\mathsf{DC}_{\mathsf{master}}$ means a master credential, and $\mathsf{DC}_{\mathsf{context}}$ means a context-based credential in their work.

The CanDID system protocols provide three APIs for issuing credentials, `issuePreCred`(), `issueMasterCred`(), and `issueCtxCred`(). 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. 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 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

[4]https://www.w3.org
[5]https://openid.net/foundation/
[6]https://fidoalliance.org

[7]https://www.microsoft.com/en-us/security/business/identity-access-management/decentralized-identity-blockchain
[8]https://identity.foundation/ion/
[9]https://identity.foundation

matching for scanning sanction lists to avoid Anti-Money-Laundering (AML).

C. Self-Sovereign Identity (SSI)

Christopher Allen published his blog article entitled "The Path to Self-Sovereign Identity" in 2016 [1]. 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.*
9) **Minimalization**. *Disclosure of claims must be minimized.*
10) **Protection**. *The rights of users must be protected.*

As David Chaum proposed in 1985, the motivation toward Self-Sovereign Identity (SSI) described by Christopher Allen has been encouraging many researchers to create and establish proper SSI systems preserving privacy from various perspectives [2] [23] [24] [4] [5]. Some research addressed if the ten principles express all the principles that may describe the essentials of SSI [9] [25]; however, they have still been treated as foundational principles.

1) Extended Principles: Some research addressed if the ten principles express all principles which may describe the essentials of SSI. Quinten Stokkink et al. proposed to add another principle **Provable** [9]. Md Sadek Ferdous et al. tried to do a comprehensive survey and proposed five taxonomies and 17 principles under the taxonomies of classes derived from the ten principles [4]. 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 [25].

2) Building Blocks and Blockchain: Many pieces of research addressed how to build SSI systems; essential components [8], design patterns [26], and needs of, how to utilize, or if it requires blockchain technology [9] [27] [10] [28] [5] [29]. Two of these researches concluded that blockchain was not mandated. 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.

3) SSI Systems with Mobile Devices: There are several SSI implementations such as uPort and Sovrin [28] [13]. Also, some experimental research and prototypes in the tech industry support governmental agencies' interests [30] [31]. Through such activities, including W3C's efforts on verifiable credentials and DIDs [32] [22], 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.

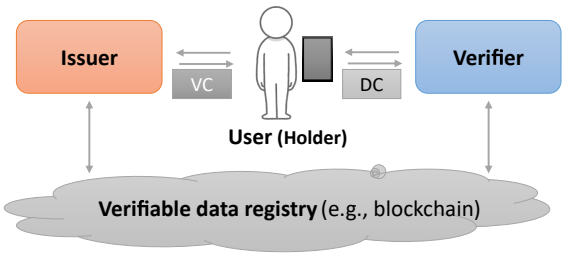

Fig. 2. Proposed Self-Sovereign Identity Systems in the Tech Industry, Today

Fig. 2 illustrates such a proposed SSI solution architecture in our interpretation. An issuer issues a verifiable credential (VC in the figure) for the holder. To minimize disclosure, they may have a derived credential (DC in the figure) for presentation. A verifier may verify with the received derived credential per a request. Blockchain technology can be a verifiable data registry in the proposed structure.

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 to utilize hardware-assisted security features of the mobile device. Kalman C. Torh et al. addressed utilizing users' mobile devices as a digital identity for each of them [24]; however, they have not mentioned opportunities for hardware-backed attestations.

III. Preliminaries - Attested Execution Secure Processors (AESPs)

Hardware-assisted security has recently been becoming the norm for mobile devices. Apple's iPhone implements Secure Enclave, a dedicated secure subsystem, and it is isolated from the apps execution environment on the main processor[10]. 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[11], 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[12]. There are also other design choices among implementations in the industry,

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

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

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

such as Global Platform-supported Secure Elements[13] and Intel's SGX[14], in addition to TrustZone and TPM 2.0.

