



University of
Zurich^{UZH}

Identity Management in a Decentralized Remote Electronic Voting System

Simon Bachmann
Zurich, Switzerland
Student ID: 14-709-893

Supervisor: Christian Killer, Eder John Scheid,
Prof. Dr. Burkhard Stiller
Date of Submission: June 18, 2021

Abstract

Remote Electronic Voting (REV) systems introduce new challenges that do not exist in the in-person or postal voting process. The right of ballot secrecy poses one of its core challenges. Identity verification is closely coupled with the voters' privacy and plays a crucial role in REV systems. However, the digital identities that are used for most online services today, do not respect data protection. Most applications outsource identity management to third parties such as Google and Facebook, allowing them to track their users across many other services. The convenience of only having one account for most online interactions comes at a price in the form of privacy intrusion. The problem is that users do not own their digital identity but Identity Providers (IdPs) are responsible for issuing, storing and verifying credentials. As a consequence of having a single entity in charge of all three duties, IdPs aggregate vast amount of sensitive data and become a popular target with cyber criminals.

Another challenge for a digital identity is that many credential are issued in the form of physical documents. Analyzing, evaluating and verifying the security features of physical credentials in the digital world results in a probabilistic outcome. Thus, physical credentials are considered non-native to the web and require slow, expensive and error-prone verification methods. Self-Sovereign Identity (SSI) is a form of digital identity that allows users to regain control by maintaining the credentials and selectively disclose them with trusted entities without the need of a Trusted Third Party (TTP) and without a cumbersome certificate verification process.

This thesis focuses on Identity Management (IdM) systems of REV applications and conducts a security analysis on ProvoTum's authentication and authorization process. The result of the audit discloses several vulnerabilities in terms of privacy and security due to the trusted role of the IdP and the usage of email addresses as identifiers. An SSI-based authentication and authorization system is designed and implemented addressing the privacy and security concerns. The new design radically changes the processes of credentials issuance, storage and disclosure which is crucial for providing privacy to voters. Instead of using a predefined list of eligible identifiers from an internal IdM system, an election is configured such that voters can only participate if the necessary credentials are presented. The digital certificates used in the new architecture are cryptographically linked to Decentralized Identifier (DID) which are maintained on a public permissionless ledger. The combination of self-certifying, ledger-based identifiers and credentials that can selectively be disclosed and verified by anyone without the need of a TTP, guarantees a higher degree of privacy and security by design.

Zusammenfassung

Systeme zur elektronischen Fernabstimmung (Remote Electronic Voting, REV) bringen neue Herausforderungen mit sich, die es bei der persönlichen Stimmabgabe oder der Briefwahl nicht gibt. Das Wahlgeheimnis in einem elektronischen Prozess sicherzustellen, ist eine der Kernherausforderungen. Die Identitätsprüfung ist eng mit der Privatsphäre der Wähler verbunden und spielt in REV-Systemen eine entscheidende Rolle. Die digitalen Identitäten, die heute für die meisten Online-Dienste verwendet werden, berücksichtigen den Datenschutz jedoch nicht. Die meisten Anwendungen lagern das Identitätsmanagement an Dritte wie Google und Facebook aus und ermöglichen es ihnen, ihre Nutzer über viele andere Dienste hinweg zu verfolgen. Für den Komfort, nur ein Konto für alle Web-Interaktionen zu besitzen, wird ein hoher Preis in Form von Eingriffen in die Privatsphäre bezahlt. Das Problem besteht darin, dass die Benutzer nicht Eigentümer ihrer digitalen Identität sind, sondern Identitätsanbieter (Identity Providers, IdPs) für die Ausstellung, Speicherung und Verifizierung von Anmeldeinformationen verantwortlich sind. Als Folge davon, dass eine einzige Instanz für alle drei Aufgaben zuständig ist, sammeln IdPs riesige Mengen sensibler Daten und werden zu einem beliebten Ziel für Cyber-Kriminelle.

Eine weitere Herausforderung für eine digitale Identität ist, dass viele Zertifikate in Form von physischen Dokumenten ausgestellt werden. Die Analyse, Bewertung und Verifizierung der Sicherheitsmerkmale von physischen Ausweisen in der digitalen Welt führt zu einem probabilistischen Ergebnis. Daher gelten physische Zertifikate als nicht webfähig und erfordern langsame, teure und fehleranfällige Verifizierungsmethoden. Selbstbestimmte Identity (Self-Sovereign Identity, SSI) ist eine Form der digitalen Identität, die es dem Benutzer ermöglicht, die Kontrolle über seine digitale Identität wiederzuerlangen und Zertifikate selektiv an vertrauenswürdige Instanzen weiterzugeben, ohne dass eine vertrauenswürdige dritte Partei (Trusted Third Party, TTP) benötigt wird und ohne einen umständlichen Zertifikatsverifizierungsprozess.

Diese Arbeit konzentriert sich auf das Identitätsmanagementsysteme in REV-Anwendung und führt eine Sicherheitsanalyse des Authentifizierungs- und Autorisierungsprozesses von ProvoTum durch. Das Ergebnis der Prüfung offenbart mehrere Schwachstellen in Bezug auf Datenschutz und Sicherheit aufgrund der vertrauenswürdigen Rolle des IdP und der Verwendung von Email-Adressen als Identifikatoren. Ein SSI-basiertes Authentifizierungs- und Autorisierungssystem wird entworfen und implementiert, das die Datenschutz- und Sicherheitsbedenken adressiert. Das neue Design ändert den Prozess der Ausgabe, Speicherung und Offenlegung von Berechtigungsnachweisen, was für die Gewährleistung der Privatsphäre der Wähler entscheidend ist. Anstatt eine vordefinierte Liste von berechtigten Identifikatoren aus einem internen IdM-System zu verwenden, wird eine Abstimmung

so konfiguriert, dass Wähler nur dann teilnehmen können, wenn die erforderlichen Zertifikate vorgelegt werden. Die digitalen Zertifikate, die in der neuen Architektur verwendet werden, sind kryptographisch verknüpft mit dezentrale Identifikatoren (Decentralized Identifiers, DID), die in einem öffentlichen Register verwaltet werden. Die Kombination aus öffentlichen, selbstzertifizierenden Identifikatoren und signierte Zertifikate, die selektiv offengelegt und von jedem verifiziert werden können, ohne dass ein TTP erforderlich ist, garantiert ein höheres Mass an Privatsphäre und Sicherheit.

Acknowledgments

I would like to express my sincere gratitude to my supervisor Christian Killer for the continuous support of my Master's Thesis. His patience, motivation and profound knowledge in the field of DLT and e-voting is inspiring.

Besides my supervisor, I would like to thank Prof. Dr. Burkhard Stiller, head of the Communication Systems Research Group (CSG), for giving me the opportunity to work on such interesting and currently relevant topics.

Contents

Abstract	i
Zusammenfassung	iii
Acknowledgments	v
1 Introduction	1
1.1 Description of Work	2
1.2 Thesis Outline	3
2 Background	5
2.1 Centralized Identity Model	5
2.2 Federated Identity Model	6
2.3 Decentralized Identity Model (Self-Sovereign)	7
3 Related Work	11
3.1 World Wide Web Consortium (W3C)	11
3.2 Decentralized Identity Foundation (DIF)	11
3.3 JSON-LD Data Encoding Standard	12
3.3.1 Schema.org	13
3.4 Veramo	14

4	Self Sovereign Identity (SSI) Primitives	15
4.1	Decentralised Identifier	15
4.1.1	Binding Problem in PKI	15
4.1.2	DIDs in the Context of Other Identifiers	17
4.1.3	The Scheme, DID Method and DID Method-Specific String	19
4.1.4	DID Document	21
4.1.5	DID Resolution and DID Resolvers	21
4.1.6	DID Auth	22
4.2	Verifiable Credential (VC)	22
4.3	Selective Disclosure Request (SDR)	23
4.4	Verifiable Presentation (VP)	23
4.5	Fork of the Verifiable Registry	23
4.6	Identity Contexts	24
5	Provotum 3.0 Security Analysis	25
5.1	Provotum 3.0 Architecture	25
5.2	Eligibility Verification Process	27
5.2.1	Registration	27
5.2.2	List Communication	28
5.2.3	Authentication	29
5.2.4	Authorization	31
6	Design of an SSI-based IdM System for Provotum	33
6.1	Stakeholders	33
6.2	Processes	35
6.2.1	Identity Issuance	35
6.2.2	Authentication Configuration	36
6.2.3	Identity Verification	37

<i>CONTENTS</i>	ix
7 Implementation	39
7.1 DID Creation	41
7.2 Credential Issuance	41
7.3 Authentication Configuration	42
7.4 Identity Verification	45
8 Evaluation	49
8.1 Privacy	50
8.1.1 IdP IA Collusion	51
8.2 Security	52
8.3 Infrastructure	53
8.3.1 Veramo	54
8.3.2 Risk of Centralization Due to Vast Amount of DID Methods	54
8.3.3 JSON-LD Encoding Schema	54
8.3.4 Incomplete W3C Specification	55
9 Summary and Conclusion	59
9.1 Future Work	59
9.1.1 IdP as TTP	60
9.1.2 IdP as SPOF	61
9.1.3 ZKP-based Authentication Process	62
9.1.4 Mobile Client for DID and VC Maintenance	62
9.1.5 Credential Registries and Governance Frameworks	62
Abbreviations	67
Glossary	69
List of Figures	69
List of Tables	72

A	Installation Guidelines	75
B	Contents of the CD	77

Chapter 1

Introduction

”The chain is only as strong as its weakest link, for if that fails the chain fails and the object that it has been holding up falls to the ground” [24]. This analogy closely relates to the security of software applications. Each component of a systems has to be evaluated separately and the overall security is defined by its weakest element. In the context of ledger-based Remote Electronic Voting (REV) systems, it is necessary to also analyze the Identity Management (IdM) system which defines the set of eligible voters. IdM systems maintain highly sensitive data about their users and embodies some of the biggest honeypots for cyber-criminals.

Bringing physical certificates into the digital space is a challenging task. Today, this is solved by verifying physical security features with scanners, image recognition software or in-person checks which is a slow, expensive and error-prone process. Whenever users must provide proof of identity online, an onboarding procedure is required to validate the physical certificates. After a successful onboarding process, the user account of that person is considered as identified in the given context. However, the user account is loosely coupled to the physical certificate and therefore, an early invalidation of the physical document may not be noticed immediately within the application. This can create data inconsistency between the IdM system and original identity context and poses a threat to systems that depend on it. An IdM system that can digitally issue credentials and verify them at all times improves this processes in all of those shortcomings.

Since its rise to popularity, Distributed Ledger Technology (DLT) has not only been described as the missing part for a transparent, secure and tamper-proof REV system but also for an open, self-governed and cryptographically provable identity system. While DLT can guarantee many of these properties by design, empowering people to maintain their own digital identity in the form of cryptographically signed credentials, managing identities on a public registry introduces new challenges and attack vectors. The use of a decentralized IdM system in combination with a ledger-based REV system has to be assessed before it is deployed on a large scale.

The CSG@IFI studies, designs and implements different ledger-based REV architectures [7]. ProvoTum 3.0, or ProvoTumRF [3], uses the concept of a Public Bulletin Board (PBB) [13] and addresses the architectural limitations of the Ethereum Blockchain (BC) such as

the hard key size limitation (256 bit) and the required account funding which is needed to execute on-chain transactions. Furthermore, using a combination of cryptographic principles, the system achieves receipt-freeness on a public permissioned network.

The voting authority (VA) is the entity in Provotum responsible for bootstrapping the system. Provotum 3.0 splits the responsibility of operating an IdM system and access control to the PBB between the VA and the Identity Provider (IdP). It is assumed that the VA runs an IdM system that maintains a list of eligible voters identified by their email address. At the start of a vote, the IdP receives the list from the VA and uses it to authenticate and authorize voters to submit a ballot. The goal of this separation is that the IdP cannot link registered users to the real identity of the voter. The issues and possible attack vectors associated with this design, the use of a centralized IdM system and the use of email addresses as identifiers are analyzed and an improved architecture is proposed. The analysis shows that the need for a new form of digital identity reaches much further than Provotum's e-voting system. Account-based and federated identity models raise concerns in terms of privacy and exhibit weak security promises for the systems that depend on them.

Many security and privacy concerns rise from the usage of email addresses as identifiers because email addresses often contain personal information or can be associated with other online profiles such as a company website, social profile etc. Furthermore, email communication is usually not E2E (end-to-end) encrypted, allowing the email provider to learn when a voter registers on the platform. In addition, email providers often also act as Single-Sign-On (SSO) providers, enabling them to aggregate meta information about its users and create detailed user profiles. As seen in the past [5, 27], target advertising alongside the intimate data about the users can be misused for voter manipulation.

To guarantee a higher level of security and privacy to its voters by design, a radically different architecture for issuing, storing and verifying credentials is proposed. Therefore, the responsibility for each of the three tasks is split among three independent entities. Using randomized identifiers which are maintained on a decentralized registry and linked to cryptographically verifiable credentials, results in more flexibility and a higher level of privacy and security. These properties are enforced by design and align with Provotum's security requirements.

1.1 Description of Work

The goal of this thesis is to analyze Provotum 3.0 in terms of identity management and access control. Potential security concerns are to declare and address. Furthermore, a detailed analysis of the strengths and weaknesses of the different forms of today's digital identities is required. With the obtained knowledge about different identity management (IdM) systems, the goal is to design and develop the best suitable architecture for Provotum 3.0. The new architecture is evaluated in terms of privacy, verifiability and usability.

1.2 Thesis Outline

This thesis is structured as follows. Chapter 2 evaluates the strengths and weaknesses of existing forms of digital identities. Related work is discussed in Chapter 3. Chapter 4 introduces the fundamental building blocks for a decentralized IdM system. Chapter 5 evaluates ProvoTum 3.0 in regards to security and privacy. The processes around issuing and maintaining credentials as well as authenticating and authorizing eligible voters are investigated. Chapter 6 proposes a new architecture that addresses the issues identified in the current implementation. A prototype is developed and its implementation details are stated in Chapter 7. Finally, the new architecture is reevaluated in Chapter 8, improvements are highlighted and ongoing challenges are described for future work.

Chapter 2

Background

The technological advances of the internet also introduced novel mechanisms for digital authentication. In this chapter, three different models of digital identity are explained and analyzed by evaluating their strengths and weaknesses. Since digital identities are not limited to humans, the more generic term *subject* or *resource* is used for the person or item which is to be identified. Three primary roles exist around the issuance, management and verification of credentials:

- **Issuers** are the source of credentials. They underline a claim about the subject in the form of a credential like a passport issuing authority issues a document with multiple claims about the subject.
- The **holder** is responsible for storing the credentials privately and securely.
- **Verifiers** - or also referred to as the relying parties - check whether a credential is valid and grant access to the requested service.

2.1 Centralized Identity Model

The centralized identity model is the form that appeared first on the internet. This form of internet identity is created by registering an account directly with the requested service or application. For this reason, it is also referred to as the account-based identity. The application issues identifiers such as unique username. A username/password combination is used to access the service.

As shown in Figure 2.1, a digital representation of the subject can only exist by creating an account with the organization. The system acts as a TPP. The users trust the system that sensitive information is not shared with other parties. The service provides identifiers and credentials that represent subjects in the application context and enable interactions between them. These credentials belong to the application and the system controls the permissions associated with them as illustrated in Figure 2.3. Even after deleting the account, the data may live on within the organization and is outside of the subject's

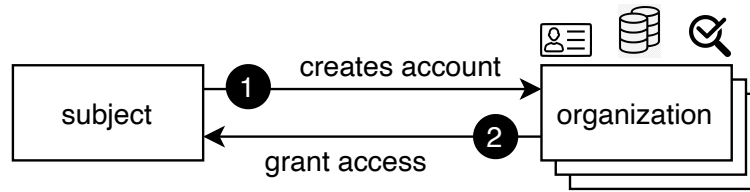


Figure 2.1: Centralized identity model

control. Thus, in the centralized identity model, the application that is providing the authentication service acts as the issuer, holder and verifier.

Problems that occur with an account-based identity model:

Data Breaches Centralized databases with aggregated personal information are popular targets for attacks. As seen in the past [19], even the largest technology companies have difficulties protecting their systems against data breaches.

Credential Management Creating multiple accounts for different services imposes a challenging task on the user of remembering many username/password combinations. Some services enforce different rules about passwords such as special characters. These organizations suggest not to use the same username/password combination for other services.

Portability The identity created with the account cannot be reused for other services. Although this is a desirable feature from a privacy point of view, the user experience is directly affected by it.

2.2 Federated Identity Model

With the drawbacks of the account-based model (Section 2.1) in mind, a new form of internet identity was invented. In the context of the federated identity model, a TTP - also known as the Identity Provider (IdP) - is introduced. The responsibility of the IdP is to issue identifiers and credentials that can be used across multiple services without having to manage multiple username/password combinations. The IdP only shares data with another organization after the subject grants permission to do so. However, it is out of the subject's control to monitor the data which is shared with other companies. Thus, the IdP acts as the issuer, holder and verifier. The collection of all the services that use the same IdP is called a federation and gives this form of digital identity its name.

To simplify the interoperability of online services three standards have evolved SAML [22], OAuth [25] and OpenID Connect [21]. Social login buttons have become the de facto standard for most internet services today. Figure 2.2 illustrates the role of the IdP in an OAuth 2.0 authentication flow.

Although the end-user benefits from not having to remember an abundance of logins, and the authentication and authorization process is simple to integrate for application developers, this convenience has other disadvantages:

Data Breaches Compared to the centralized identity model, the data stored at the IdP is even more attractive for an attacker. Large IdP such as Google and Facebook represent some of the biggest honeypots for cyber-criminals since they create detailed user profiles from the metadata generated from the authentication requests.

Credential Management The end-user has to remember fewer credentials. However, when multiple IdPs are supported by the service, the end-user must remember which IdP was used to initially sign up.

Portability Although there is only one credential necessary within a federation, there is no IdP that is supported across all services. Credentials cannot be reused outside of the federation. If the subject decides to leave an IdP, all the logins on other applications associated with this credential are lost with it creating a lock-in effect.

Privacy The IdP can monitor the online behaviour across multiple services and create very detailed user profiles from their online activity and the metadata that is attached to it. This data can be used for target advertising and is often considered a win for both parties. However, as seen in the past [5, 27], this data can also be used to unknowingly manipulate people.

