05-Self_Sovereign_Identity_SSI

Introduction

This note gather information obtained during my discovery of **What is a Self Sovereign Identity SSI?** (called SSI).

Study roadmap

- [x] 1) Understand what is Self Sovereign Identity SSI?
- [x] 2) Create a diagram flow to understand the exchange flows.
- [ ] 3) Create a POC.
- [ ] 4) Create the blog post.

Data source

- Blog post

Global sequences flow understood

Terminology

Decentralized identifier (DID)

A globally unique persistent identifier that does not require a centralized registration authority and is often generated and/or registered cryptographically.

Source: https://w3c.github.io/did-core/#terminology

DID document

A set of data describing the DID subject, including mechanisms, such as cryptographic public keys, that the DID subject or a DID delegate can use to authenticate itself and prove its association with the DID.

Example:

A DID is a simple text string consisting of three parts: 1) the `did` URI scheme identifier, 2) the identifier for the DID method, and 3) the DID method-specific identifier.

Figure 1 A simple example of a decentralized identifier (DID)

The example DID above resolves to a DID document. A DID document contains information associated with the DID, such as ways to cryptographically authenticate a DID controller.

EXAMPLE 1: A simple DID document

```
{
  "@context": [
    "https://www.w3.org/ns/did/v1",
    "https://w3id.org/security/suites/ed25519-2020/v1"
  ]
  "id": "did:example:123456789abcdefghi",
  "authentication": [{
    // used to authenticate as did:...fghi
    "id": "did:example:123456789abcdefghi#keys-1",
    "type": "Ed25519VerificationKey2020",
    "controller": "did:example:123456789abcdefghi",
    "publicKeyMultibase": "zH3C2AVvLMv6gmMNam3uVAjZpfkcJCwDwnZn6z3wXmqPV"
  }]
}
```

Source: https://w3c.github.io/did-core/#terminology

Verifiable credential

A standard data model and representation format for cryptographically-verifiable digital credentials as defined by the W3C Verifiable Credentials specification.

Example of verifiable credential issued: A national identity card.

Note about a **verifiable presentation** of a **verifiable credential**: "a verifiable presentation expresses data from one or more verifiable credentials, and is packaged in such a way that the authorship of the data is verifiable. If verifiable credentials are presented directly, they become verifiable presentations".

Source: https://w3c.github.io/did-core/#terminology

Holder

A role an entity might perform by possessing one or more verifiable credentials and generating presentations from them. A holder is usually, but not always, a subject of the verifiable credentials they are holding. Holders store their credentials in credential repositories (wallet).

Source: https://www.w3.org/TR/vc-data-model/#terminology

Issuer

A role an entity can perform by asserting claims about one or more subjects, creating a verifiable credential from these claims, and transmitting the verifiable credential to a holder.

Example of issuer: Government.

Source: https://www.w3.org/TR/vc-data-model/#terminology

Verifier

A role an entity performs by receiving one or more verifiable credentials, optionally inside a verifiable presentation for processing. Other specifications might refer to this concept as a relying party.

Example of verifier: Bank.