Among numerous implementations and researches addressing hardware-assisted security including how to realize [14] and how to utilize [33], Rafael Pass et al. uniquely addressed hardware-assisted security, secure processors, in a formal fashion [16]. In their words, trusted hardware is commonly believed to provide a very 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 $\mathcal{G}_{\mathtt{att}}$ from the common belief. Also, they successfully demonstrated an additional observation regarding composable two-party computation with attested execution processors.

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

According to their efforts, 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 compute $\mathtt{outp} := \mathtt{prog}(\mathtt{inp})$. The secure processor then signs the tuple $(\mathtt{prog}, \mathtt{outp})$ with a secret signing key 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), in the sense that 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 intend to provide. $\mathcal{G}_{\mathtt{att}}$ is parametrized with a signature scheme Σ and also a registry $\mathtt{reg}$ that is meant to capture all the platforms equipped with an AESP. The registry $\mathtt{reg}$ is treated 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 getpk(), and stateful enclave operations of Install() and Resume(), which realize the anonymous attestation capability. A platform $\mathcal{P}$ that is in the registry $\mathtt{reg}$ may invoke those enclave operations[15].

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

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

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

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

IV. 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.

A. Architecture

Fig. 3 illustrates 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 ask authorities such as a governmental agency, a school, or other service providers to issue a verifiable credential consisting of claims and a proof π for each claim.
- The person may have a mobile device equipped with an AESP, complying with the proposed AESP-based SSI architecture, needs to set up for making their device as a Self-Sovereign Identity holder. This setup operation includes a device key pair $(\mathtt{pk}_M, \mathtt{sk}_M)$ generation.

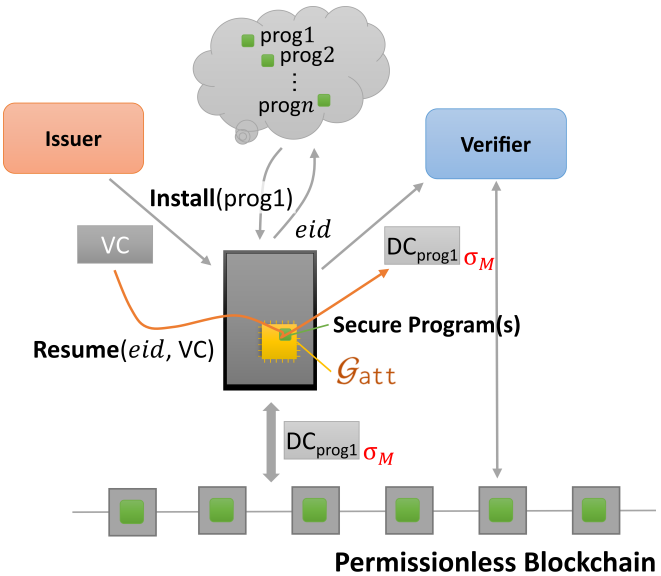

Fig. 3. Overview of the Proposal Architecture and the AESP Enclave Operations of Install() and Resume()

- The person may install programs, $\texttt{prog}_{1,\ldots,n}$, by Install() for enabling the device SSI-operations capable such as creating derived credentials. For example, a prog is designed and implemented for minimizing disclosure of their original, verifiable credentials, less than 18 years old in particular.
- Once the installed programs are executed by Resume(), and the AESP digitally signs an output outp to prove that the programs have 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; thus, their identity is to be represented with a key pair $(\texttt{pk}_U^E, \texttt{sk}_U^E)$. For simplicity, we will describe such a key pair like $(\texttt{pk}_U, \texttt{sk}_U)$ in this paper.
- Such verifiable credentials or derived credentials signed by the AESP with each proof are registered to a permissionless blockchain system as a repository.
- Verifiers may utilize the signed credentials with a corresponding proof for each credential to verify if a person is requesting to subscribe and use services provided by the verifiers.

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 or not 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.

B. 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 birthday. 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 allows 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 $(\texttt{pk}_U, \texttt{sk}_U)$. Fig. 4 illustrates the relationship among real identities $\mathcal{I}$, Symbil-resistant credentials $\mathcal{C}$, and derived credentials some of which are Sybil-resistant.