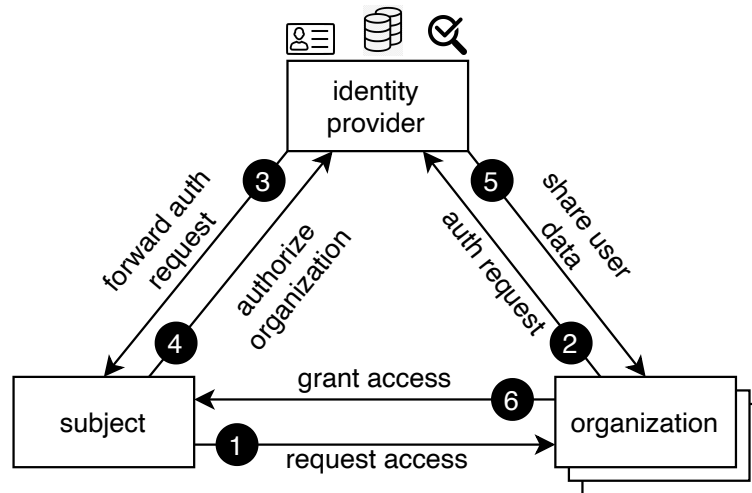


Figure 2.2: Federated identity model in the example of OAuth 2.0

In March 2021, the population of Switzerland voted against a national digital identity that is issued and managed by private IdPs. Several architectural uncertainties lead to a rejection. This form of identity does not enforce *privacy by design*. Instead *privacy by trust* is applied where users rely on the promises of private institutions that sensitive data is not shared with third parties. [6]

2.3 Decentralized Identity Model (Self-Sovereign)

This form of online identity is not account-based and addresses the issues with the federated identity model as highlighted in Section 2.2. The goal is to have an independent

entity for each of the three different roles. This third iteration of digital identity is inspired by the identity system of the real world where an identity is based on a direct relationship between the subject and its peers. The responsibility of issuing, holding and verifying identities is no longer the duty of a single party. Figure 2.3 illustrates how these responsibilities are distributed among the peers. This form of internet identity has recently become more popular under the name of Self-Sovereign Identity (SSI).

The authentication flow is best described as part of a real-world example. For this scenario, there is the government (issuer), Alice (holder) and a liquor store (verifier or relying party). Alice wants to buy a bottle of wine. The liquor store must verify that Alice is eligible by law to buy alcoholic beverages. Firstly, the government as well as Alice create a key pair each. From this key pair, an identifier is derived and registered publicly. In a secure channel or a public credential registry (Section 8.3.4), the government communicates this identifier with the liquor store. Alice identifies herself to the government and in return, the government issues a signed statement to Alice that contains her age. This claim is stored locally in Alice's wallet and is only known to the government and Alice. In the checkout procedure at the liquor store, Alice can present this claim to the cashier. With the help of the verifiable data registry, the liquor store can resolve the associated public keys from the identifiers that are part of the signed claim. The liquor store can verify this claim without the need of contacting the government. As a result, the government does not learn when Alice uses the credential. Furthermore, with the help of a challenge-response mechanism, Alice proves that she is in control of the private key that is associated with the claim.

Additionally, it is possible to include biometrics within the claim. Alice would then have to disclose the signed image to the cashier which then can be used by a face comparison by the cashier. The entire process at the liquor store occurs without the issuer noticing since the government is not involved in the disclosure process.

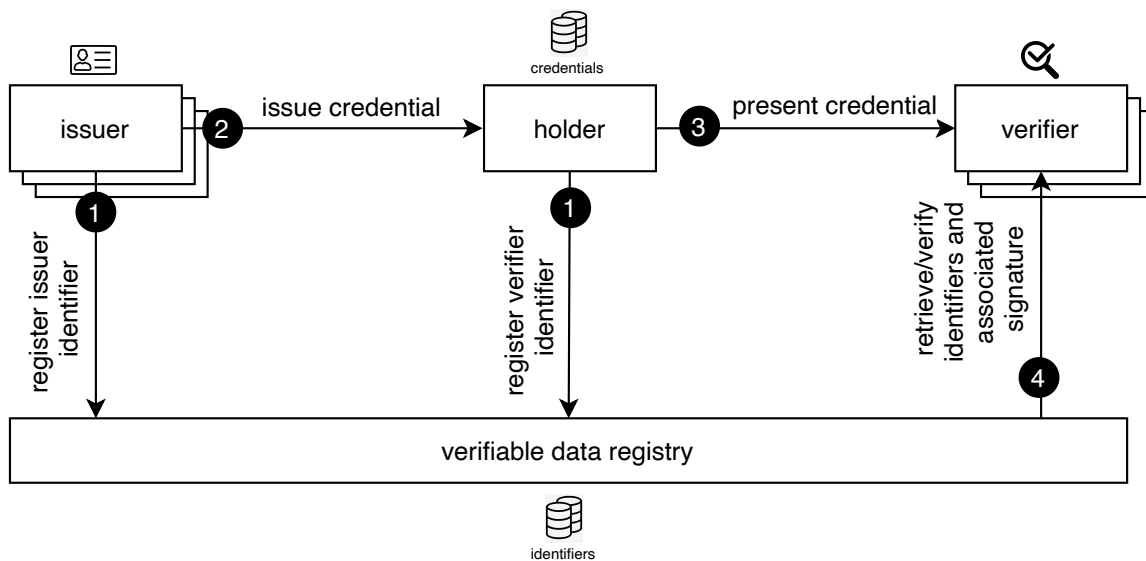


Figure 2.3: The decentralized identity model splits the roles of issuing, holding and verifying among three separate entities.

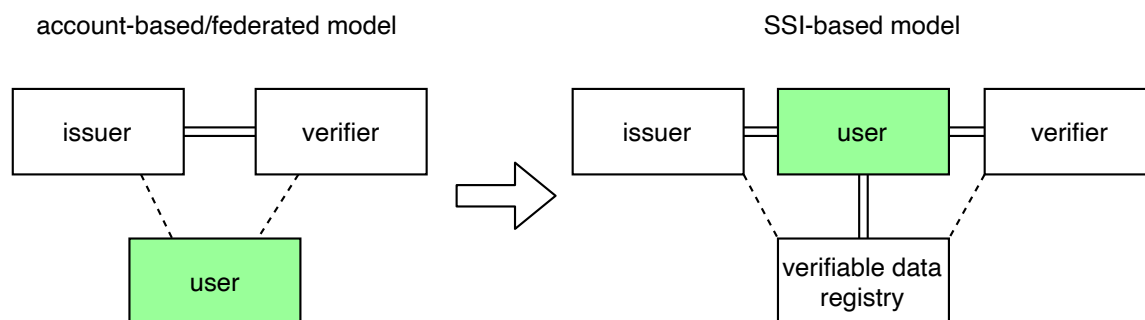


Figure 2.4: The locus of control that happens in the transition from account-based identity models to the self-sovereign identity model [2]

The decentralized identity model improves the account-based and federated identity model in the following aspects:

Privacy Decentralized public key infrastructure (DPKI) enables the subject to create a direct connection to its peers. Using DLT removes the need for a TTP to verify that a claim about the subject is valid since the associated keys are publicly available. There is no need for a connection between the issuer and the verifier. Therefore, no other node in the network except the two connected peers learn about the authentication request and the data that is shared in the process. Thus, a higher degree of privacy is guaranteed in comparison to the centralized and federated identity models. Figure 2.4 illustrates how the control shifts to the edges of the network by putting the individual at the centre of the identity model. By design, the subject is in control of storing and disclosing credentials.

Data Breaches IdPs only act as credential issuers and not verifiers and therefore, they cannot track the online activity of a user. Since authentication requests are not logged by the identity issuer, the system is less vulnerable to cyber attacks that exploit large amounts of sensitive user data. To access a collection of credentials about a subject, the attacker must be able to bypass the security restrictions of a voter's wallet.

Credential Management Instead of username/password combinations for authentication, DPKI is applied mitigating the risk of possible attacks such as phishing attacks [12], dictionary attacks [20], credential stuffing [15]. Private keys are maintained in a wallet. It is the wallets responsibility to enforce protection against the leakage of cryptographic material.

Portability Similar to the federated identity model, the decentralized identity model allows the subject to use the same form of identification across multiple applications. It improves the authentication process since the user must not remember which IdP was used to initially sign up for each application.

One of the challenges of a decentralized identity model is to create a user-friendly wallet recovery mechanism. Even though physical identity cards cannot be restored, people are

used to a simple restoring mechanism for their digital identities. Therefore, it is desirable to have a recovery mechanism for decentralized identifiers with a similar degree of ease of use. When a user loses the private key that corresponds to the digital identity and no backup was created, the identity is lost and cannot be restored. However, there are mechanisms developed that enable a recovery schema based on a group of trusted entities which is often referred to as social recovery with Shamir Secret Sharing [35].

Chapter 3

Related Work

A standardized technology around decentralized identity is necessary such that competing and complementary implementations can emerge and communicate with each other. Multiple organizations are working towards a common specification. Most of the specifications referenced in this section are ongoing and yet to be finalized. This chapter covers relevant projects and standards in the SSI ecosystem.

3.1 World Wide Web Consortium (W3C)

The W3C is specifying international standards for the internet such as CSS, HTML, XML, etc. The consortium is organized by working groups that develop, maintain and communicate the outcomes of their field of research. The W3C Credentials Community Group (CCG) focuses on the creation, storage, presentation and verification of credentials including SSI. The CCG has published a working draft [23] for Decentralized Identifiers (DIDs) and Verifiable Credentials (VCs) which is the first iteration in a four-step maturation process [36]. The goal of the specification is to develop a standard that is as independent as possible of the under-laying infrastructure. Furthermore, the the data encoding format JSON-LD [29] (Section 3.3) which is heavily used in SSI protocols is also specified and maintained by the W3C.

3.2 Decentralized Identity Foundation (DIF)

The Decentralized Identity Foundation (DIF) is also divided up into working groups that are scoped by functional areas in the scope of digital identity. Their contributions in the area of the discovery and the registration of digital identities with the Universal Resolver¹ [26] and the Universal Registrar² provide implementations that allows the creation and resolution of digital identities across multiple networks. Also, the DIDComm Messaging

¹<https://dev.uniresolver.io>

²<https://uniregistrar.io>

specification [10] is relevant in the context of a standardized means of authenticated message passing between decentralized identities. Furthermore, the Authentication Working Group created the Self-Issued OpenID Connect Provider (SIOP) specification [32] which enables the usage of the SSI building blocks in the OpenID Connect authentication and authorization process and thus being backwards compatible with most existing application logins used today.

3.3 JSON-LD Data Encoding Standard

JSON-LD is a data encoding format that extends the JSON format. This standard is also maintained by the W3C consortium [29]. The intended purpose of this extension is the ability to represent digital objects and link these resources in a standardized manner. In the context of data sharing, data vocabularies or schemas are necessary tools for representing data accurately from the point of creation to sharing and verification. If every application chooses to represent data differently, it creates incoherency between the entities.

Furthermore, a JSON-LD document can be serialized in the same way as JSON documents are parsed. The JSON-LD data format specifies a number of reserved keywords that are part of the language³. The most relevant keywords for understanding the following sections explained in Table 3.1.

A JSON-LD processor is a program that serialises JSON-LD documents and creates a context across multiple documents. There exist multiple techniques on how the context of a document can be communicated. The simplest form is to host the JSON-LD document on the same link that is specified in the context property. However, more complex resolving strategies exist and as explained in Section 8.3.3

Keyword	Description
@id	This reserved token indicates that the node that is described is uniquely identified by this property.
@type	Base on the properties of a document, the type can often be implied. However, in order to mitigate ambiguity the type keyword can be incorporated as shown in Listing 1 on line 9. An array of types indicate that a document is composed of multiple types. Furthermore, this token is used to represent a typed value which is a value with an associated data type.

³<https://www.w3.org/TR/json-ld11/#syntax-tokens-and-keywords>

@context This is a reserved keyword that defines a common vocabulary that is used to describe the document. It maps terms to Internalized Resource Identifiers (IRIs) or nested structures. Terms are non-reserved strings in the JSON-LD namespace while IRIs identify resources. The context is used to serialize the rest of the document. Listing 1 illustrates how the context can be defined. Line 3 defines that `name` can be used as a shorthand notation for the following IRI. The same applies for the property `image`. Additionally, line 6 specifies that this property must be identifiable as an identifier that is an IRI.

Alternatively, to the illustrated in-line context definition, which does not require a connection to the internet to serialise the document, the context can be referenced by an URL which hosts the context definition. Furthermore, the context may contain a list of context to indicate that document is composed of multiple external context definitions.

Table 3.1: Reserved tokens in the JSON-LD namespace that are relevant for this paper

```
1 {
2   "@context": {
3     "name": "http://schema.org/name",
4     "image": {
5       "@id": "http://schema.org/url",
6       "@type": "@id"
7     }
8   },
9   "@type": "http://schema.org/Person",
10  "name": "Alice Keller",
11  "image": "http://alice.keller.org/"
12 }
```

Listing 1: Example JSON-LD document with a in-line context definition

3.3.1 Schema.org

Schema.org is founded by Google, Microsoft, Yahoo and Yandex and its community is working towards a unified set of schemas for structured data on the internet, web pages, email messages and more. The vocabulary can be used with different encodings including JSON-LD documents and covers entities, relationships between entities and actions. Schema.org is a popular tool to improve the ranking of a website in terms of Search Engine Results Pages (SERP). Adding metadata in the form of well-understood Schema.org vocabulary to a website helps search engines to understand the content and its relationship to other websites. In SSI, Schema.org is used to indicate the type of credentials that are issued such that the issuer, holder and verifier understand the context.

3.4 Veramo

Veramo is an open-source set of modular libraries and APIs for verifiable data and SSI. At the time of writing this thesis, Veramo supports the creation of Ethereum-based and web-based DIDs. Also, a library for issuing JWT-based VC is available. Furthermore, a custom Selective Disclosure Request (SDR) with Verifiable Presentations (VP) workflow is present. This process is custom to Veramo and uPort [17] and does not fully align with the W3C specification. There are multiple ways to create and interact with a Veramo agent: *(i)* CLI program, *(ii)* React Native for mobile clients, *(iii)* Nodejs client for backend applications and *(iv)* a cloud-agent that is hosted on a server and exposes its functionality through a RESTful API.

Chapter 4

Self Sovereign Identity (SSI) Primitives

SSI is built on two pillars of standardization. Decentralized Identifiers (DIDs, Section 4.1) and their cryptographic counterpart Verifiable Credentials (VCs, Section 4.2) enable a decentralized and privacy-preserving form of digital identity. This section explains their structure, function and application.

4.1 Decentralised Identifier

DIDs globally unique identifiers and are at the core of SSI. SSI introduces this new type of identifier that exhibits two unique properties: (i) it can be created and managed without a centralized certificate authority and (ii) is bound to PKI such that the ownership of the identifier can be proven without a TTP. The binding problem of PKI illustrates in Section 4.1.1 the need for a new type of identifier with the mentioned properties.

4.1.1 Binding Problem in PKI

DIDs solve the fundamental binding problem that exists in PKI. When creating a PKI key pair, the private and public parts are cryptographically bound to each other. However, the controller which maintains the key pair is often non-digital. As a consequence, non-digital controllers are represented with an identifier in the digital world. Usually, this identifier is not strongly bound to the key pair and thus, creating two fundamental challenges. (i) The introduced identifier is not strongly bound to the public key and (ii) the identifier is not strongly bound to the controller as illustrated in red in Figure 4.1.

As an example, these two binding problems also exist in SSL/TLS, where digital certificates bind an IP address to a public key. As a result, the controller is able to prove ownership of the certificate and thus, a trusted connection to the IP address is established. However, certificates are issued by a certificate authority that acts as a TTP. This solution is centralized and embodies a SPOF.

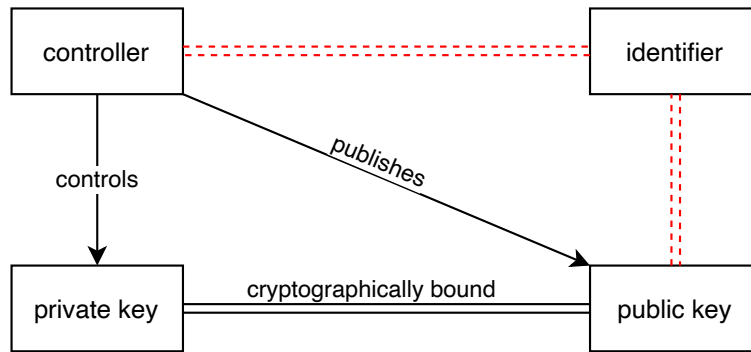


Figure 4.1: DIDs as self-certifying identifiers enable a strong binding between the public key, identifier and controller

DIDs are self-certifying identifiers meaning that the ownership can be proven without the need for a TTP. A self-certifying identifier is derived from the public key. Thus, the binding between the identifier and the public key is as strong as the binding between the public key and the private key solving the binding issue *(i)*. Furthermore, since the identifier correlates with the public key, the ownership over the identifier can be proven with the private key. Therefore, only the controller can make this proof and thus, the controller is strongly bound to the identifier solving the binding issue *(ii)*.

By binding a persistent identifier to a public key, a key rotation for that identifier is not possible. A new key pair would also result in a new identifier. However, key rotations are desired as the controller may need to adapt to newer and more secure encryption algorithms. This is solved by introducing the DID document (Section 4.1.4), which holds meta-information about the identifier itself such as the controlling public key and encryption algorithm as illustrated in Figure 4.2. Updating the associated key in this document requires the previous key to approve this change and does not modify the identifier itself. All updates together form a chain of trust. If each change is recorded publicly on a decentralized data registry, anybody can verify if the associated key controls the identifier and therefore, the need for a TTP is removed.

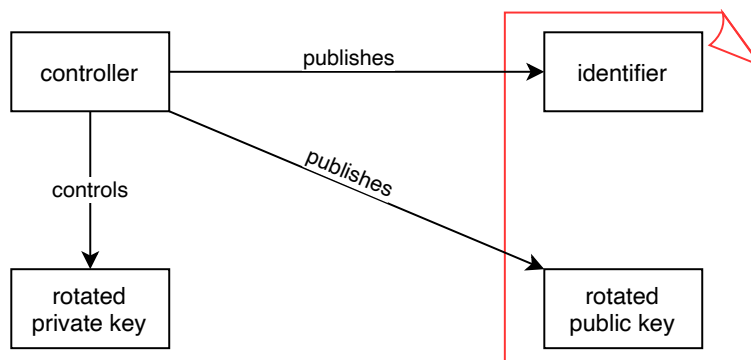


Figure 4.2: DID documents enable holders to execute a key rotation without changing the identifier

4.1.2 DIDs in the Context of Other Identifiers

To compare DIDs to other identifiers, the following description summarizes the difference between existing types of digital identifiers.