Source: https://www.w3.org/TR/vc-data-model/#terminology

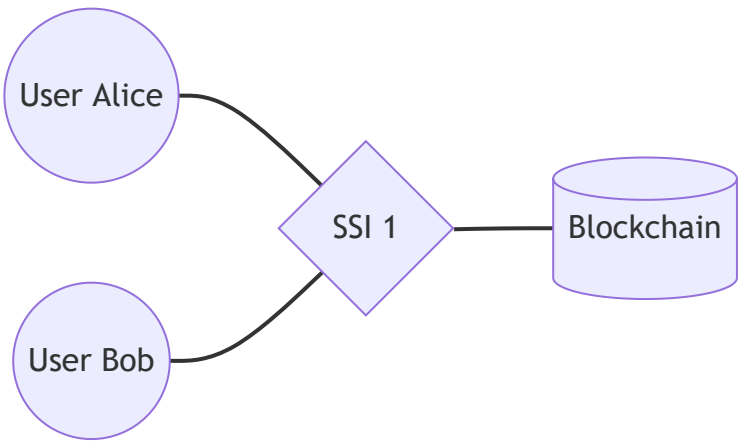
Blockchain

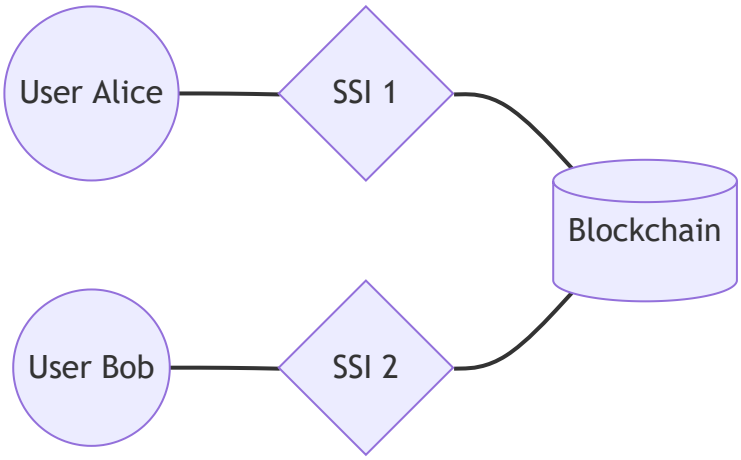
A blockchain is a growing list of records, called blocks, that are linked together using cryptography. Each block contains a cryptographic hash of the previous block, a timestamp, and transaction data.

Source: https://en.wikipedia.org/wiki/Blockchain

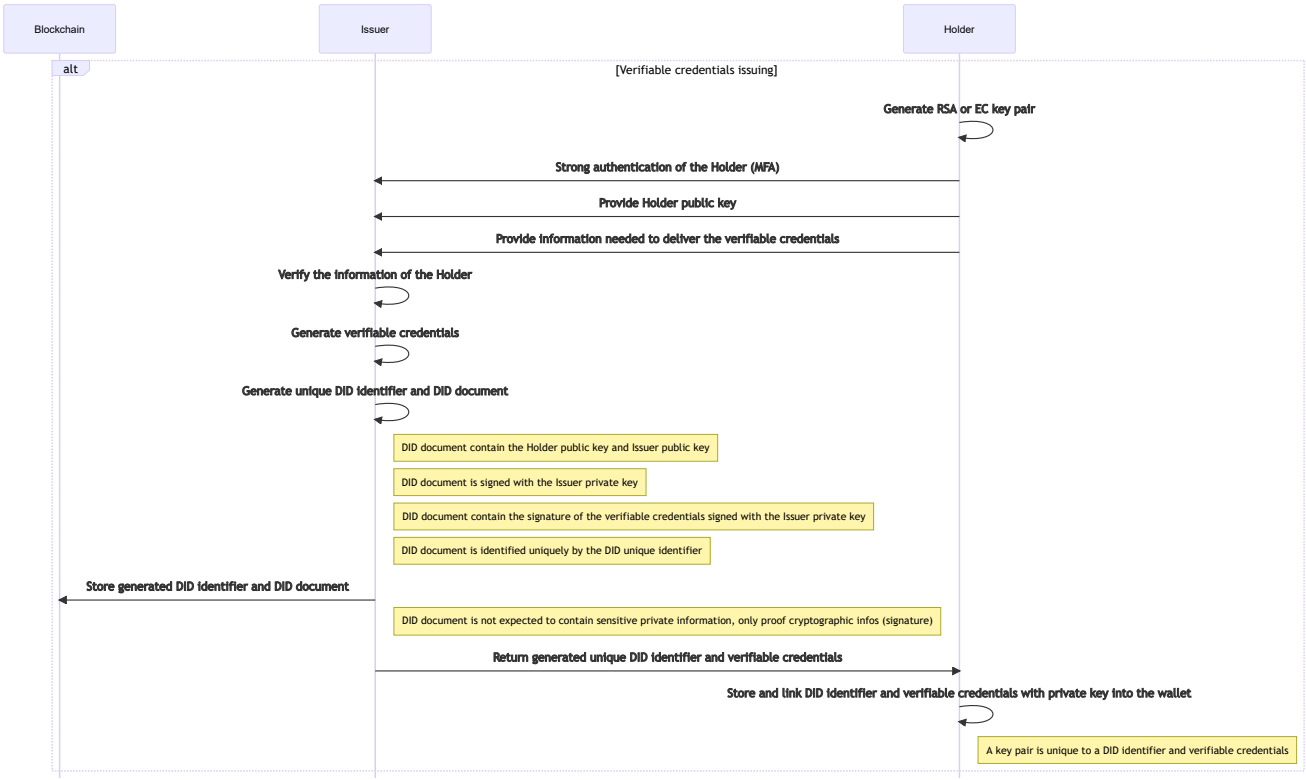
High level view

The two schemas below provide a high-level view:

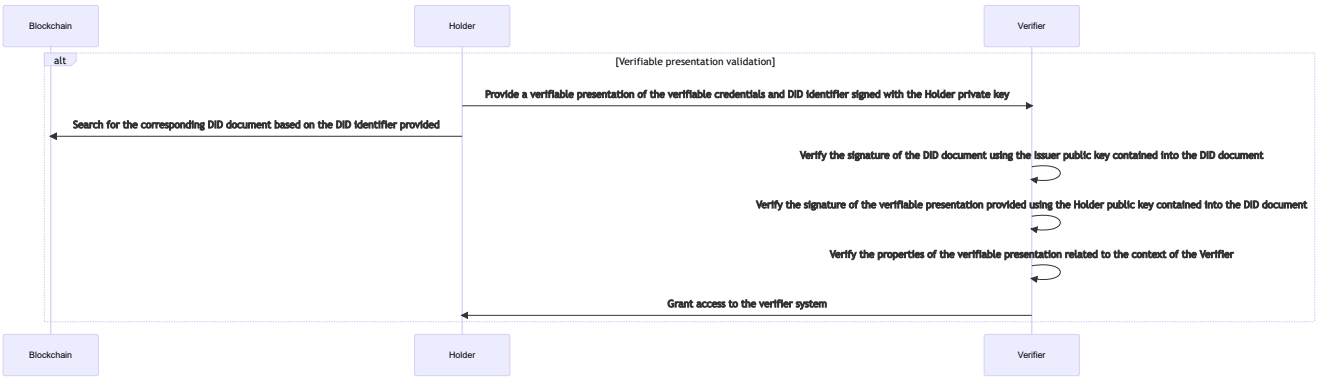




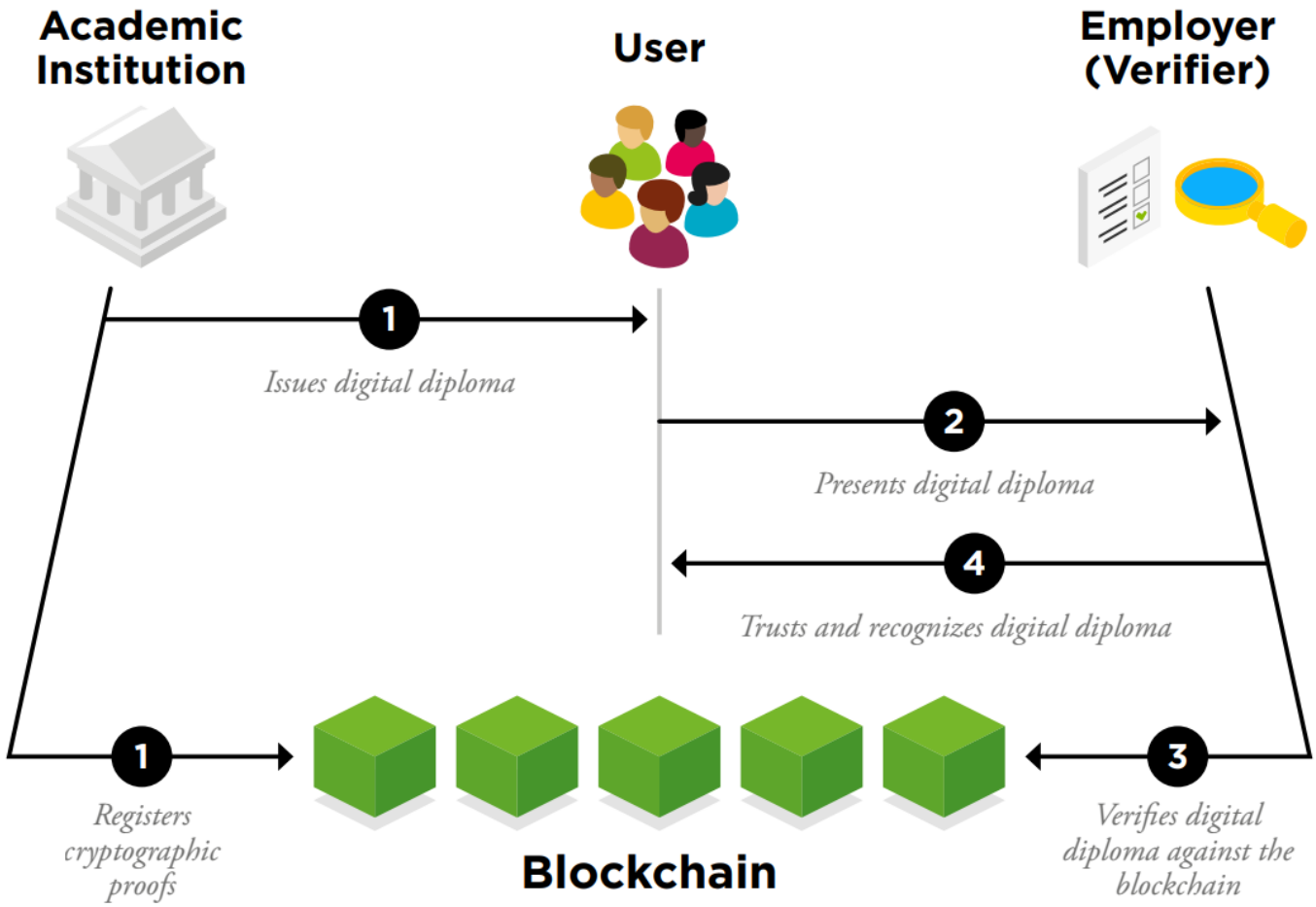
Verifiable credentials issuing flow

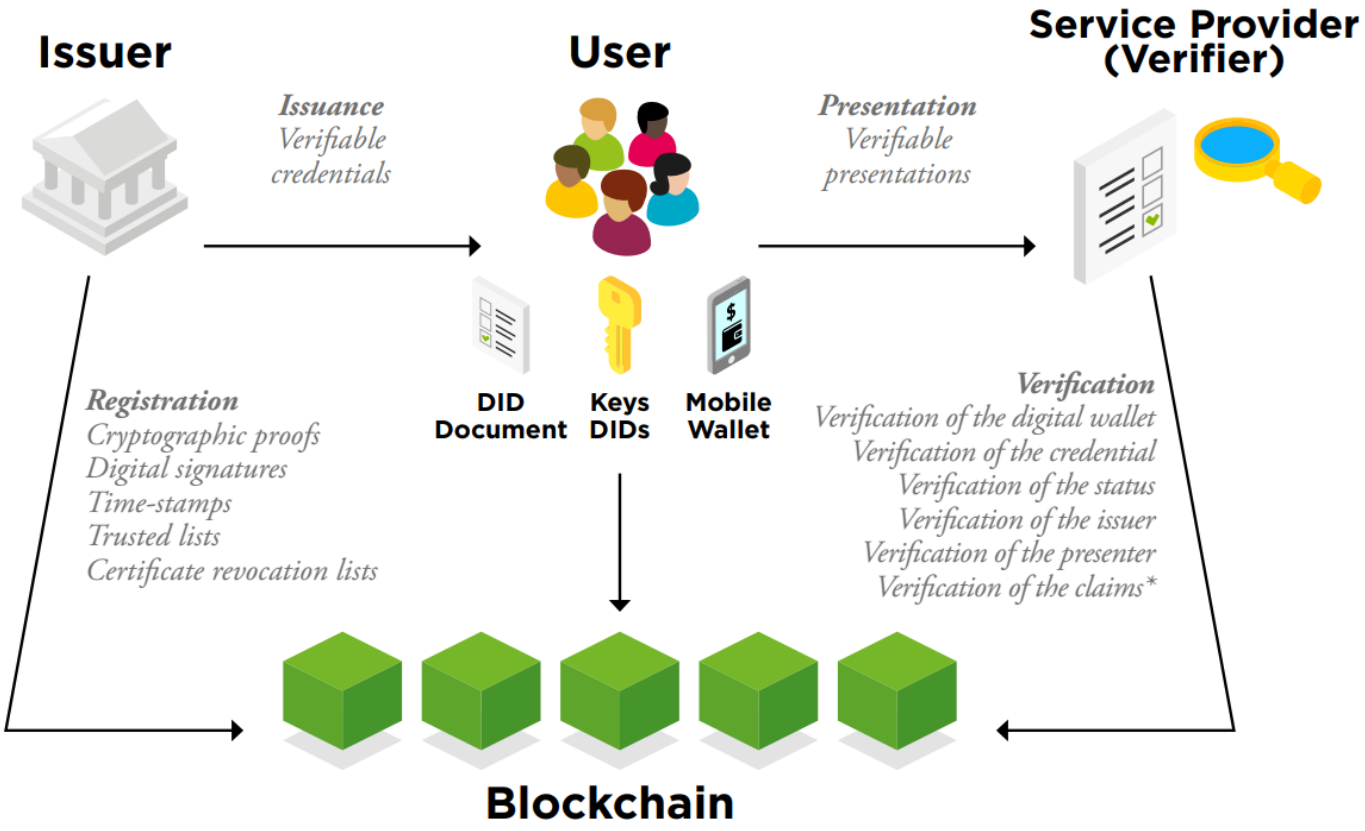


Verifiable presentation validation flow



High level flows example





Source (page 28/29 of the PDF): https://publications.iadb.org/publications/english/document/Self-Sovereign-Identity-The-Future-of-Identity-Self-Sovereignity-Digital-Wallets-and-Blockchain.pdf

SSI key aspects

- It is possible to verify a verifiable credentials even if the Issuer is not available or do not exist anymore.
- There is no direct communication between an Issuer and a Verifier during the issuing of a verifiable credentials or validation of a verifiable presentation.
- Once stored in a blockchain, a DID document cannot be modified or deleted.
- Holder fully control the level of information provided to a Verifier within the verifiable credentials provided to it.
- Digital signature, using asymmetric RSA/EC key pair, is a critical pillar of SSI and trust/integrity foundation.
- Availability of the blockchain used by the Issuer/Verifier is a critical pillar of SSI.

Security attention points noticed

- Privacy of data stored into the DID document.
- Privacy disclosure into DID identifier.
- Strengh of the cryptographic algorithms used for keys and signature.