Programs that require to manage credentials that meet the Sybil-resistance requirement should implement a function of injective *identification map* ψ between Sybil-resistant derived credentials and $\mathcal{C}$. AESP may install and execute such programs capable of treating ψ securely. The following section and Fig. 6 will describe the detailed protocol for creating Sybil-resistant credentials.

V. Protocols in Detail

The proposed SSI architecture defines and provides some primitive protocols as described in Fig. 5 and Fig. 6. 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 $\texttt{pk}_M$ and $\texttt{sk}_M$ as determined in the Rafael Pass et al.'s works [16].

Definition 1 (A mobile device equipped with $\mathcal{G}_{\texttt{att}}$)**.** The ideal formal abstraction of Attested Execution Secure Processors (AESPs) is said to be $\mathcal{G}_{\texttt{att}}$, and let assume that every natural person's mobile device who needs SSI is equipped with $\mathcal{G}_{\texttt{att}}$.

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 an AESP, $\mathcal{G}_{\texttt{att}}$ enables the above proposed protocol Π to realize a set of secure Self-Sovereign Identity (SSI) system protocols $\Pi^{\mathcal{G}_{\texttt{att}}}$ for them.*

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

VI. Security Properties

With respect to the CanDID's contributions [17], we will follow how CanDID demonstrates their protocols of decentralized identity systems are designed securely as much as possible. In particular, they define CanDID API; in some of their definitions, adversaries have unlimited access to the entire

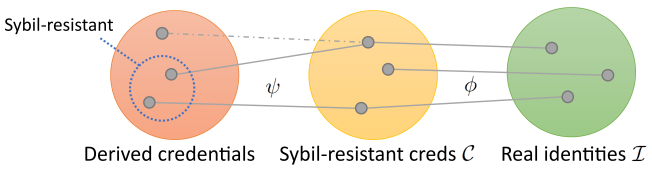

Fig. 4. Sybil-resistant derived credentials and the *identification map* $\psi : \texttt{cred} \to \mathcal{C}$

AESP-based Secure Self-Sovereign Identity (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 Attested Execution Secure Processors (AESPs). The other class is a set of primitive functions for Self-Sovereign Identity (SSI) working with an AESP. The protocol $\Pi^{\mathcal{G}_{\mathtt{att}}}$ provides flexibility by allowing a natural person to choose and install programs $\mathtt{prog}$ for various needs.

The followings are $\Pi^{\mathcal{G}_{\mathtt{att}}}$ generic functions relying on $\mathcal{G}_{\mathtt{att}}$:

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

The followings are $\Pi^{\mathcal{G}_{\mathtt{att}}}$ SSI-featured functions:

- $\mathsf{KeyGen}(1^\lambda) \to (\mathtt{pk}_U^E, \mathtt{sk}_U^E)$.
  1: An AESP generates a user's key pair $(\mathtt{pk}_U^E, \mathtt{sk}_U^E)$, a pseudonym for each Entity. For simplicity, we omit E in 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;
  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.

- $\mathsf{IssueDCred}\,(\mathtt{sk}_M, \mathtt{sk}_U, \mathtt{pk}_U^{\mathtt{new}}, \mathtt{ctx}, \mathtt{cred}) \to \mathtt{derivedCred}$.

  This function, $\mathsf{IssueDCred}(\mathtt{ctx})$, is a program $\mathtt{prog}$, which can be vary for different context $\mathtt{ctx}$. To install and execute $\mathtt{prog}$,
  1: $\mathcal{G}_{\mathtt{att}}.\mathsf{Install}(\mathtt{prog}) \to eid$;
  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, outputs $\mathtt{output}$ are $\{\mathtt{pk}_U^{\mathtt{new}}, \mathtt{prog}, (\mathtt{claim}_j)_{j=1,\ldots,m}, \sigma_M\}$ as $\mathtt{derivedCred}$, where σ_M is $\Sigma.\mathsf{Sig}_{\mathtt{sk}_M}(eid, \mathtt{prog}, \mathtt{outp})$, and $\mathtt{prog}$ should be an open-source map 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 credetial, the program $\mathtt{prog}$ should satisfy the construction defined in Fig. 6.