Uniform Resource Identifier (URI) - Web Identifiers The W3C has adopted the IETF standard (RFC 3986) [4] of URIs for identifying any type of resource on the internet. A resource is any object from the real world that can be identified such as a person or organization or from the digital world such as an image or a web page. A URI is defined as a sequence of characters in a specific layout that makes the string globally unique.

Uniform Resource Locator (URL) - Network Location A URL is a type of URI that is used to locate a representation of the associated resource on the internet. A representation is anything that describes the resource. In the example where a person is the resource, only a representation of that person can exist on the internet. Thus, the URL pointing to a web page, file, image is a location pointer to the resource representation. A pointer to a resource is subject to change.

Uniform Resource Names (URN) - Persistent Name Since a URL for a resource representation can change, a persistent identifier - also referred to as permanent identifier - is used to identify the abstract resource itself. This kind of identifier is designed to never change and in the context of the internet referred to as a URN. URNs and URLs are both subclasses of URIs.

Like URNs, a DID is also globally unique and also identifies a resource. Additionally, a DID is resolvable for meta-information associated with the subject similar URLs. The resolved information conforms to a standardized document format such that the retrieved information about the subject is understood. Thus, a DID combines characteristics of both URLs and URNs. The design goals defined by the W3C CCG in the DID specification [23] are summarized in Table 4.1.

Property	Description
Persistence	The identifier must never change.
Resolvability	The identifier must be resolvable in order to discover metadata about the subject.
Verifiability	The identifier is controlled by DPKI. By including cryptography in the generation process of the identifier, the ownership of the identifier can be cryptographically verified by the holder of the associated key pair. The associated public key is included in the resolvable metadata along with metadata about the encryption algorithm used for generating the key pair.
Decentralization	The cryptographic verifiability of the identifier removes the need for a registration authority. Furthermore, the entire life-cycle from creating, resolving, updating and disabling the metadata associated with the identifier can be managed without the need without requiring a centralized entity.

Interoperability	The identifier must be globally unique regardless of the under-laying network that is used to manage the identifier. Thus, the namespace of DID is shared across multiple networks.
------------------	---

Table 4.1: Design goals for decentralized gloabl identifier

To achieve the design goals from Table 4.1, the W3C standard proposes the architecture shown in Figure 4.3. The fundamental building blocks in the DID architecture are explained in the following sections and summarized in Table 4.2.

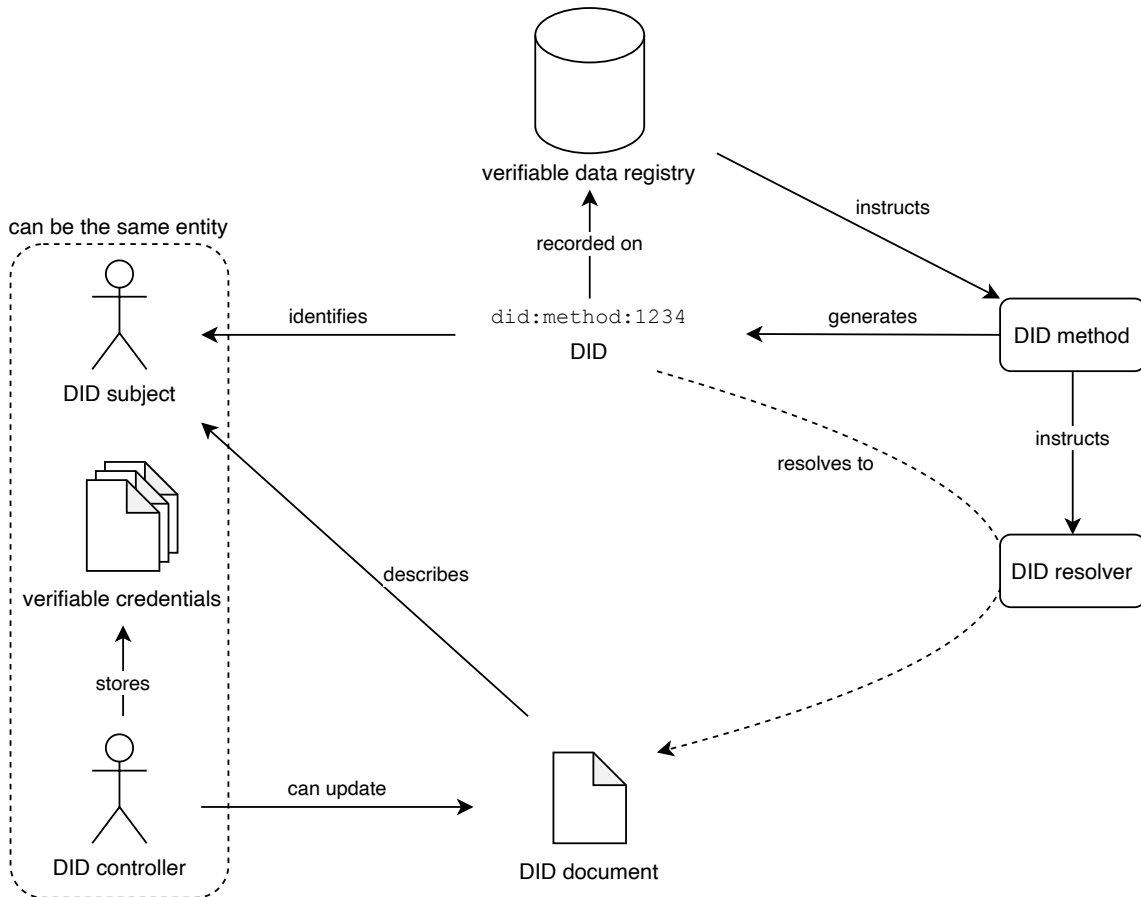


Figure 4.3: Overview of the DID architecture proposed by the W3C CCG

Element	Definition
DID	A DID is the URI that identifies a resource (also called the DID subject).
DID subject	The subject or resource is the entity identified by the DID.

DID controller	<p>The controller of a DID is the entity that is able to make changes to the DID document. A DID document may have multiple DID controllers. The DID subject may not always be the part of the set of DID controllers.</p> <p>In the example of an infant (DID subject), it makes sense that the parents (DID controllers) are in maintaining the DID and DID document and not the infant itself. The parent can update the DID document at any time to transfer the ownership of the DID.</p>
DID document	<p>By resolving a DID, a DID document is discovered. This document contains information associated with the DID subject such as verification methods and services that are relevant to interact with the DID subject. The contents of the DID documents can be serialized according to the definition found in the specification [23]. Every DID resolves exactly to one DID document and can be updated by the DID controller.</p>
DID method	<p>A DID method defines how a particular type of DID and its associated DID document can be created, resolved, updated and deactivated. The DID method depends on the underlying data registry that is used (see Section 4.1.3).</p>
DID resolution	<p>The process of resolving a DID document for a given DID is called DID resolution.</p>
DID resolvers	<p>A program that can find the DID document for a given DID.</p>
Verifiable Data Registry	<p>A DID must be resolvable to a DID document. The verifiable data registry is the system or network that supports a particular DID method and connects the DID with the associated DID document. The DID architecture is specified such that different types of registries are supported. Section 4.1.3 explains different kinds of networks used for registering DIDs.</p>

Table 4.2: Components of DID architecture suggested by the W3C CCG

4.1.3 The Scheme, DID Method and DID Method-Specific String

A DID consists of three parts separated by colons as shown in the example in 4.1: *(i)* scheme, *(ii)* DID method and *(iii)* DID method-specific string. The scheme is the first part and indicates that the URI is to be interpreted as a decentralized identifier.

DIDs are network independent. Therefore, the DID method indicates the underlying network and makes DIDs unique across multiple networks. A standard specification is created by the W3C Working Group for implementing a new DID method [30]. At the time of writing this thesis, there are more than 70 DID method names registered in the DID Specification Registries [31]. Every DID method has its implementation of creating, reading, updating and deactivating its identifiers. These are the so-called DID operations. Their implementation depends on the underlying network. Thus, generic statements about the

implementation of the four operations cannot be made. For example, distributed ledger-based DIDs usually require an on-chain transaction to update their DID document while web-based DIDs can be updated by uploading a new DID document to the storage bucket where the DID document is hosted. An overview of different types of DIDs is shown in Table 4.3. As shown in the table, DLT is not required to build an SSI infrastructure. However, the benefits of registering DIDs on a public distributed ledger are the data availability, immutability, security and trustless network assumptions given by the characteristics of DLT.

The third part of a DID consists of the unique identifier in the namespace of the given DID method and is called DID method-specific string. It is typically generated using random numbers and cryptographic functions.

In the example 4.1, the DID method indicates that this DID must be resolved through the Ethereum Rinkeby testnet. The DID method-specific string is an Ethereum address. Table 4.3 highlights the different types of DIDs.

$$\underbrace{\text{did}}_{\text{Scheme}} : \underbrace{\text{ethr:rinkeby}}_{\text{DID Method}} : \underbrace{\text{0xa8b4d9327e0776ad3765fb3baf6811e6f1390271}}_{\text{DID Method Specific String}} \quad (4.1)$$

Type	Description
Ledger	Maintains DIDs on the main chain of a BC or DL. <code>did:ethr:0xb9c5714089478a327f09197987f16f9e5d936e8a</code> <code>did:sov:2wJPYULfLLnYTEFYzByfUR</code> <code>did:btc:12dRugNcdxK39288NjcDV4GX7rMsKCGn6B</code>
Layer 2	With the increased transaction costs and slow execution, layer 2 solutions on top of the base layer enable faster execution times at lower transaction costs or no fees without sacrificing security. This can be achieved by bundling multiple DID operation in batches and only store the proofs on the main chain. Crypto-economics are also required such that network operators are not incentiviced to cheat. <code>did:ion:EiAnKD8jfdDcZUjAbRgaThBrMxPTF0xcnfJhI7Ukaw</code>
Peer	Identifiers that are not recorded on a globally shared registry, can be established directly within a small set of peers. <code>did:peer:EiAnKD8jfdDcZUjAbRgaThBrMxPTF0xcnfJhI7Ukaw</code>
Static	Static identifiers can only be created and resolved, but not updated or deactivated. They can be algorithmically be resolved without requiring any data other than the DID itself. This is best described as a wrapper around a public key. <code>did:key:12dRugNcdxK39288NjcDV4GX7rMsKCGn6B</code>
Others	The DID specification is flexible enough such that other forms of DID methods can be created. DIDs on other networks such as git, IPFS or web exist. <code>did:git:625557b5a9cdf399205820a2a716da897e2f9657</code> <code>did:ipld:12D3KooWMHdzrcwpjbdRZs5GGqERAvcgqX3b5dpuPtPa9ot69yew</code> <code>did:web:uport.me</code>

Table 4.3: Types of DIDs

4.1.4 DID Document

The DID document contains information about its associated subject. In theory, it is possible to attach arbitrary information about the subject to the DID document. However, this document is publicly accessible and for privacy reasons, only the minimum necessary information should be incorporated into the document. This includes one or more public keys that are used to verify the ownership of DID during an authentication process. Furthermore, services associated with the subject can be attached to the document. A wide range of protocols can also be incorporated such as an HTTP endpoint to a cloud agent which provides an interface for exchanging information directly between peers in a private and secure channel (DIDComm Messaging [9]). The data format for DID documents is JSON-LD (Section 3.3).

4.1.5 DID Resolution and DID Resolvers

The process of resolving a DID document for a given DID is called DID resolution. In the context of the DID method, it represents the read operation. The DID resolver is a program that fulfils such a request. Thus, the DID resolver must be able to connect to the under-laying data registry in which the DID was created. Each new type of DID comes with a new implementation of a DID method and a DID resolver.

The Universal Resolver [26] is a program that can resolve many different types of DIDs from different underlying networks. It is open-source and available as a web application¹ for testing but can also be deployed using *docker-compose*².

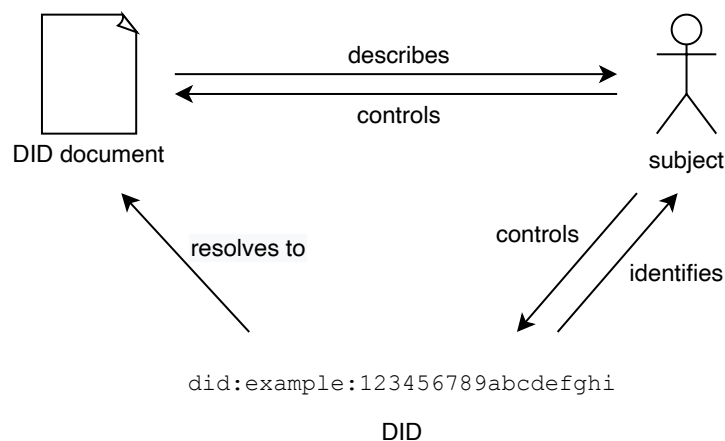


Figure 4.4: Relationship between identifier, metadata and the subject

¹<https://dev.uniresolver.io>

²<https://github.com/decentralized-identity/universal-resolver>

4.1.6 DID Auth

The goal of DID Auth is to cryptographically authenticate the subject and create a trusted connection between the holder and the verifier. This is achieved by proving control over a DID. The verifier creates a challenge in the form of a random string that is subsequently signed by the holder with the private key associated with the DID. This process can be implemented with different protocols and architectures³. One of the preferred methods is using Json Web Tokens (JWT). Other possible protocols include OpenID Connect [18], TLS handshake or HTTP.

4.2 Verifiable Credential (VC)

VCS account for the second pillar in SSI besides DIDs. The goal of VCs is to create a common interface for issuers, verifiers and holders to exchange privately and securely verified claims about a subject.

A VC is used to assert personal information. It contains one or multiple attributes about a subject and is usually issued in the form of a JSON-LD document (Section 3.3) whereas the subject is commonly represented as DID. Other attributes of this document are the issuer, the context and type of the credential as well as the signature and meta-information about the proof itself. Such claims are signed by the private key that is attached to the issuer's DID. As explained in Section 4.1.1, a DID is bound to key pair allowing anyone to verify the authenticity of the VC. The verification process only requires resolving the issuer's DID and validating the signature against the resolved public key.

Unlike DIDs, VCs are not stored on a public ledger. Instead, they are stored privately in the holder's wallet since they contain sensitive information.

There are many advantages of digitally signed claims over traditional physical credentials. VCs cannot be cloned or counterfeit and can only be stolen if the attacker can access the most sensitive information which is the private key in the holder's wallet. Furthermore, the holder of the credential has control over the disclosure of the credentials increasing privacy protection. Also, they are more portable than bulky plastic cards, and cheaper to produce.

Verifiers trust the issuer of VC and thus, together they form a so-called *trust network*. To establish a trust relationship between entities, the DID must be exchanged in a secure channel beforehand. Credential registries and governance frameworks can help communicating trusted DIDs. However, these concepts are not part of the VC or DID specification and are further discussed in Section 8.3.4.

³<https://github.com/WebOfTrustInfo/rwot6-santabarbara/blob/master/final-documents/did-auth.md#architectures>

4.3 Selective Disclosure Request (SDR)

An SDR is issued when a verifier asks the holder to reveal a set of previously issued VCs. This request is usually a JSON-LD encoded document or a JWT that contains information about the credential context and the accepted issuers along with meta-information about the request. An SDR can contain multiple credentials that need to be disclosed and the holder has to approve each of the requested claims individually in the form of a Verifiable Presentation (VP).

4.4 Verifiable Presentation (VP)

Upon receiving an SDR, the holder can either approve or discard the request. If the holder is willing to disclose the requested information, another JSON-LD encoded document or JWT is generated. It combines all the requested VCs into a single document. These VCs can originate from different issuers. This document is signed and sent to the verifier.

To only disclose minimal information, there exist two strategies: *(i)* atomic issuance *(ii)* Zero-Knowledge Proof (ZKP) enabled credentials. With atomic issuance, every VC only contains a single claim. This guarantees that a VP can be constructed from multiple atomic claims. For example, a passport has the following four properties: name, date of birth, country, gender. Applying the atomic issuance strategy results in four separate atomic VCs. When a verifier requests the disclosure of the holder's age, only one of the attributes is included in the VP which would not be possible if they were issued as one VC.

Using ZKPs enables the holder to only reveal certain fields of the VC or to prove information about a claim without revealing all the attributes of the VC. This strategy is more complex and does not work with every type of signature. Multi-message digital signature schemes (such as BBS+ signatures) are used for creating ZKP in the context of linked data such as JSON-LD. These signature schemes allow an issuer to sign an array of messages, rather than a single string. This puts the holder in control of disclosing only a subset of the values in a VC while maintaining verifiability over the VC. [2, 33, 34]

4.5 Fork of the Verifiable Registry

If a decentralized verifiable data registry faces a hard fork, registered identifiers exist on both networks. This can lead to inconsistencies. The DID resolver has to decide which chain is considered as the source of truth. In the case where no consensus exists as to which fork is the mainnet, the DID resolver has to be adjusted accordingly. For example, the Ethereum community solves this issue by having a DID resolver for each chain ID. In the scenario of an Ethereum fork, the DID resolver⁴ would reject any DID that solely

⁴<https://github.com/decentralized-identity/ethr-did-resolver>

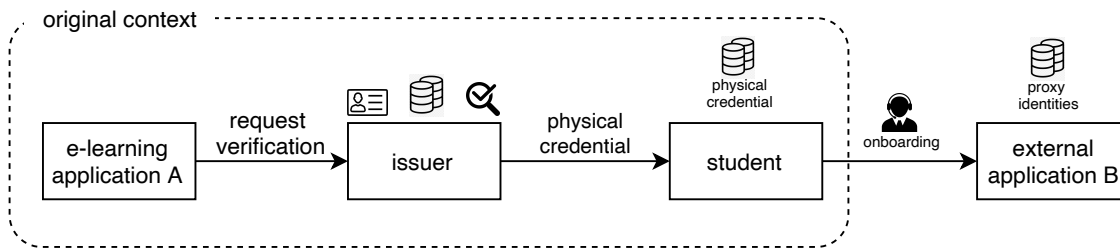


Figure 4.5: Original identity context ends whenever a proxy identity is created.

uses the `ethr` DID method (without a network identifier) as it is not up to the resolver to decide on which chain is valid. Thus, it is recommended to use the chain identifier as part of the DID to mitigate data inconsistencies as for the Ethereum Rinkeby testnet as shown in Listing 4.2.

```
did:ethr:0x4:0xpubkey (4.2)
```

4.6 Identity Contexts

The following definition describes the concept of cloned identities (proxies) and the meaning of the original identity context. These definitions are used throughout the thesis to reason about the data dependency of credentials among different systems.