- Quality of the implementation on the **credential verification operation** performed by the Verifier: signature validity, signature coverage (like unsigned claims), type of credentials received against expected, expiration date, revokation state, issuer expected, alteration, credential replay attack, etc:
    - Include also the validation of the integrity of the elements stored outside a *verifiable credential* like mentionned in the Verifiable Credentials spec:

§ 8.2 Content Integrity Protection

This section is non-normative.

Verifiable credentials often contain URLs to data that resides outside of the verifiable credential itself. Linked content that exists outside a verifiable credential, such as images, JSON-LD Contexts, and other machine-readable data, are often not protected against tampering because the data resides outside of the protection of the proof on the verifiable credential. For example, the following highlighted links are not content-integrity protected but probably should be:

EXAMPLE 34: Non-content-integrity protected links

```
{
  "@context": [
    "https://www.w3.org/2018/credentials/v1",
    "https://www.w3.org/2018/credentials/examples/v1"
  ],
  "id": "http://example.edu/credentials/58473",
  "type": ["VerifiableCredential", "AlumniCredential"],
  "credentialSubject": {
    "id": "did:example:ebfeb1f712ebc6f1c276e12ec21",
    "image": "https://example.edu/images/58473",
    "alumniOf": {
      "id": "did:example:c276e12ec21ebfeb1f712ebc6f1",
      "name": [{
        "value": "Example University",
        "lang": "en"
      }, {
        "value": "Exemple d'Université",
        "lang": "fr"
      }]
    }
  },
  "proof": { ... }
}
```

- Attacks on the wallet itself targeting the holder side, protection of the secret keys like ensuring that access to a credentials require entering an secret or a physical action like pushing a button (ref in the Verifiable Credentials spec)
- Attacks on deserialization processes used on Issuer and Verifier sides.
- Attack targeting to perform unexpected alteration of DID documents in the Verifiable Data Registry (blockchain) because the VDR spec define the following statement in its Trust Model: *"All entities trust the verifiable data registry to be tamper-evident and to be a correct record of which data is controlled by which entities."*
- Misusage of Verifiable Credentials for authorization decision. Indeed, Verifiable Credentials are intended as a means of reliably identifying subjects and not for authorization purpose (see this section for the VDR spec)
- Specific points to Bearer Credentials: Are single-use where possible, do not contain personally identifying information, are not unduly correlatable.