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

  User U inputs $\mathtt{sk}_U$ and $\mathtt{cred}$, verifying party V inputs a challenge c, and the public key $\mathtt{pk}_M$ is a public input. To verify $\mathtt{cred}$,
  1: User U sends $(\mathtt{cred}, \sigma_M)$ to V where $\sigma = \mathsf{Sig}_{\mathtt{sk}_U}(c)$;
  2: Verifying party V checks if

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

Fig. 5. The Construction of $\Pi^{\mathcal{G}_{\mathtt{att}}}$, AESP-based Self-Sovereign Identity (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 `prog` should satisfy the following construction: the program `prog` treats $(\mathtt{pk}_U, \widehat{\psi})$ as inputs `inp`, where $\widehat{\psi}$ is Sybil-resistant pseudonymizer to transform verifiable credentials to a set of claims satisfying the injective *identification map*

$$\psi : \mathtt{cred} \to \mathcal{C}$$

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

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

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

The generated derived credential consists of $\mathtt{pk}_U$, $\widehat{\psi}$ as `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 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 credential 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. 6. The Construction of the program for creating Sybil-resistant credentials in $\Pi^{\mathcal{G}_{\mathtt{att}}}$

CanDID API, which they model for conciseness as an oracle $\mathcal{O}^*$. Also, in their security definitions, the adversaries may have access to an external account oracle $\mathcal{O}^*_{\mathtt{ext}}$ that models the legacy providers called by CanDID. We will reuse the same oracle models for our AESP-based SSI system protocols $\Pi^{\mathcal{G}_{\mathtt{att}}}$.

The protocol $\Pi^{\mathcal{G}_{\mathtt{att}}}$ 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.

A. Sybil-Resistance

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

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

In the real world, a national PKI system, e.g., JPKI (described in VIII-A), 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 $\mathcal{C}$ be a set of all Sybil-resistant credentials. A derived credential `cred` is said to be Sybil-resistant with respect to $\mathcal{C}$ if and only if, for any PPT (Probabilistic Polynomial-Time) adversary $\mathcal{A}$ and security parameter λ, there exists an *identification map* $\psi : \mathtt{cred} \to \mathcal{C}$,

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

Informally, this definition captures the infeasibility of an adversary to obtain a credential that is not in the set of all Sybil-resistant credentials. Here, the *identification map* ψ is defined over all elements in derived credentials such that $\psi(\mathtt{cred}) \in \mathcal{C}$. 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.

B. Unforgeability

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

Definition 6 (Unforgeability)**.** Let `chals` denote a set of all challenges and their responses produced by $\mathcal{A}$ in oracle access with $\mathcal{O}^*$ and a special oracle $\mathcal{O}^*_{\mathtt{sk}_U}$ 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 $\mathcal{A}$,

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

The definition captures that it must be infeasible for an adversary to impersonate users, i.e., forge signatures with users' keys.