Identity Context The original identity context is defined by the issuer of a credential. In Figure 4.5, application A can request a verification from the issuer directly. For example the original identity context of a university student is defined as all the systems that directly interact with the IdM system of the university. This can be an e-learning platform from the university itself.

On the other hand, in the example where a student registers for an student discounted licence at application B, the student card is shared or the student email address is verified and in return a new user account is created within the application B. This is also called an identification or onboarding process. This new user account is loosely coupled to the original identity context meaning that a change in the original context may not be noticed in application B.

Proxy Identity A proxy identity is an account with credentials that are loosely coupled to other identity contexts. The verified claims associated with that account originate from another identity context. In Figure 4.5, the newly created user account in application B is the proxy identity.

Chapter 5

Provotum 3.0 Security Analysis

The security and the privacy of a REF system are crucial for a fair execution of the voting process. There are several highly motivated adversaries with the intention to counterfeit identities, tamper with the tally or cause system outages with DoS attacks. This chapter explains the architecture of Provotum 3.0, its stakeholders and their responsibilities in the system. Subsequently, a security analysis on the eligibility verification process of Provotum 3.0 [3] is conducted. Only the relevant components for IdM as well as the authentication and authorization process are reviewed. In this context, the trust assumptions, threats and vulnerabilities are evaluated in terms of security and privacy. The threat model is summarized in Table 5.1.

5.1 Provotum 3.0 Architecture

Provotum 3.0 is the result of a third iteration of a fundamentally improved redesign of the application. It is a Distributed Ledger (DL) based system that uses the concept of a Public Bulletin Board (PBB) [13]. It addresses the architectural limitations of its predecessor Provotum 2.0 [14] where the hard key size on the Ethereum BC is limited to 256 bit and the voters' accounts must be funded to execute an on-chain transaction. Using a combination of cryptographic principles, Provotum 3.0 achieves receipt-freeness on a public permissioned network. It continues to use a Proof-of-Authority (PoA) consensus algorithm, however, the underlying DL infrastructure has been changed from Ethereum to Substrate.

Figure 5.1 shows the components of the system and its stakeholders. The following list explains each in more detail:

- The Substrate PoA DL is the system's **PBB** and enforces the protocols security and privacy rules. The exposed API enables all other stakeholders to read the state of the PBB and send transactions to the network.

- **Voters** are qualified entities participating in the election. They interact with the IdP during the authentication/authorization process and the randomizer to make the vote receipt-free and submit the ballot directly to the PBB.
- The **Identity Provider (IdP)** is a TTP that is responsible for the eligibility verification. Upon successful authentication, voters submit a blinded public address of their wallet. The IdP signs the address without seeing it in plaintext and returns it to the voter. The signature of the IdP allows anyone to verify that a ballot was approved by the IdP. Thanks to the blind signatures, the IdP is not able to connect any of the ballots to the real identity of the voters. This mechanism guarantees a privacy-preserving authorization process.
- The **randomizer** is responsible for achieving receipt-freeness. This is a countermeasure against vote selling. By adding non-determinism to the process of creating a ballot in the form of a blinding factor, the voter's ballot is randomized (or blinded) and therefore, a voter is unable to reproduce the same ballot which is published on the PBB. The randomizer does not learn anything about the content of the ballot since the blinding is performed without knowledge of the plaintext.
- The **Voting Authority (VA)** is responsible for bootstrapping and orchestrating the Provotum system. It coordinates the distributed key generation (DKG) with the sealers, creates the questions of the elections and closes the election by signalling to the sealers to start tallying. Furthermore, the VA creates a list of eligible voters and communicates it to the IdP during the bootstrapping phase. This part is not implemented in Provotum 3.0 but in theory, the VA is responsible for maintaining the set of entitled voters.
- **Sealers** are DL nodes validating and appending transactions to the Provotum chain. During the bootstrapping phase (pre-voting), sealers also participate in the DKG which is used to encrypt the ballots. In the tallying phase (post-voting), sealers will collaboratively decrypt the final tally.
- The general **public** is interested in having fair elections. Since the DL is public, anybody can verify the state of the PBB.

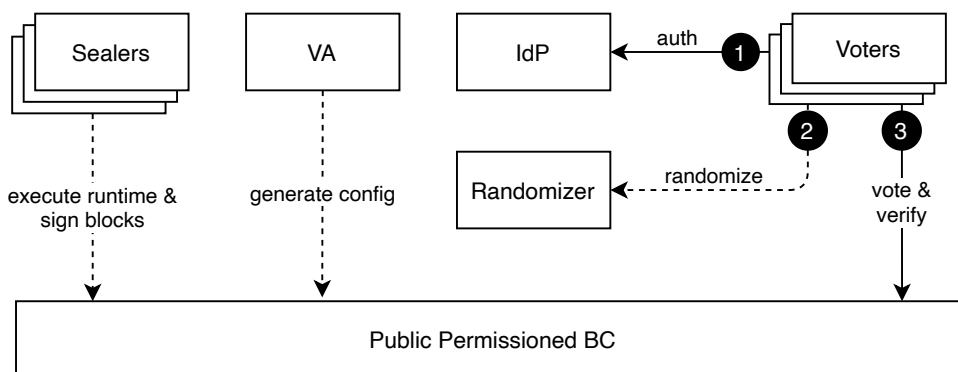


Figure 5.1: Provotum 3.0 stakeholders

5.2 Eligibility Verification Process

Before being authorized to submit a ballot, a voter must go through the following sequential steps. Each step is analyzed in a separate section below and summarized in Table 5.1:

1. Registration
2. List Communication
3. Authentication
4. Authorization

The paper does not mention what type identifier is used to determine an eligible voter. The identifier must have a property such that the voter can prove ownership of that identifier. The prototype suggests to use email addresses as the decisive identifier. An alternative to email addresses is to use phone numbers that work in a very similar way since the ownership can also be proven with a One Time Password (OTP). But for the rest of the thesis it is assumed that email addresses are used as designed in the Provotum 3.0 prototype.

Furthermore, it is unclear how the IdP knows the correct passwords of the users. It is assumed that the VA does not share the list of passwords that are used to login to the IdM system of the VA. As a consequence, the only factor of authentication is the email address. It is assumed that a voter can use the registered email address to obtain a OTP and set a password.

5.2.1 Registration

The VA defines and maintains a list of eligible voters, usually in the form of a centralized database. The list of qualified voters is passed to the IdP in the bootstrapping phase. Depending on the context of the identity management (IdM) system, an entry in that system often requires the user to present several claims about themselves.

In Provotum 3.0 the means of authentication is defined as a user account in the form of an email address and password. For opening such an account and becoming an eligible voter, the user must undergo an onboarding procedure. Although this is conceptually not defined in Provotum 3.0, a standard registration process at an administration office of a Swiss canton is implied. In this process, users must present valid claims about themselves and the identifying person must validate those claims.

For example, when registering at the municipal administration office of Zurich, several documents must be presented such as the passport, certificate of residence, health insurance card and rent contract for the apartment¹. If all the documents are valid, a loosely

¹<https://www.stadt-zuerich.ch/prd/de/index/bevoelkerungsamt/umziehenmelden/zuzug.html>

coupled proxy identity is created. In the context of Zurich's IdM system, this proxy is considered a valid citizen until the earliest expiration date on any of the documents. The proxy is represented in the form of a username/password combination that simplifies online interactions with services within same context. Although the proxy inherits the properties from the original identity, it is fundamentally different since it is a copy that is only loosely coupled to the original identity context. This means that an update in the original context may not be noticed in the context of the clone.

The following list describes the vulnerabilities in regards to the process of registering a new identity with the IdM system of the VA:

- T1 – IdM System Insider Attack** The paper assumes that the IdM system can maintain the set of eligible voters securely. However, these IdM systems are centralized and vulnerable to insider attacks. An employee with the necessary system authorizations can inject new entries into the system and bypass the IdP's authentication process multiple times. As long as the IdM system is not monitored properly, such an attack could go through unnoticed.
- T2 – Forged Identification Documents** The employee at the administration office validates the documents on their security properties. Either specialized hardware is used to scan these documents and algorithms analyze the security characteristics or a trained person can identify the security features by eye. Either way, the output of the analysis of these features is probabilistic. There is no 100% guarantee that the documents are issued by the authority that they claim to be. The evaluation of physical governmental certificates is slow, expensive, error-prone and the outcome is not binary. There is a chance that forged identification documents pass the verification of a scanner or the employee evaluating the document.
- T3 – Loosely Coupled Identity Contexts** If one of the documents becomes invalid before the expiration date and the administration office is not notified immediately, the proxy identity remains a valid citizen inside the context of the IdM system and may still use the services as a valid citizen. This is the consequence of the chain of loosely coupled dependencies and derived identities. Ideally, the user has to prove the validity of these documents with every meaningful interaction. However, with the current infrastructure of credential issuance, this is too cumbersome and as a result, many systems have adapted to the proxy identity pattern. Creating new identity contexts that are loosely coupled, inherits the risk of data inconsistency where the same credential is valid and expired in two different contexts.

5.2.2 List Communication

The set of entitled voters is generated at the time of the election creation and passed to the IdP in the form of a list of unique identifiers. The following list describes the vulnerabilities in regards to the process of communicating the list of eligible voters to the IdP:

- T4 – Self-Explanatory Identifier** Email addresses contain linkable data such as a name, birth date, company name etc. This may enable the IdP to link a voter to the real identity and the IdP can learn which voters have registered.
- T5 – Reused Identifier** Anonymized email addresses can be linked to a voter’s real identity if the same email address is also used for other services. It may be listed on a company’s website, leaked from an online store alongside the name and address of a user, published as part of a social profile, etc. Thus, the IdP may be able to link the voter’s email address to the real identity and learns which voters have registered. The content of the vote is unknown to the IdP. The risk of linkage can be minimized by creating a new email that only interacts with the Provotum system. However, it is out of Provotum’s control to make sure that the email is not used for other services too.
- T6 – Email Account Hijacking** Any password-based authentication is vulnerable to dictionary attacks [20] and credential stuffing [15]. The enforced security rules on the voters’ email accounts is outside of Provotum’s control. However, the Provotum’s security is only as strong as its weakest link. Using emails for the authentication process creates a security dependency on the email provider and the email client. If the email provider does not enforce strong passwords, the voter’s email account is at risk and so the ballot. Such an attack would be noticed when the user attempts to authenticate with the IdP.
- T7 – Static List of Eligible Voters** The paper does not describe how this list is exchanged. Adding or removing a voter from the set of eligible voters after the election has been created is not possible.
- T8 – Limited Flexibility** Requiring the VA to maintain an IdM system, restricts the flexibility and complexity of authentication configurations. As stated in *T3*, incorporating new credentials into the IdM system is expensive, slow and susceptible to errors. The following example illustrates this issue. The VA wants to make an poll where only people can participate that are vaccinated against Covid19. There are two options to verify if a voter is vaccinated. *(i)* If a digital vaccination card is available from a TTP, the VA can verify the validity from this service. As previously explained in Section 2.2, this raises privacy concerns since a third party is involved in the process. *(ii)* If the vaccination card is also available as a physical certificate, the VA must verify these documents in a error-prone verification process as further explained in *T2*. Both solutions are not desirable and thus, the flexibility of the election configuration is restrictive.

5.2.3 Authentication

The IdP acts as a TTP. Its role is to blindly sign the public key of entitled voters that can later be used to cast a vote. The following list describes the vulnerabilities in regards to the authentication process:

- T9 – DoS Attack on IdP** If the service of the IdP is not available, voters can no longer request blind signatures and cannot cast a valid ballot. It represents a SPOF of the system. If the service of the IdP is not available, voters can no longer request blind signatures and cannot cast valid ballots. Attacking the web servers of the IdP can make the system halt.
- T10 – IdP Voting** The IdP must not prove that the circulating signatures were only issued upon successful authentication. No mechanism prevents the IdP from creating new addresses, blindly signing them and casting valid ballots. Note that this would only be noticed if the IdP casts more ballots than there are eligible voters defined by the VA.
- T11 – Target Advertising by SSO Provider** The prototype suggests using SSO as an alternative to an email address and password login. The incentives of many SSO providers is to aggregate lots of metadata about their users and monetize these detailed user profiles. Integrating a commercial SSO provider leaks information. The provider learns if the user has already registered for a certain election alongside metadata such as the location, time, web sites visited prior and after the registration, etc. Depending on the size and reach of the provider, this information can be connected and used for targeted and manipulated advertisement. [5, 27]
- T12 – Email Provider Attack** To prove ownership of the registered email address, the IdP sends OTPs to the voter to prove ownership of the identifier. Although this is not yet implemented, this is the assumed process. Popular email clients such as Gmail and Outlook do not enable end-to-end (E2E) encryption allowing them to read the contents of your emails. If an OTP is leaked, the user would not be able to authenticate with the IdP. This would be noticed. The same problem occurs with ISPs and OTPs sent in SMS. This would be less problematic if the VA communicates two authentication factors with the IdP such as a password or phone number. However, this increases the chances of threat T_4 and T_5 where the IdP can link a user to his real identity.
- T13 – Email Provider Data Leak** Due to the missing E2E encryption the email provider learns which voters have registered for an election.
- T14 – VA and IdP Collusion** In most use cases, the VA has sensitive information about its users and can identify them. Thus, Provotum 3.0 outsources the eligibility verification to an independent entity (IdP) that ideally cannot link the identifier to the real identity. However, the IdP and an insider of the VA with reading access to the IdM system can collide to find out which users have cast a ballot and which did not.

5.2.4 Authorization

After being successfully authenticated the voter can register on the PBB. This direct interaction with the DL has to be made before any vote can be cast. To register, the voter must submit a public key that is blindly signed by the IdP. Whenever a vote is cast, the network validates if the transaction originates from a registered public key. The following list describes the vulnerabilities in regards to the vote authorization process:

T15 – Binary Registration Provotum’s PBB distinguishes between registered and not registered addresses. However, a registered voter may not be eligible for every election that is made on the Provotum PBB. As a result, it is not possible to host two elections in the same Provotum network for two different sets of eligible voters.

ID	Title	Threat
T1	IdM System Insider Attack	An insider of the VA with the necessary credentials can inject forged identities into the IdM system.
T2	Forged Identification Documents	The process of evaluating physical identification documents is error-prone since it relies on probabilistic evaluation methods.
T3	Loosely Coupled Identity Contexts	Loosely coupled identity contexts and proxy identities result in data inconsistency.
T4	Self-Explanatory Identifier	Email addresses often contain personal information. This may allow the IdP to link the voter’s email address to the real identity.
T5	Reused Identifier	Email addresses are reused among different services. This may allow the IdP to link the voter’s email address to the real identity.
T6	Email Account Hijacking	Enforcing strong passwords on email accounts is out of Provotum’s control.
T7	Static List of Eligible Voters	A voter that loses the required credentials during election runtime, can still cast a vote since the list is defined during bootstrapping
T8	Limited Flexibility	Configuring complex authentication configurations is expensive, slow and error-prone.
T9	DoS Attack on IdP	A DoS attack on the IdP can make the entire system halt.
T10	IdP Voting	The IdP can generate and sign public keys and use them to submit a valid vote.
T11	Target Advertising by SSO Provider	Election manipulation by targeted advertisement campaigns enabled by the SSO provider.
T12	Email Provider Attack	Email is the only authentication method and emails are not E2E encrypted allowing an insider to use the OTPs to register.

T13	Email Provider Data Leak	Email provider learns which users have requested a registration OTP.
T14	VA and IdP Collusion	Insider of VA and IdP can collide to find out which users have not registered.
T15	Binary Registration	A voter is either registered for all or no election.

Table 5.1: Threat model of Provotum 3.0 in regards to IdM

Chapter 6

Design of an SSI-based IdM System for Provotum

This chapter describes Provotum 3.0 with the extension of a decentralized IdM system. The proposed architecture is illustrated in Figure 6.1. The issuance, storage and verification of credentials align with the concepts of SSI described in Section 2.3. Thus, this chapter makes use of the terminology and abbreviations explained in Chapter 4.

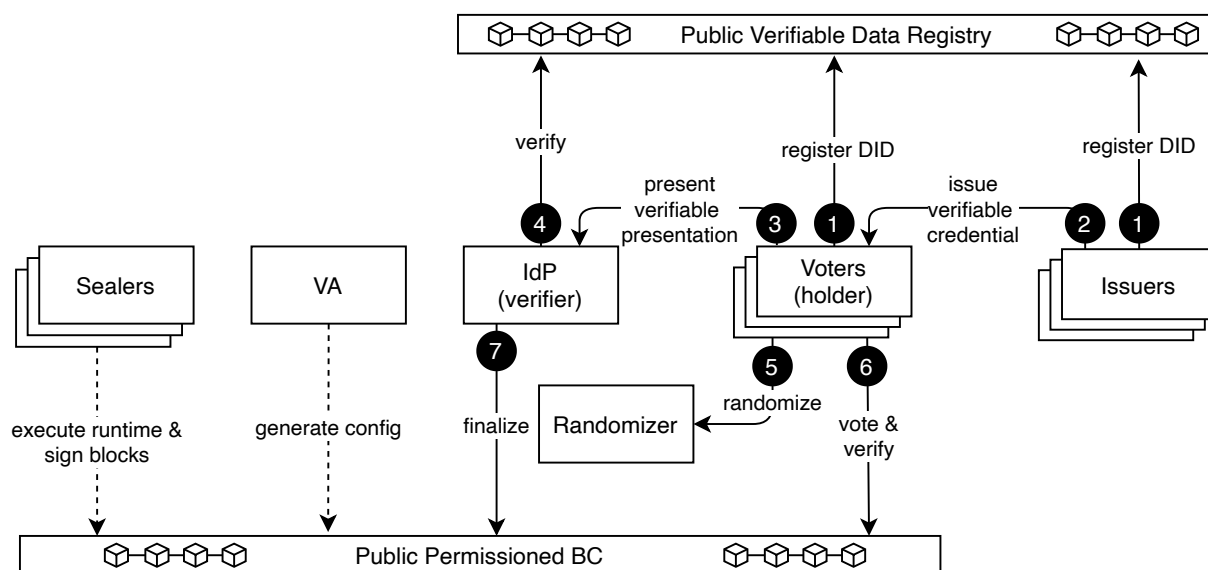


Figure 6.1: Provotum 3.0 extended with a decentralized IdM system

6.1 Stakeholders

All processes relevant to the correctness and the configuration of the Provotum voting system are mostly unchanged. The focus of this thesis is on the authentication and authorization process. The proposed design incorporates the concepts of an SSI-based

IdM system to achieve a secure and privacy-preserving eligibility check. The new design consists of the same entities as in Provotum 3.0 alongside Issuing Authorities (IAs) and a public data registry which is a public permissionless BC that manages the lifecycle of decentralized identifiers (DIDs).