- Secure protocols are not used on Issuer and Verifier sides.

Remarks

- At present it is not possible to use any wallet with any SSI solution. Each solution is locked to its own wallet (October 2021 - source).
- A SSI provider (or solution) seems to provide the Issuer and Verifier parts of a Verifiable Credentials and is not seems to be possible to currently mix two different SSI provider (one for Issuer and one for Verifier) even if it is theoretically possible:

righettod · March 21, 2022 - 16:59 · *Reply→*

Thank you very much for such great infos and blog posts!!!!

I have a small question to ensure my correct understanding:

Currently it is not possible to use a SSI provider A or language verification API to verify the Verifiable Credentials issued by a SSI provider B?

Thank you very much in advance for your feedback

damienbod · March 21, 2022 - 19:57 · *Reply→*

Hi righttod, If the SSI A can understand the VC produced by the SSI B and both can use the same ledger where the DID is persisted and communicate using an agent using a common protocol then these can work together when the verifier knows and trusts the issuers DID

Greetings Damien

Source: https://damienbod.com/2021/10/11/challenges-to-self-sovereign-identity/comment-page-1/#comment-139146

- A SSI provider (or solution) seems to include the API to Issue + Verify Verifiable Credentials, the wallet manager and the blockchain to store DID identifier + DID document.

Study notes

What is SSI?

- **Self-Sovereign Identity** (called **SSI**) is an emerging solution built on blockchain technology for solving digital identities which gives the management of identities to the users and not organisations.
- It does not solve the process of authenticating users in applications.

- The credentials DIDs (Digital Identity, Decentralized identifiers) are stored to a blockchain and to verify the credentials you need to search in the same blockchain network.

Party involved in SSI?

Digital Identity, Decentralized identifiers (DIDs)

> Spec: https://w3c.github.io/did-core/

- A digital identity can be expressed as a universal identifier which can be owned and can be publicly shared.
- A digital identity provides a way of showing a subject (user, organisation, thing), a way of exchanging credentials to other identities and a way to verify the identity without storing data on a shared server.
- A digital identity can be found using decentralized identifiers (DID) and this has working group standards in the process of specifying this.
- The DIDs are saved to a blockchain network which can be resolved. The DIDs representing identities are published to a blockchain network.
- Decentralized identifiers (DIDs) are a new type of identifier that enables verifiable, decentralized digital identity.
- A DID refers to any subject (e.g., a person, organization, thing, data model, abstract entity, etc.) as determined by the controller of the DID.
- Specifically, while other parties might be used to help enable the discovery of information related to a DID, the design enables the controller of a DID to prove control over it without requiring permission from any other party.
- DIDs are URIs that associate a DID subject with a DID document allowing trustable interactions associated with that subject.
- Each DID document can express cryptographic material, verification methods, or services, which provide a set of mechanisms enabling a DID controller to prove control of the DID. Services enable trusted interactions associated with the DID subject. A DID might provide the means to return the DID subject itself, if the DID subject is an information resource such as a data model.
- The Decentralized Identifiers (DIDs) defined in this specification are a new type of globally unique identifier. They are designed to enable individuals and organizations to generate their own identifiers using systems they trust. These new identifiers enable entities to prove control over them by authenticating using cryptographic proofs such as digital signatures.

§ 1.1 A Simple Example

This section is non-normative.

A DID is a simple text string consisting of three parts: 1) the `did` URI scheme identifier, 2) the identifier for the DID method, and 3) the DID method-specific identifier.

Figure 1 A simple example of a decentralized identifier (DID)

The example DID above resolves to a DID document. A DID document contains information associated with the DID, such as ways to cryptographically authenticate a DID controller.