C. 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 $\mathcal{A}$,

$$\left|\Pr\left[b=b' \;\middle|\; \begin{array}{c} \mathtt{pk}_M,\mathtt{sk}_M \leftarrow \mathsf{KeyGen}(1^\lambda); \\ \mathtt{pk}_U,\mathtt{sk}_U,\{c_1^0,\ldots,c_l^0\},\{c_1^1,\ldots,c_l^1\} \leftarrow \mathcal{A}^{\mathcal{O}^*,\mathcal{O}^*_{\mathsf{ext}}}(\mathtt{pk}_M); \S \\ \mathtt{cred}^0 \leftarrow \mathsf{IssueDCred}(\mathtt{sk}_M,\mathtt{sk}_U,\mathtt{pk}_U,\{c_1^0,\ldots,c_l^0\},\mathtt{prog}), \\ \mathtt{cred}^1 \leftarrow \mathsf{IssueDCred}(\mathtt{sk}_M,\mathtt{sk}_U,\mathtt{pk}_U,\{c_1^1,\ldots,c_l^1\},\mathtt{prog}) \\ \text{where } \mathtt{cred}^0=(\mathtt{pk}_U,\{\mathtt{claim}_j^0\}_{j=1,\ldots,m},\widehat{\psi^0},\mathtt{prog},\sigma_M^0) \\ \text{and } \mathtt{cred}^1=(\mathtt{pk}_U,\{\mathtt{claim}_j^1\}_{j=1,\ldots,m},\widehat{\psi^1},\mathtt{prog},\sigma_M^1); \\ \text{assert } \{\mathtt{claim}_j^0\}_{j=1,\ldots,m}=\{\mathtt{claim}_j^1\}_{j=1,\ldots,m} \text{ as sets}; \\ b \leftarrow \$\{0,1\}; \\ b' \leftarrow \mathcal{A}^{\mathcal{O}^*,\mathcal{O}^*_{\mathsf{ext}}}(\mathtt{cred}^b) \end{array}\right] - \frac{1}{2}\right| \leq \mathrm{negl}(\lambda)$$

§ where $\{\mathtt{cred}_k^0\}_{k=1,\ldots,l}$ is denoted $\{c_1^0,\ldots,c_l^0\}$ and $\{\mathtt{cred}_k^1\}_{k=1,\ldots,l}$ is denoted $\{c_1^1,\ldots,c_l^1\}$ as a set of claims for each $\{0,1\}$.

D. Privacy - Credential-Verification

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

Definition 8 (Credential verification privacy)**.** Given an open-source map `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 `prog`.

E. 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 $\mathcal{A}$,

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

VII. 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. 5 and Fig. 6, which is a set of secure Self-Sovereign Identity (SSI) system protocols.

A. Sybil-Resistance

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

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

it implies that $\mathcal{G}_{\mathtt{att}}$ can decrypt $\widehat{\psi}$ to get `cred` and verify the relation $\mathtt{cred} \in \mathcal{C}$ unless the signature σ_M is forged. The latter probability is negligible given Σ is EUF-CMA signature scheme.

B. 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 VerifyCred(). Thus, unforgeability of the $\Pi^{\mathcal{G}_{\mathtt{att}}}$ based SSI systems follows in a straightforward way.

Here, `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 `prog` with inputs $\{\mathtt{cred}_k\}_{k=1,\ldots,l}$. Thus, `chals` contains all tuples appeared in the oracle access by $\mathcal{A}$ of the form $(\mathtt{pk}_U, \widehat{\psi}, \{\mathtt{claim}_j\}_{j=1,\ldots,m})$. 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 $\mathrm{negl}(\lambda)$, that is negligible in the security parameter λ.

C. 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 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.

D. 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 IssueDCred() and VerifyCred(). We also expect that only `prog` will be accepted by both users and providers that reach the consensus. Such `prog` only leaks required privacy information described as a set of claims $\{\mathtt{claim}_j\}_{j=1,\ldots,m}$. This process are expected to leak any more information as defined in **Definition 8**.

E. 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 $\texttt{cred}^0$ or $\texttt{cred}^1$ 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 for unlinkability in our scheme. □

VIII. Applications, Limitations and Future Directions

There are many opportunities and applications of this proposal 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 blockchain in this proposal plays a role in building Self-Sovereign Identity (SSI) systems as a foundation. Like previous research and implementations, it works 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 blockchain 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 $\texttt{prog}$ depending on different context $\texttt{ctx}$, 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 of blockchain-based business ecosystems, opportunities to utilize the main idea of this proposal to combine permissionless blockchain and AESPs are unlimited.

A. Applications

One of the well-known initiatives is mDL, mobile driver's license[16]. 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 show 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: An ongoing initiative, driven by the Japanese government, is to enable JPKI, a part of My Number Individual Card capabilities, on smartphones. In order to realize a goal to duplicate digital certificates with key pairs, the initiative plans to utilize 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 those. For example, a mobile device may become a digital identity for the user once the registration process is completed. 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.