- The Substrate PoA DL acts as the system’s **Public Bulletin Board PBB**. The DL implements the Provotum protocols directly into its runtime, exposing an API with which it is possible to submit transactions and read state tied to the voting process. The PBB is also used as the communication channel for all entities.
- As with the previous iteration of Provotum, the **Voting Authority (VA)** is responsible for bootstrapping and orchestrating the Provotum system. It coordinates the distributed key generation (DKG) with the sealers, creates the election questions, and closes the election by signalling the sealers to start tallying. In the previous implementation, a list of eligible email addresses was passed to the IdP. This functionality is replaced with an SSI-credential-based authentication process. Newly, the VA must define authentication configuration specifying the claims that voters must present to the IdP. As an example, this configuration may contain a claim such as “the voter must have a valid Swiss passport and the date of birth must be more than 18 years ago”.

In the context of SSI, the classical role of the verifier is split among the IdP and the VA which is unique to Provotum. While the VA defines the claims and trusted IAs, the IdP executes and validates the received claims.

- An **Issuing Authority (IA)** is a TTPs that creates signed claims about a subject in the form of VCs. These credentials must be disclosed to the IdP during the authentication process. In SSI terminology, the IA has the role of this issuer. The VA, IdP and the IA form a trust-network. The VA has to trust the IA that credentials are issued truthfully and the IdP that the claims are validated veridically. The task of the IA could be carried out by the administration office that also issues physical passports and other identification cards. Furthermore, it is also possible to invalidate credentials by maintaining a list of revoked DIDs [1]. Revocation lists are also maintained publicly on the data registry enabling the IdP to check the validity of a credential at runtime without contacting the IA.
- **Voters** are eligible people that participate in the election. They interact directly with the PBB when casting their vote. To do so, they are responsible for the private key management. With the integration of SSI concepts in the process, they are in charge of managing DIDs and VCs. In the SSI context, the voter has the role of the holder.
- The **Identity Provider (IdP)** receives instructions from VA in the form of the authentication configuration on the task of access control to the election. Voters are asked to disclose credentials about themselves in order to register on the PBB. The IdP validates those credentials with the help of the data registry. Thus, in terms of SSI, the IdP takes the role of the verifier.

Theoretically, it would be possible to validate VCs directly in the Provotum runtime. However, the proposed design does not omit the IdP for privacy reasons. If VCs

were evaluated directly on the PBB, the IAs of the VCs learn which users have cast a vote. One can argue that with the proposed architecture, the IdP gains this knowledge instead of the IA. However, the IA usually has detailed information about the user. For example, the passport issuing authority controls very sensitive information. During the Provotum authentication process with the IdP, voters may only have to disclose the nationality and date of birth on the passport to prove the nationality and adulthood. This limited information does not allow the IdP to link a voter to his real identity.

- **Sealers** secure the network by running DL validator nodes. They also participate in the DKG during the bootstrapping phase which is used for the vote encryption process.
- The **Randomizer** blinds the voter's ballot for achieving receipt-freeness. A blinding factor is used to make it impossible for a voter to reproduce the same ballot, making vote-buying much harder.

6.2 Processes

Mainly, there are three new processes introduced: *(i)* credential issuance by the IA to the voter, *(ii)* the authentication configuration by the VA and *(iii)* the disclosure and verification process between the voter and the IdP. Interactions such as bootstrapping the system or casting a vote have not changed and are not covered. These workflows continue to work as in Provotum 3.0 [3].

6.2.1 Identity Issuance

IAs act as TTPs besides the IdP and the randomizer. The difference between these entities is that IAs also exist outside of Provotum's context. Therefore, the registration and maintenance of DIDs must not be recorded in the Provotum runtime. Instead, a public permissionless BC is used to maintain DIDs since anybody has to be able to read the registered DIDs. The network that maintains DIDs and resembles the foundation of a digital identity should not be governed by a set of selected trustees and the network should be as censorship-resistant as possible. Due to the available tools and libraries, Ethereum as a public permissionless BC is chosen but can be replaced with any other public permissionless BC.

Before a VC can be created, the IA and the voter must both first create a DID. Ethereum uses the Elliptic Curve Digital Signature Algorithm (ECDSA) for deriving the public key from a private key. With the generation of a key pair, a valid DID is created accordingly¹ without the need of an on-chain transaction.

Usually, the IA requires an onboarding procedure where users must identify themselves independent of physical or digital credentials. If the user passes the identity check, the

¹Multiple DIDs are created since this key pair can also be used on Ethereum's testnets.

user's newly created DID is communicated to the IA. The IA creates one or multiple atomic VCs containing all the verified attributes, signs them and passes them on to the voter. The voter stores the VCs locally in a secure wallet.

Ideally, an IA for commonly used identification documents such as passports, divers licence, etc., uses the same DID for all VC of the same type to its customers. Since the issuing DID on a VC substitutes all the security features of a physical passport, publicly know DIDs of such an authorities enable the government to publicly maintain a curated list of trusted issuing authorities². On the other end, the voter is encouraged to use a new DID for each VC that is issued. This guarantees a higher degree of privacy since a verifier is not able to link multiple VCs to the same identity when they are disclosed in independent authentication processes.

To create a common context for VCs among the issuer, verifier and the holder, Schema.org is used to define the vocabulary used in the credential. Using Schema.org is not required but is a recommended practice since it makes the credentials interoperable with other systems. New schemas can be defined if needed.

6.2.2 Authentication Configuration

Defining the set of eligible voters is fundamentally different to Provotum 3.0. The VA is not required to maintain the list of eligible voters. Instead, an authentication configuration is created and published which consists of a list of tuples. Each tuple has two properties:

- A list of required credentials must be disclosed in the authentication process with the IdP. Each claim itself contains a list of accepted IAs in the form of DIDs. Furthermore, a reason can be stated to communicate to the voter why this claim must be disclosed.
- The interpretation and evaluation of the disclosed value is not part of the SSI framework. The standard does not incorporate rules or instructions on how the value of a claim is to be analyzed. Thus, the configuration also instructs the IdP on how each value of a claim must be evaluated.

As an example, the VA creates an election that only allows adult voters. The requested credential is the date of birth. The list of accepted IAs could consist of all the passport issuing authorities in Switzerland. The included reason could state that only adult people can participate in this particular election. The constraint declaration would be defined such that the disclosed value must be more than 18 years ago.

Note that the two-folded configuration also applies to the verification process of physical credentials. Even though the government defines how to identify the validity of the physical credentials, applying access restriction rules is not within the same scope. It is up to the verifier to analyze and compare the disclosed value. One verifier may do the calculation of an age verification solely in the head, while others use a calculator.

²Ideally, this list is maintained as a credential registry as discussed in Section 8.3.4

Complex authentication restrictions can be enforced by the VA by including multiple claims into the authentication configuration. If the set of eligible voters is too complex to define with existing VCs, the VA can also become an IA and issue custom VC to the voters themselves. However, this requires the VA to maintain an IdM system.

6.2.3 Identity Verification

Although incorporating more trusted entities in the form of IAs seems to make the system less decentralized at the first glance, this is not the case since in both architectures there has to be one trusted source of truthful identification issuance. Either this role is assigned to the VA or a trusted IA. Using a governmental IA benefits from the fact that the original identity context is used. Thus, the risk of data inconsistency between the original identity context and the internal IdM system is eliminated and it makes the system less error-prone, more efficient and privacy-preserving. One might argue that the introduction of IAs resurrects the Access Provider (AP) from Provotum 2.0 [14]. This entity acted as a gatekeeper and increased privacy by decoupling the authentication process from the election process. The key difference between the IA and the AP is that IAs also exist outside of Provotum. SSI-based authentication enables Provotum and relying parties in general to directly connect to IA's IdM system without the need for cloning the identity into a new context. The original credentials signed by the trusted authority are used as a means of authentication.

User accounts do not exist in the proposed architecture. One can imagine the authentication process as if a voter has to disclose the real identification document when casting a ballot. Since a digital version of the passport can easily be verified, the overhead of disclosing and verifying the document is minimal.

The IdP reads the authentication configuration from the PBB and creates a Selective Disclosure Request (SDR). The SDR contains the required claims with its accepted IAs and a reason for each request. The SDR is signed by the IdP and sent to the voter. Since the authentication configuration is stored publicly, the voter can verify that the IdP does not ask for more credentials than defined by the VA. Also, the voter can verify that the SDR was signed by the IdP as the DID of the IdP is also recorded on the PBB.

Upon receiving the SDR, the voter is in control of disclosing previously issued VCs and must give permission to share them with the IdP. By doing so, a Verifiable Presentation (VP) is constructed containing the signed credentials. By proving ownership across all DIDs that are used in the VCs, the VP cannot maliciously reuse these VCs since the IdP has no access to the associated private keys.

The signed VP undergoes a three-step verification process by the IdP. *(i)* The VP is validated against the SDR to make sure that the VP contains all the necessary VCs and are signed by the required IAs. The IdP does not have to contact the IAs to verify if the credentials are valid. Instead, the DIDs of the IAs are resolved directly from the verifiable data registry. This ensures that the IA is not involved in the authentication process. *(ii)* The IdP checks its database if any of the VC was previously used for the same election. *(iii)* The constraint declaration for each disclosed VC is evaluated. If all three checks

are valid, the IdP stores the VCs locally and returns the blindly signed public key to the voter.

The proposed IdM system requires a radical change in how credentials are issued and stored. By not incorporating the IA in the authentication process, they do not learn when their issued VCs are used and the voter's privacy is protected. Since atomic VC are used where only one claim is contained in a VC, the IdP only learns the minimal necessary information about the voters.

Chapter 7

Implementation

Alongside this thesis, a prototype was developed implementing the proposed architecture. The source code of all the packages is published in the Provotum Github organization¹. Figure 7.1 illustrates the relevant packages for the credential issuance and SSI authentication process.

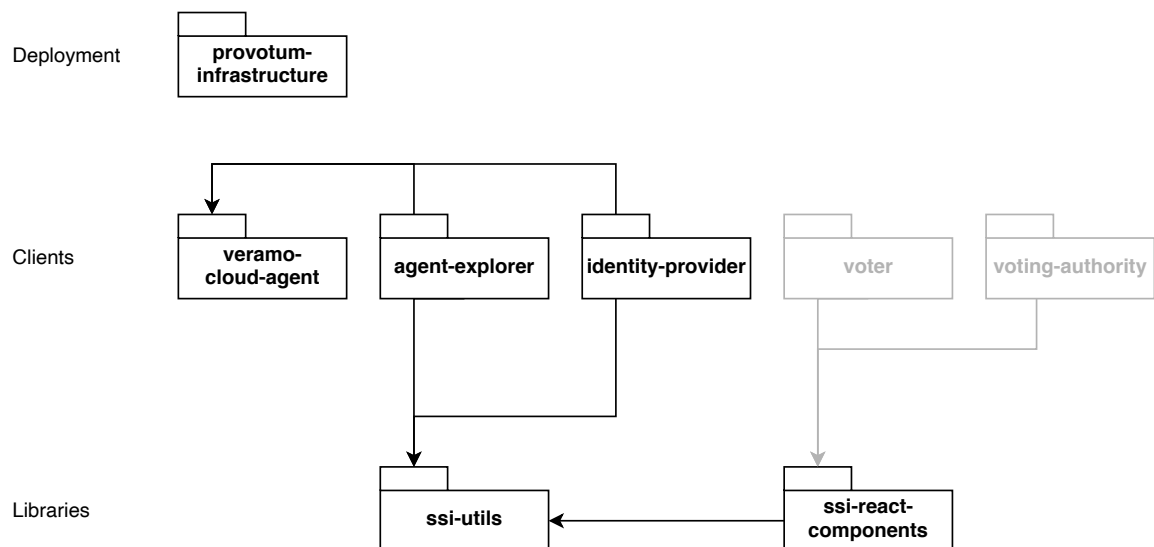


Figure 7.1: Overview of Provotum's SSI repositories

Veramo (Section 3.3.1) currently only supports Ethereum-based and web-based DIDs. However, Veramo can be replaced with any other SSI system as long as it supports the W3C DID [23] and VC [28] specification. The reason for choosing Veramo for this thesis is its modular structure. There is no lock-in effect as with other platforms where only one type of Ledger is supported. It is open-source and new modules can be added to support other types of DIDs or signature schemes in the future.

¹<https://github.com/provotum>

veramo-cloud-agent

The Veramo framework (Section 3.3.1) is used for managing all SSI relevant processes. Veramo provides interfaces for several clients. However, at the time of writing this thesis, the Veramo cloud agent is the most complete and best-documented implementation due to its OpenAPI² specification. Thus, the implementation uses the `veramo-cloud-agent` repository which was generated using the official Veramo template³⁴. Every entity that either issues, holds or verifies VCs runs an agent in the cloud. The service exposes all its functionality in the form of a RESTful API. In the future, a dedicated mobile app for managing DIDs and VCs locally could offer a simpler setup process for the voters.

agent-explorer

This implementation is a ReactJS web application enabling a GUI-based interaction with the `veramo-cloud-agent`. It is used by the IAs to create DIDs and VCs. Voters also use it to create DIDs, import VCs and selectively disclose VCs in the form of VPs. It is forked from a core project by Veramo⁵. Many of the required features are missing at the time of the fork, such as importing VCs, issuing VCs from predefined context templates, scanning an SDR and responding with a VP. These features are added such that an E2E process from issuing credentials to the voter to casting a vote can be demonstrated.

ssi-utils

This TypeScript library shares common functionalities between the `agent-explorer`, identity provider and the voter and voting authority web application. It includes interfaces and predefined schemas for VCs. It is available as a public NPM package under the name `provotum-ssi-utils`.

ssi-react-components (voter, voting-authority)

Parallel to this thesis, the Provotum frontend applications are rewritten. For this reason, a ReactJS component library is created that makes the integration of all the SSI functionality into new projects easier. Storybook⁶ is integrated to showcase and document the usage of the components. The library is available as a public NPM package under the name `provotum-ssi-components`. Mainly the following two components are relevant for integrating the SSI functionality:

²<https://swagger.io/specification/>

³<https://github.com/uport-project/veramo-agent-deploy>

⁴https://veramo.io/docs/deployment_tutorials/deployment_aws

⁵<https://github.com/uport-project/veramo-agent-deploy>

⁶<https://storybook.js.org>

- The `AuthConfigurator` enables the voting-authority to generate an authentication configuration for an election as described in Section 6.2.2. This component is intended to be used in the VA frontend.
- The `IdpButton` targeted at the voter frontend that manages the authentication process with the IdP is provided. It establishes a connection the the IdP and initiates the authentication process.

To showcase the integration of the `IdpButton`, the `voter` frontend application was updated such that an E2E process can be demonstrated including a credential-based authentication process.

provotum-infrastructure

To build and deploy all the services with *docker-compose*, `provotum-infrastructure` was updated with the new services. All the docker images are hosted on Docker Hub⁷.

7.1 DID Creation

As described in Section 4.1.1, DIDs are self-certifying identifiers whose ownership can be proven by their controller. The Ethereum DID method⁸ using the DID registry⁹ has the benefit that the creation of a DID does not require an on-chain transaction. This is possible since the DID itself contains a public key as shown in Table 4.3. If the DID resolver is not able to find any transaction associated with a particular DID on the ledger, it implies that the controller is the same public key as illustrated in the Listing 2. Thus, creating a new DID on Ethereum is fast and does not cost any network fees.

7.2 Credential Issuance

Issuing a VC requires a DID from the IA and the voter. The voter creates a new DID for every new credential to preserve privacy, while the issuer's DID is often required be commonly known to simplify the verification process. Section 8.3.4 discusses how publicly know DIDs can be maintained. The voter's DIDs is released to the IA. Meaningful credentials such as passports are often issued in a personal verification procedure. In this scenario, the voter's DID can be transferred offline by scanning a QR code on the spot.

Incorporating commonly used schemas such as those from Schema.org, makes VCs interoperable and more likely to be accepted by other parties. However, the underlying

⁷<https://hub.docker.com>

⁸<https://github.com/uport-project/ethr-did>

⁹<https://github.com/uport-project/ethr-did-registry>

¹¹DIDs are shortened for displaying reasons

```

1  {
2    "@context": [
3      "https://www.w3.org/ns/did/v1",
4      "https://identity.foundation/EcdsaSecp256k1RecoverySignature2020/lds-ecdsa
5        -secp256k1-recovery2020-0.0.jsonld"
6    ],
7    "id": "did:ethr:0xb9c...e8a",
8    "verificationMethod": [
9      {
10       "id": "did:ethr:0xb9c...e8a#controller",
11       "type": "EcdsaSecp256k1RecoveryMethod2020",
12       "controller": "did:ethr:0xb9c...e8a",
13       "blockchainAccountId": "0xb9c...e8a@eip155:1"
14     }
15   ],
16   "authentication": [
17     "did:ethr:0xb9c...e8a#controller"
18   ],
19   "assertionMethod": [
20     "did:ethr:0xb9c...e8a#controller"
21   ]
22 }

```

Listing 2: Default DID document from Ethereum DID-resolver when no transaction has been recorded¹¹

data structure of the JSON-LD encoding is a graph. Therefore, the schemas may contain cycles. For example, person P may be part of an organization O and organization O has a set of employees where person P is part of. Thus, programmatically extracting the fields of a given schema type is expensive as it may result in an infinite loop [8]. As a result, JSON-LD schemas as those from Schema.org are helpful for creating a common vocabulary among two entities but not for instantiating new objects for a given schema type. Creating a VC based on a given schema type, is achieved with the help of the NPM package `ssi-utils` which contains a subset of the schema types from Schema.org without cycles.