EXAMPLE 1: A simple DID document

```
{
  "@context": [
    "https://www.w3.org/ns/did/v1",
    "https://w3id.org/security/suites/ed25519-2020/v1"
  ]
  "id": "did:example:123456789abcdefghi",
  "authentication": [{
    // used to authenticate as did:...fghi
    "id": "did:example:123456789abcdefghi#keys-1",
    "type": "Ed25519VerificationKey2020",
    "controller": "did:example:123456789abcdefghi",
    "publicKeyMultibase": "zH3C2AVvLMv6gmMNam3uVAjZpfkcJCwDwnZn6z3wXmqPV"
  }]
}
```

Source: https://w3c.github.io/did-core/#a-simple-example

§ 1.3 Architecture Overview

This section is non-normative.

This section provides a basic overview of the major components of Decentralized Identifier architecture.

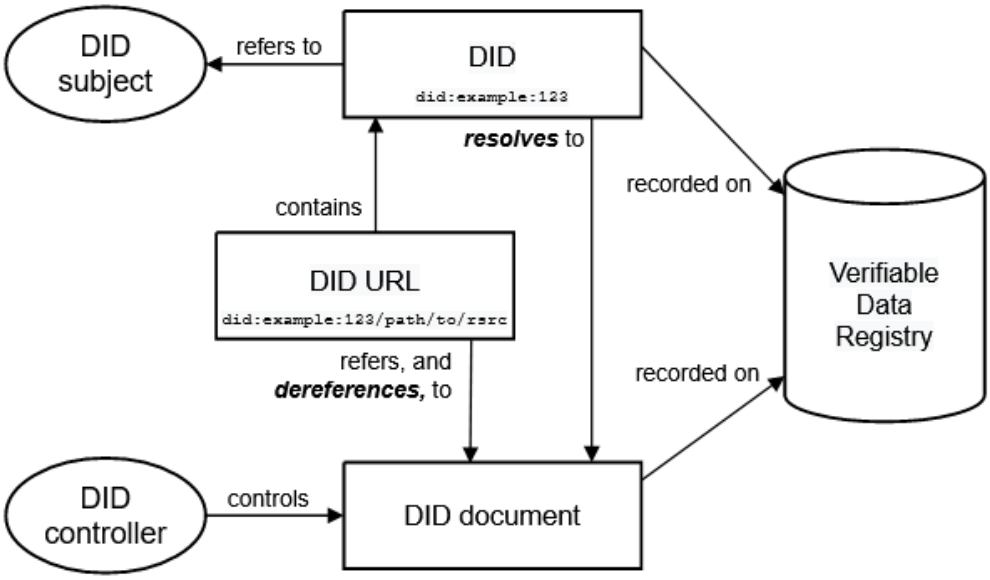


Figure 2 Overview of DID architecture and the relationship of the basic components. See also: narrative description.

Source: https://w3c.github.io/did-core/#architecture-overview

§ B.4 Referring to the DID document

The DID refers to the DID subject and *resolves to* the DID document (by following the protocol specified by the DID method). The DID document is not a separate resource from the DID subject and does not have a URI separate from the DID. Rather the DID document is an artifact of DID resolution controlled by the DID controller for the purpose of describing the DID subject.

This distinction is illustrated by the graph model shown below.