B. Limitations and Future Directions

We have focused on utilizing the abstraction of Attested Execution Secure Processors (AESPs) together with permissionless blockchain in order to build a secure SSI system through defining the architecture and the system protocols, $\Pi^{\mathcal{G}_{\mathtt{att}}}$. In this section, we would like to describe three problems that are remaining and will address to solve as our future works.

1) Addressing Complexity in the Real World: We propose to incorporate Rafael Pass et al.'s contribution regarding the formal abstraction of Attested Execution Secure Processors (AESPs) [16] 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.

2) Potential Vulnerability of Hardware-Assisted Security: Some readers might be concerned about the vulnerability residing tamper-resistant secure processors to compromise. We plan to address defining a treat model to cover such vulnerability. One of those is the globally shared key pair of $\texttt{pk}_M$ and $\texttt{sk}_M$, 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 to address the problem [34], firstly adopted onto TPM. Further, Christina Garman et al.'s contributions proposed decentralized direct anonymous attestation [35].

3) Practice: We have not addressed in detail how to utilize permissionless blockchain in aligning with the proposed architecture and the system protocols $\Pi^{\mathcal{G}_{\mathtt{att}}}$ in this paper. Some existing SSI systems are already deployed utilizing permissionless blockchain, and we plan to design a prototype of the proposed architecture based on the existing permissionless blockchain systems [11] [12] [13] such as Ethereum 2.0[17].

The further detailed design includes *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*.) Interface for such a natural person to access programs that can and should be maintained on the permissionless blockchain, and *c*.) some applications, such as a scenario where a natural person purchases something with their wallet on the permissionless blockchain only when a shop there may verify if the person's age is over 18 years old, but other private information is not disclosed.

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

[17]https://ethereum.org/en/eth2/

IX. 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 $\Pi^{\mathcal{G}_{\mathsf{att}}}$ along with security properties including Sybil-resistance and a proof sketch of the security properties.

Assuming AESPs and $\mathcal{G}_{\mathsf{att}}$, the AESP-based SSI system protocols $\Pi^{\mathcal{G}_{\mathsf{att}}}$ eliminates the online distributed committee of trusted nodes assumed in CanDID; thus, $\Pi^{\mathcal{G}_{\mathsf{att}}}$ 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.

Acknowledgment

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; we have been very encouraged by their comments.

References

[1] C. Allen, "The Path to Self-Sovereign Identity," Apr. 2016. [Online]. Available: http://www.lifewithalacrity.com/2016/04/the-path-to-self-soverereign-identity.html

[2] 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.

[3] 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)*, pp. 90–95, Aug. 2020.

[4] M. S. Ferdous, F. Chowdhury, and M. O. Alassafi, "In Search of Self-Sovereign Identity Leveraging Blockchain Technology," *IEEE Access*, vol. 7, pp. 103 059–103 079, Aug. 2019.

[5] G. Kondova and J. Erbguth, "Self-sovereign identity on public blockchains and the GDPR," *Proceedings of the 35th Annual ACM Symposium on Applied Computing*, Mar. 2020.

[6] C. Gomez Munoz, "eIDAS Supported Self-Sovereign Identity," Tech. Rep., May 2019. [Online]. Available: https://ec.europa.eu/futurium/en/system/files/ged/eidas_supported_ssi_may_2019_0.pdf

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

[8] A. Mühle, A. Grüner, T. Gayvoronskaya, and C. Meinel, "A Survey on Essential Components of a Self-Sovereign Identity," *ArXiv*, vol. abs/1807.06346, Jul. 2018.

[9] Q. Stokkink and J. Pouwelse, "Deployment of a Blockchain-Based Self-Sovereign Identity," in *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, Jun. 2018, pp. 1336–1342.

[10] D. van Bokkem, R. Hageman, G. Koning, L. Nguyen, and N. Zarin, "Self-Sovereign Identity Solutions: The Necessity of Blockchain Technology," *ArXiv*, vol. abs/1904.12816, Apr. 2019.

[11] 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," in *2020 IEEE International Symposium on Systems Engineering (ISSE)*, Nov. 2020, pp. 1–7.