VCS are issued in the form of a JWT. Veramo uses a single message interface that processes data from other agents. Any incoming message is processed by a chain of message handlers. If any of the message handlers can successfully parse the message, it is further processed by the agent. When the IA sends the VC in the form of a JWT, the `did-jwt` message handler¹² parses and validates the JWT. After that, the VC can be imported into the agent.

7.3 Authentication Configuration

The VA creates a configuration that defines the set of eligible voters for a particular election. This does not require any interaction with a Veramo agent. Every election is

¹²<https://veramo.io/docs/api/did-jwt.jwtmessagehandler>

The screenshot shows a web interface for issuing credentials. On the left is a sidebar with navigation options: Dashboard, Activity, Requests, Identifiers, Credentials (selected), and Import. The main content area is titled 'Credentials' and contains a form for 'Issue Credential'. The form has three main sections:

- Subject:** A text input field containing the DID: `did:ethr:rinkeby:0x826fb3201a7644a1416a2742412b21321eb8f68daf4bb8f28a47e0bdb4c288c6`.
- Issuer:** A dropdown menu showing the DID: `did:ethr:rinkeby:0x03011e433fe203bcd1e2eab680c2cb79d95e28b7b3f5a249b3a20903a13af8cf2e`.
- Claim:** A dropdown menu with the value `https://schema.org / Person / PostalAddress / postalCode` and a text input field containing the value `8006`.

 Below the form is a blue button labeled 'Sign and Issue' and a link labeled '> Preview'.

Figure 7.2: UI for issuing an atomic VC that uses Schema.org vocabulary to define a person’s postal code. As a result, a JWT is produced that can be displayed as a QR Code and imported by the holder to its own agent.

configured with a separate authentication configuration. The configuration consists of a list of two-folded tuples. (i) The claim defines which credentials must be disclosed in the authentication process with the IdP. It defines the required context, accepted issuers and reason why the claim must be disclosed. (ii) The constraint instructs the IdP with additional restrictions on how to validate the disclosed values. This is achieved with a JavaScript expression (`js`) stored as a raw string with a placeholder (`${VALUE}`) that is replaced with the disclosed value. The JavaScript `eval()` function expects a string as input and returns the result of the evaluated expression. Using raw JavaScript strings makes it possible to define any constraint that can be captured with valid JavaScript syntax. The description (`desc`) of the constraint has no functionality other than providing a human-readable string of the constraint.

Listing 3 illustrates an authentication configuration where the voter must disclose a claim that proves his postal address. This is defined by the `claim` object. Such a claim could be issued by the cantonal residents registration administration. The `constraint` adds further restrictions by only allowing certain postal codes to be accepted. To model an election without access restrictions, the authentication configuration can be omitted by submitting an empty array. This is useful for development and demo purposes.

The `ssi-components-library` currently supports the creation of the following constraints.

- Dates (ISO 8601 date format¹⁵)
 - Exact date
 - Before or after a given date

¹⁴DID is shortened for displaying reasons

¹⁵<https://www.w3.org/TR/NOTE-datetime>

```

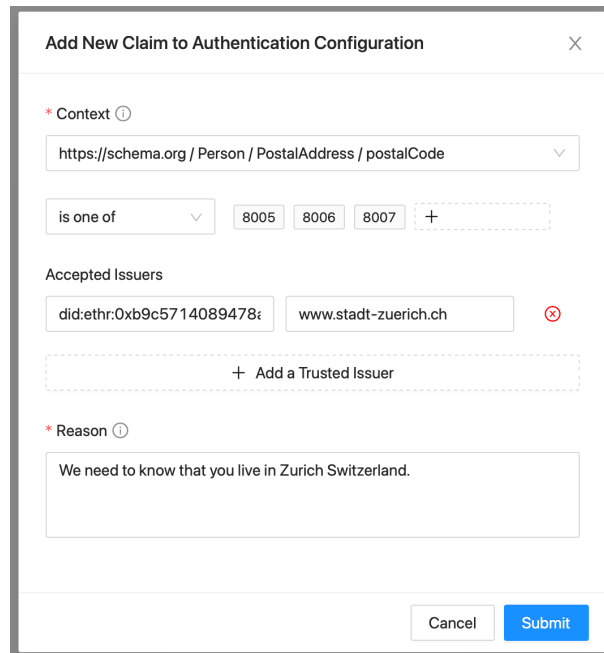
1  [
2  {
3    "claim": {
4      "credentialContext": "https://schema.org",
5      "credentialType": "PostalAddress",
6      "claimType": "postalCode",
7      "essential": true,
8      "issuers": [
9        {
10       "did": "did:ethr:0x030...f2e",
11       "url": "https://www.stadt-zuerich.ch"
12     }
13   ],
14   "reason": "We need to know that you live in Zurich."
15 },
16 "constraint": {
17   "js": "['8005','8006','8007'].includes('${VALUE}')" ,
18   "desc": "The claim must exactly match any of the elements from '['8005',
19     '8006','8007']'"
20 }
21 }
22 ]

```

Listing 3: Example of an authentication configuration that requires a voter to disclose a VC containing the postal code and the value is either 8005, 8006 or 8007.¹⁴

- Between two dates
- Any value
- Text
 - Exact string
 - Contains a specific substring
 - Is one of many possible strings
 - Any value
- Boolean
 - True or false
 - Any value
- Number
 - Exact number
 - Is greater (or equal) than
 - Is smaller (or equal) than
 - Any value

The authentication configuration should ideally be communicated through the PBB as this does not require the VA to securely manage the configuration and also enables voters to make sure that the IdP does not ask for more credentials than necessary. Due to technical difficulties with Substrate, the configuration is currently communicated to the IdP through a separate RESTful API endpoint. The entire configuration object can be stored as a byte array since the ProvoTum runtime must not evaluate it and the PPB's sole purpose is for storage and communication.



The screenshot shows a web form titled "Add New Claim to Authentication Configuration". The form contains the following fields and controls:

- Context:** A dropdown menu with the value "https://schema.org / Person / PostalAddress / postalCode".
- is one of:** A dropdown menu with the value "is one of", followed by three input boxes containing "8005", "8006", and "8007", and a "+" button to add more values.
- Accepted Issuers:** A section with two input boxes: "did:ethr:0xb9c5714089478:" and "www.stadt-zuerich.ch". There is a red "X" icon to the right of the second box. Below this is a dashed box with a "+" button and the text "Add a Trusted Issuer".
- Reason:** A text area containing the text "We need to know that you live in Zurich Switzerland."
- Buttons:** "Cancel" and "Submit" buttons at the bottom right.

Figure 7.3: UI for creating an authentication configuration for an election. This screenshot shows how an authentication configuration tuple is added. With this credential and constraint configured, the voter must disclose a VC issued by the city administration office of Zurich and the value of this claim must be one of the defined postal codes. The result of this form corresponds to the data shown in Figure 7.3.

7.4 Identity Verification

The authentication and identity verification process is illustrated in Figure 7.4. In this process, the IdP only signs the public key of a voter if the three-step identification procedure is passed. To do so, the IdP connects to its Veramo cloud agent. With the authentication configuration of the VA, an SDR in the form of a JWT is generated and signed by the agent. The JWT is sent to the voter and is displayed as a QR code in the voter frontend application. The component library (`ssi-react-components`) provides a module for integrating this logic into any ReactJS application. It has to be configured with the URL of the IdP and the election id. The component also expects the public key that is to be signed. The blindly signed public key is accessible outside of the component

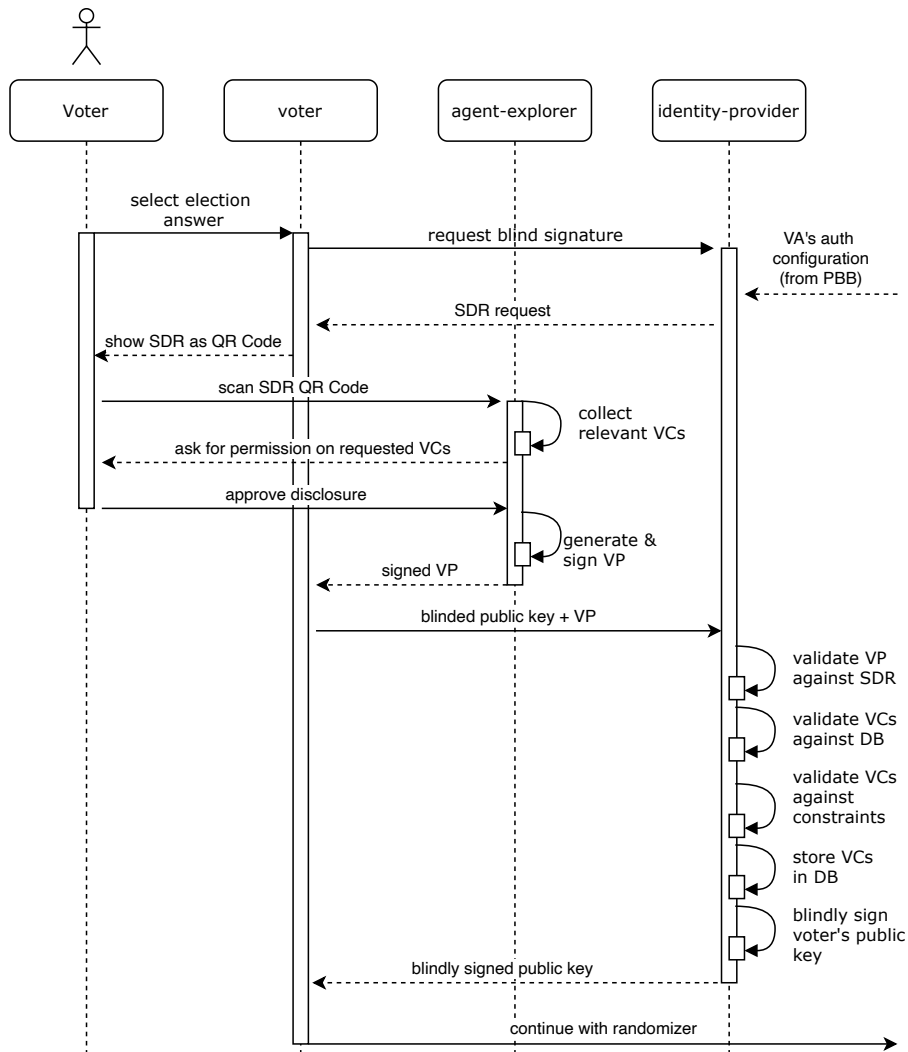


Figure 7.4: Sequence diagram for IdP authentication process

with a custom hook¹⁶. Storybook¹⁷ is integrated into the component library. It documents the usage and demonstrates its functionality.

The agent-explorer supports the functionality to import an SDR by scanning a QR code. Similar to the import of VCs, the JWT is processed by Veramo's chain of message handlers. The did-jwt message handler¹⁸ parses and validates the JWT. Each requested VC is listed and the voter must give permission to each claim individually as shown in Figure 7.5. The voter responds with a VP in the form of a JWT. It is recommended that a DID is used for the signature of the VP that is not used in conjunction with any non-disclosed VC. This guarantees more privacy to the voter. The IdP would be able to connect previous authentication processes if the same DID was used across multiple processes. Note that this is still possible if at least one of the VCs is the same in two authentication processes. If no such DID is available, a new DID can be created. The agent-explorer is implemented

¹⁶<https://reactjs.org/docs/hooks-intro.html>

¹⁷<https://storybook.js.org/>

¹⁸<https://veramo.io/docs/api/did-jwt.jwtmessagehandler>

such that the DID of the first VC also signs the VP to align with this recommendation.

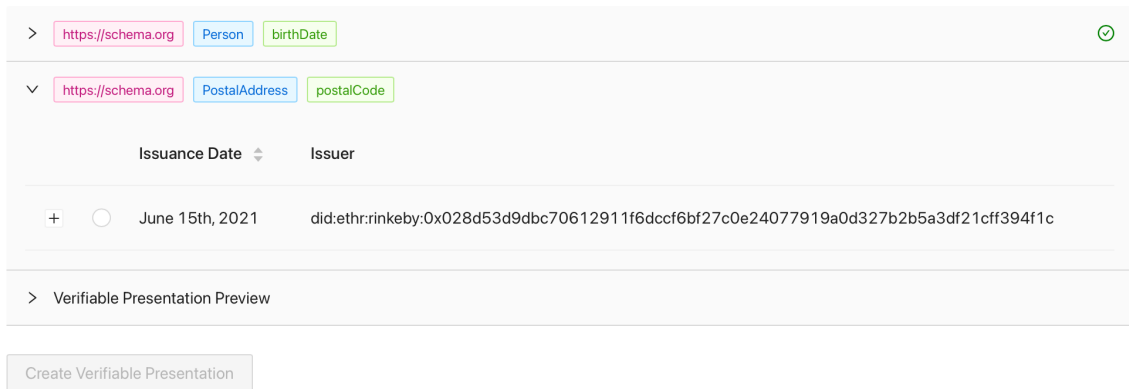


Figure 7.5: UI for selectively disclosing VCs. After scanning the SDR, the voter is presented with a list of required credentials that must be disclosed. Each credential must be permitted to be disclosed.

The resulting VP in the form of a JWT is a new argument that must be sent to the IdP when requesting the blind signature. Currently, the JWT has to be copied from the agent-explorer into the voter application. The user experience could be improved using deep links and a bridge server similar to what WalletConnect¹⁹ is providing as a communication tool between decentralized applications and wallets.

The VP and the voter's blinded public key are sent to the IdP. Veramo provides an interface for checking a VP against an SDR. It checks whether all the VCs are disclosed and issued by the correct IA. A lookup in the database makes sure that none of the disclosed VC has been used for the same election. The constraint evaluation is performed by replacing the `#{VALUE}` placeholder and calling JavaScript's `eval()` function. If all three verification steps are passed successfully, the blindly signed signature is returned back to the voter. The rest of the voting process remains unchained compared to ProvoTum 3.0.

¹⁹<https://docs.walletconnect.org/mobile-linking>

Chapter 8

Evaluation

The main focus of this thesis is to enable an authentication process to Provotum that is secure and privacy-preserving. Table 8.1 reevaluates the identified security issues from Section 5.1 with respect to the new architecture.

ID	Title	Threat	Mitigation
T1	IdM System Insider Attack	An insider of the VA with the necessary credentials can inject forged identities into the IdM system.	The new architecture does not require the VA to maintain an IdM system.
T2	Forged Identification Documents	The process of evaluating physical identification documents is error-prone since it relies on probabilistic evaluation methods.	VCs are native to the web. As a result, the outcome of their verification is binary.
T3	Loosely Coupled Identity Contexts	Loosely coupled identity contexts and proxy identities result in data inconsistency.	The original identity context is used.
T4	Self-Explanatory Identifier	Email addresses often contain personal information. This may allow the IdP to link the voter's email address to the real identity.	DIDs combined with VCs replace email addresses. Ledger-based DIDs usually consist of a random sequence of characters. DID documents do not hold personal information.
T5	Reused Identifier	Email addresses are reused among different services. This may allow the IdP to link the voter's email address to the real identity.	DIDs combined with VCs replace email addresses. Ledger-based DIDs usually consist of a random sequence of characters. DID documents do not hold personal information.
T6	Email Account Hijacking	Enforcing strong passwords on email accounts is out of Provotum's control.	DIDs use DPKI.

T7	Static List of Eligible Voters	A voter that loses the required credentials during election runtime, can still cast a vote since the list is defined during bootstrapping	The authentication configuration is flexible enough to support static and dynamics sets. An invalidation of a credential is immediately available to the IdP due to the usage of the original context. Invalidating an already submitted vote is not possible.
T8	Limited Flexibility	Configuring complex authentication configurations is expensive, slow and error-prone.	Validation of credentials is done in the original context. This process is cheap, fast and simple to verify.
T9	DoS Attack on IdP	A DoS attack on the IdP can make the entire system halt.	No mitigation implemented. Further discussed in Section 9.1.2.
T10	IdP Voting	The IdP can generate and sign public keys and use them to submit a valid vote.	No mitigation implemented. Further discussed in Section 9.1.1
T11	Target Advertising by SSO Provider	Election manipulation by targeted advertisement campaigns enabled by the SSO provider.	No third party involved in authentication process.
T12	Email Provider Attack	Email is the only authentication method and emails are not E2E encrypted allowing an insider to use the OTPs to register.	DIDs in combination with VCs replace email addresses and thus, no email provider or TTP is involved.
T13	Email Provider Data Leak	Email provider learns which users have requested a registration OTP.	DIDs in combination with VCs replace email addresses and thus, no email provider or TTP is involved.
T14	VA and IdP Collusion	Insider of VA and IdP can collude to find out which users have not registered.	VA does not maintain an IdM system.
T15	Binary Registration	A voter is either registered for all or no election.	IdP evaluates VCs per election. (Functionality on PBB not yet implemented)

Table 8.1: Threat model evaluation after the application of the new design

8.1 Privacy

With the introduction of a decentralized IdM design, the credential-disclosure-based authentication process is facilitated only between the IdP and the voter without the need of the IA or any other TTP. This direct communication guarantees a higher level of privacy compared to other standards such as SSO or OIDC (*T13*). As a result of the improved privacy protection and the removal of TTPs from the authentication process, meta information about the voting behaviour is retained. Thus, voters are less likely to become

targets of manipulating advertisements since less data is available to advertisement platforms ($T11$).

The replacement of email addresses with DIDs as identifiers also improves the possible threat of identity linkage ($T4, T5$). DIDs are randomized strings that do not reveal personal information. Also the creation of a new DID is fast and does not cost any network fees, enabling the creation of a new identifier for every application. Manually creating and maintaining new email addresses for every application is not feasible. Alternatively, there are services that provide a workaround and forward emails from anonymized proxy addresses to a main email address such as the "*Hide My Email*" service from Apple [11]. However, these services are centralized, can track your online activity and create a dependency relation since the proxy email addresses are managed outside of the user's control. Thus, DIDs are the superior identifiers in all of those regards.