[12] S. Nakamoto, "Bitcoin : A Peer-to-Peer Electronic Cash System," 2008.

[13] P. J. Windley, "Sovrin: An Identity Metasystem for Self-Sovereign Identity," *Frontiers in Blockchain*, vol. 4, Jul. 2021. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fbloc.2021.626726/full

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

[15] E. Shi, F. Zhang, R. Pass, S. Devadas, D. Song, and C. Liu, "Trusted Hardware: Life, the Composable Universe, and Everything (2015 12 15 4 Elaine Shi, et el.)," May 2017. [Online]. Available: https://www.youtube.com/watch?v=57KnieAVFkY

[16] R. Pass, E. Shi, and F. Tramèr, "Formal Abstractions for Attested Execution Secure Processors," in *Advances in Cryptology – EUROCRYPT 2017*, vol. LNCS, volume 10210, Apr. 2017, pp. 260–289.

[17] 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," in *2021 IEEE Symposium on Security and Privacy (SP)*, May 2021, pp. 1348–1366.

[18] K. Cameron, "The Laws of Identity," May 2005. [Online]. Available: https://www.identityblog.com/stories/2005/05/13/TheLawsOfIdentity.pdf

[19] ISO/IEC, "ISO/IEC 24760-1:2019(en), IT Security and Privacy — A framework for identity management — Part 1: Terminology and concepts," 2019. [Online]. Available: https://www.iso.org/obp/ui/#iso:std:iso-iec:24760:-1:ed-2:v1:en

[20] P. A. Grassi, M. E. Garcia, and J. L. Fenton, "NIST Special Publication 800-63-3 Digital Identity Guidelines," Jun. 2017. [Online]. Available: https://pages.nist.gov/800-63-3/

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

[22] W3C, "Decentralized Identifiers (DIDs) v1.0," Aug. 2021. [Online]. Available: https://www.w3.org/TR/did-core/

[23] P. Dunphy, L. Garratt, and F. Petitcolas, "Decentralizing Digital Identity: Open Challenges for Distributed Ledgers," *2018 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, pp. 75–78, Apr. 2018.

[24] 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.

[25] A. Satybaldy, M. Nowostawski, and J. Ellingsen, "Self-Sovereign Identity Systems: Evaluation Framework," in *Privacy and Identity Management. Data for Better Living: AI and Privacy*, M. Friedewald, M. Önen, E. Lievens, S. Krenn, and S. Fricker, Eds. Cham: Springer International Publishing, Mar. 2020, vol. 576, pp. 447–461.

[26] 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*, May 2020.

[27] A. Grüner, A. Mühle, and C. Meinel, "On the Relevance of Blockchain in Identity Management," *arXiv:1807.08136 [cs]*, Jul. 2018, arXiv: 1807.08136. [Online]. Available: http://arxiv.org/abs/1807.08136

[28] A.-E. Panait, R. F. Olimid, and A. Stefanescu, "Analysis of uPort Open, an Identity Management Blockchain-Based Solution," in *TrustBus*, Sep. 2020.

[29] 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*, pp. 495–504, Jun. 2021.

[30] D. Du Seuil, "European Self Sovereign identity framework," Jul. 2019, https://ssimeetup.org/understanding-european-self-sovereign-identity-framework-essif-daniel-du-seuil-carlos-pastor-webinar-32/.

[31] A. Boysen, "Decentralized, Self-Sovereign, Consortium: The Future of Digital Identity in Canada," in *Frontiers in Blockchain*, 2021.

[32] W3C, "Verifiable Credentials Data Model 1.1," Mar. 2022. [Online]. Available: https://www.w3.org/TR/vc-data-model/

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

[34] E. Brickell, J. Camenisch, and L. Chen, "Direct anonymous attestation," in *Proceedings of the 11th ACM conference on Computer and communications security - CCS '04*. Washington DC, USA: ACM Press, 2004, pp. 132–145.

[35] C. Garman, M. Green, and I. Miers, "Decentralized Anonymous Credentials," *IACR Cryptol. ePrint Arch.*, vol. 2013, Oct. 2013.