Identity linkage is still possible with the proposed design. It is the VA's responsibility to create an authentication configuration that is generic enough such that the IdP is not able to link the disclosed credentials to a real identity. The VA should always only incorporate the minimal number of claims necessary to define the eligible set of voters. Incorporating needless credentials into the authentication configuration increases the chance that the IdP is able to uniquely identify a voter and thus, learns which voters have registered. To minimize the risk of identity linkage, the implementation supports atomic credentials, where only one claim is issued per VC. This enables voters to only disclose the credentials which are requested. However, even with a minimized authentication configuration, the IdP may still be able to link voters to their identity through the disclosed credentials. For example, requesting a voter's postal code and date of birth may already uniquely identify a person from a sparsely populated area. Ideally, the voter only discloses a proof that states that he lives in one of the accepted postal codes and that his date of birth was more than 18 years ago. Thus, ZKPs can further improve privacy. The necessary infrastructure and the appropriate type of ZKP has to be further investigated.

8.1.1 IdP IA Collusion

The IdP must protocol which credentials were used for every election to prevent double voting. If the logged credentials leak, the IA can connect the credentials to additional information within the IA's IdM system and identify which voters have registered. Such a data leak is illustrated in the following paragraph.

The Provotum architecture is not limited to national elections. For example, a student association at the University of Zurich creates a Provotum election where voters must prove their eligibility by disclosing a valid enrollment receipt. The University of Zurich acts as a trusted IA and issues such enrollment receipts to its students in the form of VCs. The IdP will only know that a voter is in possession of such a receipt but cannot link it to any personal information. If the authentication logs of the IdP leak, the University of Zurich would be able to reconstruct which students have registered since it stores personal data about its students and knows which VC belongs to which user. However, the university does not know for sure if the registered voters have effectively cast a vote and also, their actual vote would remain private.

This challenge is introduced with the new architecture. One can argue that it replaces Threat *T14* where the same collusion was previously possible between an insider of the VA and the IdP. However, an authentication configuration containing credentials from multiple IAs increases the risk of such an attack. If any of the IAs is corrupt and collides with the IdP the voter's identity can be linked. The higher the number of IAs incorporated in the authentication configuration, the higher is the chance of an insider attack between the IdP and one of the IAs. However, the only information that can be extracted from such an attack contains which voters have registered and which have not and not the content of the actual vote itself.

8.2 Security

The introduced decentralized IdM system proposes that the necessary credentials are issued outside of Provotum. The new Provotum system uses the original identity context of each credential, eliminating the risk of data inconsistency (*T3*). As a result, the VA must no longer maintain a database with proxy identities.

The previously used centralized IdM system enabled an adversary with the necessary credentials to inject new identities and bypass the authentication procedure multiple times. The removal of the internal IdM system mitigates the risk of insider attacks. In the new design, such an attack is still possible for an insider at the IdM system of the IA. However, the authentication configuration can be constructed such that the claims of multiple IAs is required. Therefore, an attacker would need to be able to create a forged credential in the IdM systems of all the IAs making such an attack more difficult to execute.

Email accounts are subject to a wide range of attacks (*T6*). The replacement of email addresses with DIDs as identifiers and their cryptographic properties eliminates the risk of credential stuffing and dictionary attacks. Passwords are replaced with cryptographic signatures guaranteeing a higher degree of security by design. Thus, the authentication process with DIDs and VCs is based on the same cryptographic principles which are used in the vote casting process.

In the previous architecture, OTPs were used to prove ownership of the identifier. The missing E2E encryption in email communication and the use of a single factor of authentication enables an insider of the email provider or the email client to leak OTPs (*T12*). The introduction of DIDs and the direct interaction between the holder and the verifier eliminates the threat of such attacks. Proving ownership of the identifier is facilitated with cryptographic signatures without the need of a TTP. As a result of the removal of email and SSO/OIDC providers, fewer trust assumptions have to be made in the E2E process.

Proxy identities are created after a successful onboarding process in which credentials are verified and allow users to act as identified users in a given context without having to reveal physical credentials every time. However, the design pattern of proxy identities results in loosely coupled identity contexts and data inconsistency between the original and the derived context. The new architecture which is always operating in the original

identity context benefits from guaranteed data consistency that previously was susceptible due to the use of proxy identities (*T3*).

Whenever a physical credential is verified and a new proxy identity is created, the system becomes less efficient and error-prone since the validation of physical certificates relies on a probabilistic evaluation of the security features (*T2*). The proposed architecture issues credentials that are native to the digital world and always operates in the original identity context of a credential. The validation of a VC evaluates cryptographic signatures which is cheap, fast and results in a binary outcome.

Previously, every externally issued credential had to be verified by the VA and credited to the proxy identity. This verification process is expensive and slow as it often relies on human interaction. With the digitally native credentials and the usage of the original identity context, complex authentication configuration can be constructed (*T8*). Credentials of different authorities can be aggregated into an authentication configuration without any additional verification process needed.

The issuance of credentials follows an interoperable standard proposed by the W3C [28, 23]. In combination with the use of a common data vocabulary of Schema.org, the identifiers and credentials used in the proposed architecture and the prototype are compatible with the credentials used in other SSI systems. However, the disclosure and verification process of credentials is currently defined by the Veramo framework. This process does not fully align with the W3C standard, making a disclosure procedure not compatible with other SSI applications. As this is a know difference to the Veramo project, it will be addressed in a future release¹.

Due to technical difficulties with Substrate, the envisioned design could not fully be implemented in the Provotum runtime. Thus, the registration process on the PBB remains the same as in Provotum 3.0. In the previous design, voters only register once on the PBB. A registered voter can participate in any election within the same runtime (*T15*). All services except the Provotum chain support the new design. The new architecture proposes an election-based authentication process where a voter must present the required credentials per election. The implemented authentication process with IdP is election-based as proposed in the design. To obtain a blind signature, a voter must present the required claims for that election. This enables a registration on the PBB. Since the registration on the PBB is runtime-based, a registered voter can participate in any other election. This has to be address in future work and requires the IdP to attach the blind signature to the election identifier.

8.3 Infrastructure

Some components used in the architecture are heavily influenced by other services, project or standards. This section evaluates the technologies used for those components.

¹<https://github.com/uport-project/veramo/discussions/356>

8.3.1 Veramo

Veramo was open-sourced in December 2020 as a public beta version. Although it covers many SSI functionalities, some crucial features are still in development. For example, a VP may contain multiple VC which are issued for different DIDs. During the disclosure, the verifier must make sure that the holder is in control of the DID in each VC. Currently, this controller check is in research and development² and not yet integrated into the prototype. Without this check, a holder can disclose a VC that is not under his control.

In the context of VPs, another crucial functionality is still missing. When a holder responds to an SDR with a VP, the VP does not contain information stating that it is generated for that particular SDR. This is not a problem as long as the holder proves ownership for all DIDs within the VCs. However, Veramo's interface for SDRs contains a significant field in the form of the tag which allows the IdP to incorporate an arbitrary string. This becomes relevant in an architecture where the IdP must prove to the PBB that it possesses a certain number of valid VCs that have been disclosed in the context of a certain election as explained in Section 9.1.1. The tag on the SDR can be used to link the SDR and VP to an election by setting the tag to the election identifier. With such a functionality in place, every VP is tagged with the election identifier.

8.3.2 Risk of Centralization Due to Vast Amount of DID Methods

At the time of writing this thesis, there are more than 70 DID method names registered in the DID Specification Registries [31]. Running a resolver for each DID method requires a connection to the underlying data registry. This task requires a large storage capacity, fast network connection and computing power that is fast enough to update the state for all the underlying networks. There are approaches to streamline the setup of a Universal Resolver³ [26]. However, if the number of DID methods expands further, an IdP needs to rely on third-party services to resolve DIDs. This leads to an undesired trend for centralization. A possible solution to that problem is to restrict the allowed DIDs on the issuer level and maintain a list of supported DID methods. For example, a passport issuing authority could only issue VCs to people that use a particular DID method. The list of supported DID methods could be incorporated in the DID document of the issuing authority or the governance framework (Section 8.3.4). This list is publicly available and the IdP would only need to host DID resolvers for those networks.

8.3.3 JSON-LD Encoding Schema

Working with the JSON-LD encoding schema is cumbersome for several reasons:

- Multiple serialization strategies exist, even within the same network. For example, a JSON-LD processor can retrieve a schema for a given URL by making an HTTP

²<https://github.com/uport-project/veramo/issues/375>

³<https://github.com/decentralized-identity/universal-resolver/>

GET request to the given URL. Schemas as those on Schema.org are retrieved differently and the resolver must know which strategy must be applied. The resolution process of the context of a Schema.org document requires the processor to query the JSON-LD document on a different URL. In order to discover this URL, a HTTP HEAD request to `https://schema.org` has to be made. The header of the response indicates where the JSON-LD schema definition is located. The processor can find the JSON-LD document that contains the entire Schema.org collection in this location. As shown in this example, the JSON-LD processor must maintain a list of URLs and a strategy on how they can be resolved or it must attempt multiple strategies before successfully retrieving the JSON-LD document. Thus, the process of retrieving JSON-LD documents is either inefficient or cumbersome to maintain.

- Schemas contain cycles and make canonicalization difficult and inefficient. [8]
- Different notations exist with the same meaning as illustrated in Listings 4 and 5. There exist even more notations such as the flattened or framed notation. The JSON-LD playground⁴ illustrates all different notations.

```
1 {
2   "http://schema.org/familyName": [{"@value": "Obama"}],
3   "http://schema.org/givenName": [{"@value": "Barack"}],
4   "http://schema.org/jobTitle": [{"@value": "44th President"}],
5   "http://schema.org/name": [{"@value": "Barack Obama"}]
6 }
```

Listing 4: Compacted JSON-LD notation for a person

```
1 {
2   "@context": "http://schema.org",
3   "type": "Person",
4   "name": "Barack Obama",
5   "givenName": "Barack",
6   "familyName": "Obama",
7   "jobTitle": "44th President"
8 }
```

Listing 5: Expanded JSON-LD notation for a person

Despite these reasons, JSON-LD is the standard encoding scheme in the SSI ecosystem. Interoperability is crucial for the success of SSI and thus, no other encoding scheme was considered to be used and integrated.

8.3.4 Incomplete W3C Specification

Although the DID specification [23] is currently still declared as a working draft, some crucial components are missing. As suggested in *Self Sovereign Identity* (Chapter 11.3.3)

⁴<https://json-ld.org/playground/>

[2], a public credential registry allows issuers to publicly register VCs to enable a wider application of DIDs and VCs. The entity maintaining such a list of public VCs is called the governance authority and publishes a governance framework. It defines the rules and policies on how an issuer can become part of that registry. Figure 8.1 illustrates how credential registries fit into the SSI architecture. The same VC that is issued to the holder, is also publicly registered such that it can be searched, discovered and verified. Such credentials are not meant for sensitive data like passports. However, they can be used for registering businesses alongside their licenses and certificates. In the ProvoTum context, the Swiss government can publicly maintain the list of certified passport issuing authorities in the form of a credential registry. During the creation of the election, using a credential registry instead of a manually registered list of DIDs of all the trusted IAs is less error-prone. Figure 8.2 illustrates how the governance authority makes sure that only certified issuers are part of the credential registry. With such an architecture in place, ProvoTum’s IdP could directly get the DIDs of all certified passport issuing authorities from the credential registry.

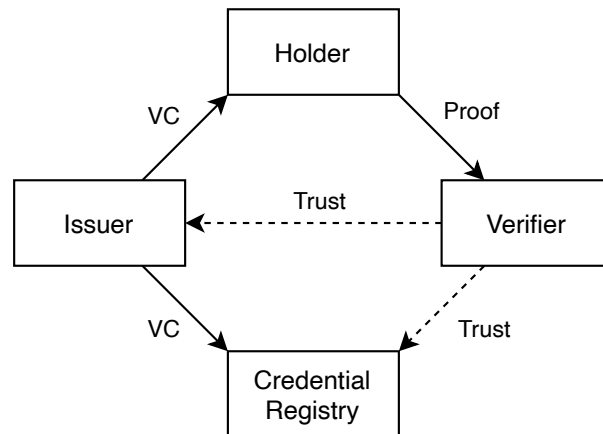


Figure 8.1: Credential registry for public VCs

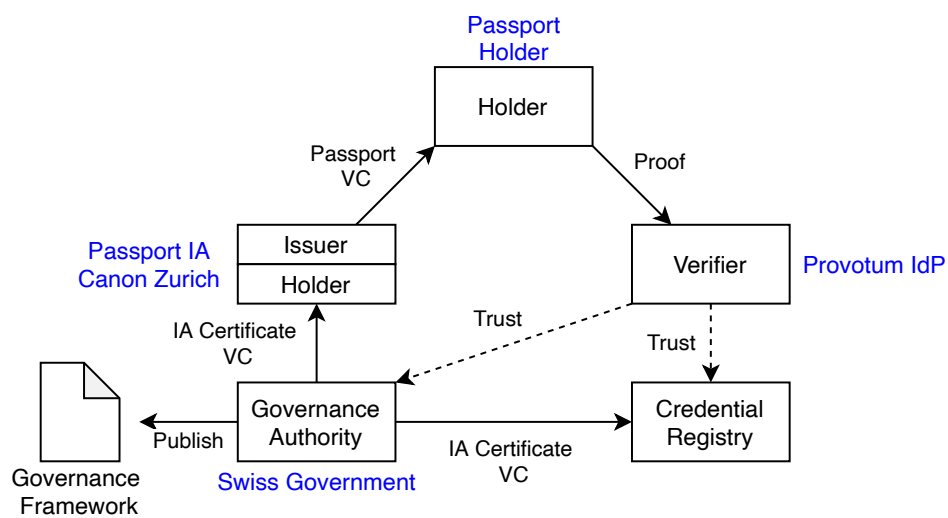


Figure 8.2: A governance framework defines the rules and policies to become a verified issuer for a given context. The example shows that the Swiss government could publicly maintain a list of verified passport IAs in the form of a credential registry. Thus, the list of eligible IAs can be discovered and verified by only knowing the DID of the Swiss government.

Chapter 9

Summary and Conclusion

This thesis sets out with the goal of analysing and improving the existing Provoctum architecture in terms of IdM. The threat analysis pinpoints the vulnerabilities, which are addressed with a decentralized IdM system that enforces voters' privacy by design. The analysis shows that the process of credential issuance, maintenance and verification is as important as the REV system itself. The security of a REV system is tightly bound to the IdM system which defines the set of eligible voters.

The architecture proposed in this thesis implements an authentication process according to the W3C principles of SSI. Thus, voters benefit from *(i)* not relying on TTPs for securing their digital identity, *(ii)* selectively approving and disclosing credentials every time they are requested, *(iii)* having the assurance that credentials are only shared with their consent and *(iv)* an overall more secure authentication due to the removal of email accounts in the process. The architecture provides a new interface for configuring an election that enables the VA to configure a credential-based authentication process. Furthermore, the prototype provides an application for issuing credentials for the VA as well as storing and disclosing credentials for the voters. The IdP can verify the credentials without connecting to the credential issuer, making the verification process exclusively between the voter and the IdP.

Introducing a radically different mechanism for issuing credentials has much greater effects on our governmental system than e-voting alone. It affects most interactions with digital services and the footprint that users leave behind. As the urge of users for a privacy-preserving digital identity interferes with the business model of many large corporations, building a decentralized, interoperable identity infrastructure has to be developed as open-source platforms by government funded developers.

9.1 Future Work

Inputs for future work mainly focus on the IdP's role as a TTP and a SPOF. Having a decentralized IdM system offers new possibilities that may solve the dilemma between being trustless while guaranteeing privacy.

9.1.1 IdP as TTP

The IdP remains a TTP. Since a public permissioned network is used to manage DIDs and the authentication configuration is publicly available, the public can verify if a given VP makes a voter eligible or not. In comparison to Provotum 3.0, a list of eligible voters is secretly passed as a set of email addresses making the verification process only available to the IdP. Email addresses do not allow a cryptographic proof of ownership. Thus, it is very difficult for the IdP to prove that only honest signatures were created. Using DIDs and a public data registry lifts this limitation. There are two possible solutions that can make the role of the IdP trustless. Both of them are explained in more detail in the following two sections.

Single IdP Generates a Large ZKP

Voters can blindly sign a public key with the IdP if they can disclose the requested credentials. These VCs are bundled and presented in the form of a VP. At the end of an election, the IdP has a valid VP for every participating voter. To finalize an election with N votes, the IdP must generate a proof that states that it possesses N valid VPs and submit it to the PBB. An election is only considered valid if the IdP can generate such a proof. If the IdP is not able to provide this proof, the public knows that the IdP has blindly signed public keys to itself or a vote-buying party.

The finalization process on the PBB requires the IdP to prove that each VP contains *(i)* the election identifier and that *(ii)* the list of credentials is valid. Thus, the proof must state that each credential in the VP is issued from one of the trusted IA and that the claim meets the claim constraints. The proof must be constructed as a ZKP since the actual VPs must not be disclosed. Otherwise, the voters' privacy is endangered as explained in Section 8.1.1. Such a design solves the trust assumption since the IdP is not able to arbitrarily generate valid VPs since they must be signed by the IA defined in the authentication configuration. However, the feasibility of such a complex proof and the associated computing power needed to generate and validate has to be evaluated in a future project.

Generating the ZKP requires that each VP is tagged with the election identifier, otherwise, the IdP can reuse VPs from previous elections. As further explained in Section 8.3.1, this is not yet possible with the Veramo framework.

N out of M IdP Signatures

Another possible solution to the TTP vulnerability is to split the responsibility of a single IdP to a group of IdPs. To cast a valid vote, a voter must obtain blind signatures from multiple IdPs. The threshold of minimal signatures needed must be chosen accordingly. If it is a minority, a voter can obtain valid signatures twice. Thus, the number of signature must be a majority of all IdPs. As illustrated in Figure 9.1, if the threshold is set to 3 in a set of 5 IdPs, a voter is not able to get 3 valid signatures twice as long as all

IdPs are honest. However, if only one of the IdPs is malicious or compromised and the IdPs do not communicate the used VCs with each other, the system can be attacked. This would allow a voter to request a valid set of signatures twice. However, if the IdPs collaboratively maintain a list of used VCs, such an attack of a minority of dishonest nodes can be mitigated. How such a list can be maintained privately among multiple IdPs is further discussed in Section 9.1.2. A similar problem occurs in a system where 5 out of 5 signatures are required as shown in Figure 9.2. If only one of the IdPs is refusing to authenticate voters, a voter is not able to obtain the required signatures and cannot cast a vote. This problem is related to the Byzantine General Problem [16] which states that more than $2/3$ of the IdP must be honest. Analysing a suitable threshold has to be further investigated.

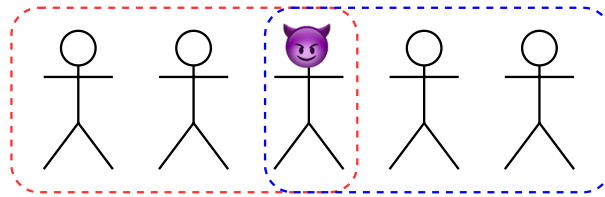


Figure 9.1: SPOF remains in a 3 out of 5 signature scheme as long as IdP do not maintain a common state of which VCs have been used

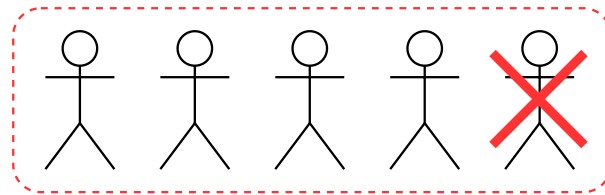


Figure 9.2: SPOF remains in a 5 out of 5 signature scheme

9.1.2 IdP as SPOF

The IdP remains a Single Point of Failure (SPOF) in the implemented prototype. If the IdP is not available, voters can no longer request the necessary blind signature and therefore, they cannot cast a valid vote. Different solutions were investigated to solve this vulnerability. The key finding of the analysis is that the architecture around the role of the IdP has to make a trade-off between decentralization and potential privacy leaks.

Single IdP There are fewer people involved that may execute an insider attack and collide with one of the IA to learn which voters have registered.

Multiple IdP A private permissioned DL can be implemented to create a consensus among multiple IdPs for maintaining the list of used VCs. It has to be private and the IA can not be part of it since the IAs must not learn which credentials have been used. Each IdP in that network provides an authentication service where voters can obtain a blinded signature removing the SPOF and the threat of censorship. Using a shared state among all IdPs makes sure that a VC can only be used once

per election and previously used VC cannot be altered at any time. As explained in Section 9.1.1, a VC would not only have to be signed by a single IdP and instead by a majority of the IdPs. Such a design benefits from being more DoS resilient as more authentication services are available and the threat of censorship is lowered. The disadvantage is that the privately managed list of used VCs may be leaked due to the higher number of services and interfaces involved.

9.1.3 ZKP-based Authentication Process

As described in the Section 8.1, depending on the defined authentication configuration, the IdP may be able to uniquely identify a voter based on the disclosed credentials. Incorporating ZKPs for the disclosed values in the authentication process helps to further reduce this risk. For example, it would be sufficient if a voter provides proof that his birthday is more than 18 years ago without revealing the actual date of birth. The necessary infrastructure and the appropriate type of ZKP has to be further investigated.

Note that other projects such as Mattr Global use ZKPs for selectively disclose claims from VCs that contain multiple credentials¹. These proofs make sure that unneeded claims within the same VC are not revealed during the disclosure of a VC with multiple claims. The use of ZKPs only guarantees that the minimal number of claims can be disclosed even if VCs contain multiple claims. The current implementation uses atomic credentials which achieves the same. However, both strategies still reveal the actual value.

9.1.4 Mobile Client for DID and VC Maintenance

Currently, Veramo cloud agents are used to store DIDs and VCs. This is not practical and user-friendly for voters as they must deploy an agent. A mobile client with storing the DIDs and VCs locally on the device would not require a cloud setup.

9.1.5 Credential Registries and Governance Frameworks

When creating an authentication configuration, the DIDs of the IAs must be known to the VA. This list of trusted IA is difficult to maintain and a change may not be noticed immediately. This is a missing component and an ongoing challenge in SSI. As explained in Section 8.3.4, public credential registries and governance frameworks can address this issue.

¹<https://learn.mattr.global/tutorials/verify/zkp-enabled/zkp-intro>

Bibliography

- [1] Andreas Abraham, Stefan More, Christof Rabensteiner, and Felix Hörandner. Revocable and offline-verifiable self-sovereign identities. <https://www.republik.ch/2021/01/28/die-probleme-mit-der-schweizer-e-identitaet>, 2020. (Accessed on 14.5.2021).
- [2] Drummond Reed Alex Preukschat. Self-sovereign identity. <https://www.manning.com/books/self-sovereign-identity>, December 2020. (Accessed on 7.2.2021).
- [3] Bruno Rodrigues Alexander Hofmann, Christian Killer. Security analysis and improvements of a blockchain-based remote electronic voting system. <https://www.merlin.uzh.ch/contributionDocument/download/14015>, November 2020. (Accessed on 9.5.2021).
- [4] T. Berners-Lee, R. Fielding, and L. Masinter. Uniform resource identifier (uri): Generic syntax. <https://tools.ietf.org/html/rfc3986>, January 2005. (Accessed on 7.2.2021).
- [5] Carole Cadwalladr and Emma Graham-Harrison. Revealed: 50 million facebook profiles harvested for cambridge analytica in major data breach. <https://www.theguardian.com/news/2018/mar/17/cambridge-analytica-facebook-influence-us-election>, 2018. (Accessed on 9.5.2021).
- [6] Adrienne Fichter. Die probleme mit der schweizer e-identität. <https://www.republik.ch/2021/01/28/die-probleme-mit-der-schweizer-e-identitaet>, January 2021. (Accessed on 14.5.2021).
- [7] Communication Systems Group. E-voting: Blockchain-based remote electronic voting. <http://www.csg.uzh.ch/csg/en/research/evoting.html>, 2020. (Accessed on 9.5.2021).
- [8] Harry Halpin. A critique of immunity passports and w3c decentralized identifiers. <https://arxiv.org/pdf/2012.00136.pdf>, November 2020. (Accessed on 14.5.2021).
- [9] Daniel Hardman. Didcomm messaging. <https://identity.foundation/didcomm-messaging/spec/>. (Accessed on 9.5.2021).
- [10] Daniel Hardman. Didcomm messaging. <https://identity.foundation/did-siop/>, January 2021. (Accessed on 12.2.2021).

- [11] Rae Hodge. Apple: 1, spam: 0? how a new email feature aims to keep spammers off your trail. <https://www.cnet.com/how-to/apple-1-spam-0-how-a-new-email-feature-aims-to-keep-spammers-off-your-trail/>, June 2021. (Accessed on 14.6.2021).
- [12] Lance James. Phishing exposed. <https://dl.acm.org/doi/book/10.5555/1121597#secAbs>, 2005. (Accessed on 14.5.2021).
- [13] Hugo Junker and Jun Pang. Bulletin boards in voting systems: Modelling and measuring privacy. <https://ieeexplore.ieee.org/abstract/document/6045953/metrics#metrics>, August 2011. (Accessed on 14.5.2021).
- [14] Christian Killer, Bruno Rodrigues, Eder John Scheid, Muriel Franco, Moritz Eck, Nik Zaugg, Alex Scheitlin, and Burkhard Stiller. Provotum: A blockchain-based and end-to-end verifiable remote electronic voting system. https://www.researchgate.net/profile/Christian_Killer/publication/345319094_Provotum_A_Blockchain-based_and_End-to-end_Verifiable_Remote_Electronic_Voting_System/links/5fa3ae75299bf10f73250fe3/Provotum-A-Blockchain-based-and-End-to-end-Verifiable-Remote-Electronic-Voting-System.pdf, November 2020. (Accessed on 27.12.2021).
- [15] Thomas Kurt, Jennifer Pullman, Kevin Yeo, Ananth Raghunathan, Patrick Gage Kelley, Luca Invernizzi, Borbala Benko, Tadek Pietraszek, Sarvar Patel, Dan Boneh, and Elie Bursztein. Protecting accounts from credential stuffing with password breach alerting. <https://dl.acm.org/doi/10.1145/1102120.1102168>, 2019. (Accessed on 14.5.2021).
- [16] Leslie Lamport, Robert Shostak, and Marshall Pease. The byzantine generals problem. <https://lamport.azurewebsites.net/pubs/byz.pdf>. (Accessed on 14.5.2021).
- [17] Dr. Christian Lundkvist, Rouven Heck, Joel Torstensson, Zac Mitton, and Michael Sena. Uport: A platform for self-sovereign identity. https://blockchainlab.com/pdf/uPort_whitepaper_DRAFT20161020.pdf, October 2016. (Accessed on 14.5.2021).
- [18] Zoltan Andras Lux, Dirk Thatmann, Sebastian Zickau, and Felix Beierle. Distributed-ledger-based authentication with decentralized identifiers and verifiable credentials. <https://arxiv.org/pdf/2006.04754.pdf>, June 2020. (Accessed on 28.3.2021).
- [19] Sheera Frenkel Mike Isaac. Facebook security breach exposes accounts of 50 million users. <https://www.nytimes.com/2018/09/28/technology/facebook-hack-data-breach.html>, September 2018. (Accessed on 7.2.2021).
- [20] Arvind Narayanan and Vitaly Shmatikov. Fast dictionary attacks on passwords using time-space tradeoff. <https://dl.acm.org/doi/10.1145/1102120.1102168>, November 2005. (Accessed on 14.5.2021).
- [21] David Neal. An illustrated guide to oauth and openid connect. <https://developer.okta.com/blog/2019/10/21/illustrated-guide-to-oauth-and-oidc>, October 2019. (Accessed on 7.2.2021).

- [22] Jeff Peters. What is saml and how does it work? <https://www.varonis.com/blog/what-is-saml/>, March 2020. (Accessed on 7.2.2021).
- [23] Drummond Reed, Manu Sporny, Dave Longley, Christopher Allen, and Ryan Grant and Markus Sabadello. Decentralized identifiers (dids) v1.0. <https://www.w3.org/TR/did-core/>, November 2021. (Accessed on 12.2.2021).
- [24] Thomas Reid. Essays on the intellectual powers of man. <https://www.cambridge.org/core/books/essays-on-the-intellectual-powers-of-man/CE5992C23D18DA1E9ABD8842D1E91BC0>, 1786. (Accessed on 14.5.2021).
- [25] Josh Fruhlinger and Roger A. Grimes. What is oauth? how the open authorization framework works. <https://www.csoonline.com/article/3216404/what-is-oauth-how-the-open-authorization-framework-works.html>, September 2020. (Accessed on 7.2.2021).
- [26] Markus Sabadello. A universal resolver for self-sovereign identifiers. <https://medium.com/decentralized-identity/a-universal-resolver-for-self-sovereign-identifiers-48e6b4a5cc3c>, November 2017. (Accessed on 14.5.2021).
- [27] James Sanders and Dan Patterson. Facebook data privacy scandal: A cheat sheet. <https://www.techrepublic.com/article/facebook-data-privacy-scandal-a-cheat-sheet/>. (Accessed on 9.5.2021).
- [28] Manu Sporny, Dave Longley, and David Chadwick. Verifiable credentials data model 1.0. <https://www.w3.org/TR/vc-data-model/>, November 2019. (Accessed on 12.2.2021).
- [29] Manu Sporny, Dave Longley, Gregg Kellogg, Markus Lanthaler, Pierre-Antoine Champin, and Niklas Lindström. Json-ld 1.1 - a json-based serialization for linked data. <https://www.w3.org/TR/json-ld11/>, July 2020. (Accessed on 12.2.2021).
- [30] Ori Steele and Manu Sporny. Did specification registries. <https://www.w3.org/TR/did-spec-registries/>. (Accessed on 9.5.2021).
- [31] Ori Steele and Manu Sporny. Did specification registries. <https://www.w3.org/TR/did-spec-registries/#did-methods>. (Accessed on 9.5.2021).
- [32] Oliver Terbu, Ivan Basart, Kyle Den Hartog, Christian Lundkvist, David Stark, Dmitri Zagidulin, Danny Strockis, and Ori Steele. Self-issued openid connect provider did profile v0.1. <https://identity.foundation/did-siop/>, November 2020. (Accessed on 12.2.2021).
- [33] unknown. Using privacy-preserving zkp credentials on the mattr platform. <https://mattr.global/using-privacy-preserving-zkp-credentials-on-the-mattr-platform/>, September 2020. (Accessed on 9.5.2021).
- [34] unknown. Bbs+ signatures 2020. <https://w3c-ccg.github.io/ldp-bbs2020/>, Mai 2021. (Accessed on 14.5.2021).

- [35] Lane Wagner. Shamir's secret sharing step-by-step. <https://qvault.io/2020/08/18/very-basic-shamirs-secret-sharing/>, August 2001. (Accessed on 12.2.2021).
- [36] World wide web consortium (w3c). <http://webdiy.org/w3c/>. (Accessed on 7.2.2021).

Abbreviations

IdP	Identity Provider
IdM	Identity Management
DL	Distributed Ledger
DLT	Distributed Ledger Technology
VA	Voting Authority
SSO	Single Sign-On
OTP	One Time Password
TTP	Trusted Third Party
BC	Blockchain
PBB	Public Bulletin Board
DKG	Distributed Key Generation
PoA	Proof-of-Authority
DoS	Denial-of-Service
REV	Remote Election Voting
RF	Receipt-Freeness
ZKP	Zero Knowledge Proof
DID	Decentralized Identifier
VC	Verifiable Credential
VP	Verifiable Presentation
SDR	Selective Disclosure Request
SSI	Self Sovereign Identity
SIOP	Self Issued OpenID Connect Provider

Glossary

Identity Context The original identity context is defined by the issuer of a credential. For example the original identity context of a student card is defined as all the systems that directly interact with the application that issued the card and can access the source of truth of that credential. When a student registers for an student discounted application X, the student card is shared and in return a user account is created within the application X. This new user account is loosely coupled to the original identity context meaning that a change in the original context may not be noticed.

Proxy Identity A proxy identity is an account with credentials that are loosely coupled to other identity contexts. The credentials associated with that account originate from another identity context.

DID A DID is a type of identifier that is resolvable and globally unique across multiple networks.

DID subject The metadata information stored in a DID document represents the DID subject.

DID controller The DID controller can verify the ownership of a DID. In many cases the DID subject is the same as the DID controller. However, in the example of an infant, it makes sense that the DID controller is the parent until the child becomes mature. At this point the control of the DID can be changed to a new key pair. The actual DID does not change in the process and all the associated metadata must not be updated.

DID resolver A program that retrieves a DID document for a given DID. A resolver may support one or multiple networks. For example a id resolver for the Ethereum blockchain is able to construct a DID document from a DID registry in the form of a Smart Contract on the Ethereum blockchain.

DID resolution The process of resolving a DID document for a given DID.

DID document A DID is associated with exactly one DID document that contains the metadata about the resource.

List of Figures

2.1	Centralized identity model	6
2.2	Federated identity model in the example of OAuth 2.0	7
2.3	The decentralized identity model splits the roles of issuing, holding and verifying among three separate entities.	8
2.4	The locus of control that happens in the transition from account-based identity models to the self-sovereign identity model [2]	9
4.1	DIDs as self-certifying identifiers enable a strong binding between the public key, identifier and controller	16
4.2	DID documents enable holders to execute a key rotation without changing the identifier	16
4.3	Overview of the DID architecture proposed by the W3C CCG	18
4.4	Relationship between identifier, metadata and the subject	21
4.5	Original identity context ends whenever a proxy identity is created.	24
5.1	Provotum 3.0 stakeholders	26
6.1	Provotum 3.0 extended with a decentralized IdM system	33
7.1	Overview of Provotum’s SSI repositories	39
7.2	UI for issuing an atomic VC that uses Schema.org vocabulary to define a person’s postal code. As a result, a JWT is produced that can be displayed as a QR Code and imported by the holder to its own agent.	43
7.3	UI for creating an authentication configuration for an election. This screenshot shows how an authentication configuration tuple is added. With this credential and constraint configured, the voter must disclose a VC issued by the city administration office of Zurich and the value of this claim must be one of the defined postal codes. The result of this form corresponds to the data shown in Figure 7.3.	45

7.4	Sequence diagram for IdP authentication process	46
7.5	UI for selectively disclosing VCs. After scanning the SDR, the voter is presented with a list of required credentials that must be disclosed. Each credential must be permitted to be disclosed.	47
8.1	Credential registry for public VCs	56
8.2	A governance framework defines the rules and policies to become a verified issuer for a given context. The example shows that the Swiss government could publicly maintain a list of verified passport IAs in the form of a credential registry. Thus, the list of eligible IAs can be discovered and verified by only knowing the DID of the Swiss government.	57
9.1	SPOF remains in a 3 out of 5 signature scheme as long as IdP do not maintain a common state of which VCs have been used	61
9.2	SPOF remains in a 5 out of 5 signature scheme	61

List of Tables

3.1	Reserved tokens in the JSON-LD namespace that are relevant for this paper	13
4.1	Design goals for decentralized gloabl identifier	18
4.2	Components of DID architecture suggested by the W3C CCG	19
4.3	Types of DIDs	21
5.1	Threat model of Provotum 3.0 in regards to IdM	32
8.1	Threat model evaluation after the application of the new design	50

Appendix A

Installation Guidelines

The source code for Provotum 3.0 with the integrated SSI-based authentication process can be found in the Provotum Github organization¹. All repositories that affected by the SSI-based authentication process are available in a branch called `vSSI`.

To simplify running Provotum 3.0 with the SSI-based authentication process, a docker compose² manifest accompanies the project, ensure Docker version 19.03.13 and Docker Compose are installed 1.27.4 or higher. For local development, NodeJs³ 12.18.03 and Rust⁴ 1.46.0 or higher are required.

There are two demos available on the infrastructure repository⁵. One is illustrating the E2E voting process including a credential-based authentication process. The other demo is independent of Provotum and illustrates the issuance, storage and disclosure of credentials. Follow the instructions in the README to run the demos.

¹<https://github.com/provotum>

²<https://docs.docker.com/compose/>

³<https://nodejs.org/en/>

⁴<https://www.rust-lang.org/>

⁵<https://github.com/provotum/provotum-infrastructure/tree/vSSI>

Appendix B

Contents of the CD

This thesis is accompanied by an archive of the following items:

- A PDF file of this report
- The \LaTeX source code of this report.
- The diagrams of this report.
- The editable source files of the diagrams used in the report which can be opened with Draw.io¹.
- The source code of the implemented prototype.

¹<https://app.diagrams.net>