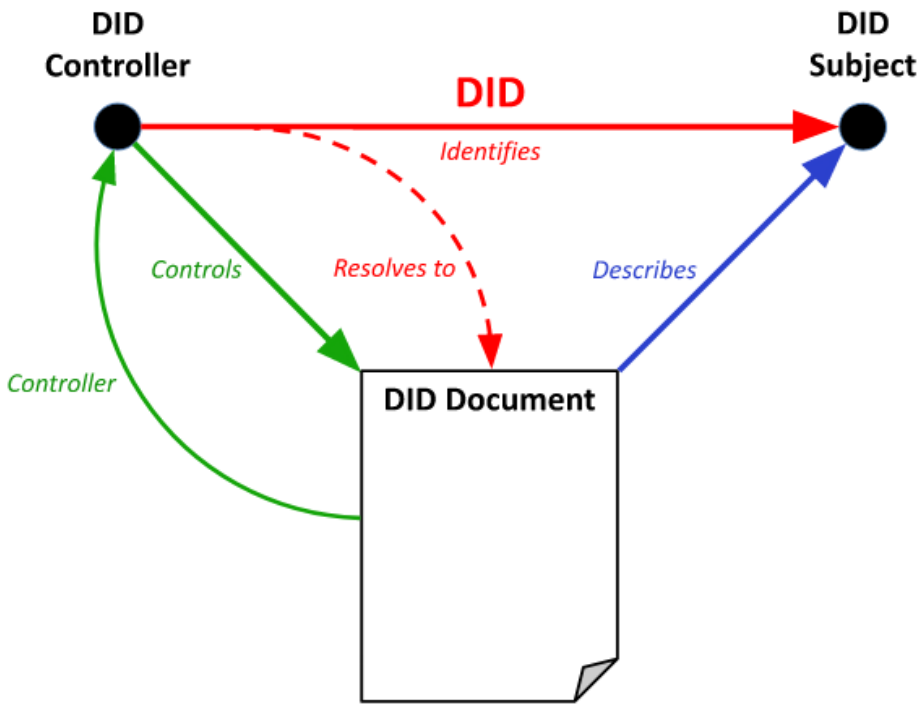


Figure 8 A DID is an identifier assigned by a DID controller to refer to a DID subject and resolve to a DID document that describes the DID subject. The DID document is an artifact of DID resolution and not a separate resource distinct from the DID subject. See also: narrative description.

Source: https://w3c.github.io/did-core/#referring-to-the-did-document

- A **DID document has exactly one DID which refers to the DID subject**. The DID is expressed as the value of the **id** property. This **property value is immutable for the lifetime of the DID document**.

Digital wallet

> Digital wallets seem to be vendor locked at the moment (march 2021) which will be problematic for mainstream adoption.

- A digital wallet is a database which stores all your verified credentials which you added to your data.
- You want to prevent all third party access to this wallet.
- A user can add or revoke credentials in the wallet.
- When you own a wallet, you would publish a **public key to a blockchain** network.

- A DID is returned representing the digital identity for this wallet and a public DID was saved to the network which can be used to authenticate anything interacting with the wallet.

Credentials, Verifiable credentials

> Spec: https://www.w3.org/TR/vc-data-model/

> This specification provides a standard way to express credentials on the Web in a way that is cryptographically secure, privacy respecting, and machine-verifiable.

- A verifiable credential is an immutable set of claims created by an issuer which can be verified.
- A verifiable credential has claims, metadata and proof to validate the credential.
- A credential can be saved to a digital wallet, so no data is persisted anywhere apart from the issuer and the digital wallet.
- The credential is **created by the issuer** for **the holder of the credential**.
- This credential is presented to the verifier by the holder from a digital wallet and the verifier can validate the credential using the issuer DID which can be resolved from the blockchain network.

§ 1.2 Ecosystem Overview

This section is non-normative.

This section describes the roles of the core actors and the relationships between them in an ecosystem where verifiable credentials are expected to be useful. A role is an abstraction that might be implemented in many different ways. The separation of roles suggests likely interfaces and protocols for standardization. The following roles are introduced in this specification:

holder
: A role an entity might perform by possessing one or more verifiable credentials and generating verifiable presentations from them. Example holders include students, employees, and customers.

issuer
: A role an entity performs by asserting claims about one or more subjects, creating a verifiable credential from these claims, and transmitting the verifiable credential to a holder. Example issuers include corporations, non-profit organizations, trade associations, governments, and individuals.

subject
: An entity about which claims are made. Example subjects include human beings, animals, and things. In many cases the holder of a verifiable credential is the subject, but in certain cases it is not. For example, a parent (the holder) might hold the verifiable credentials of a child (the subject), or a pet owner (the holder) might hold the verifiable credentials of their pet (the subject). For more information about these special cases, see Appendix C. Subject-Holder Relationships.

verifier
: A role an entity performs by receiving one or more verifiable credentials, optionally inside a verifiable presentation, for processing. Example verifiers include employers, security personnel, and websites.

verifiable data registry
: A role a system might perform by mediating the creation and verification of identifiers, keys, and other relevant data, such as verifiable credential schemas, revocation registries, issuer public keys, and so on, which might be required to use verifiable credentials. Some configurations might require correlatable identifiers for subjects. Example verifiable data registries include trusted databases, decentralized databases, government ID databases, and distributed ledgers. Often there is more than one type of verifiable data registry utilized in an ecosystem.

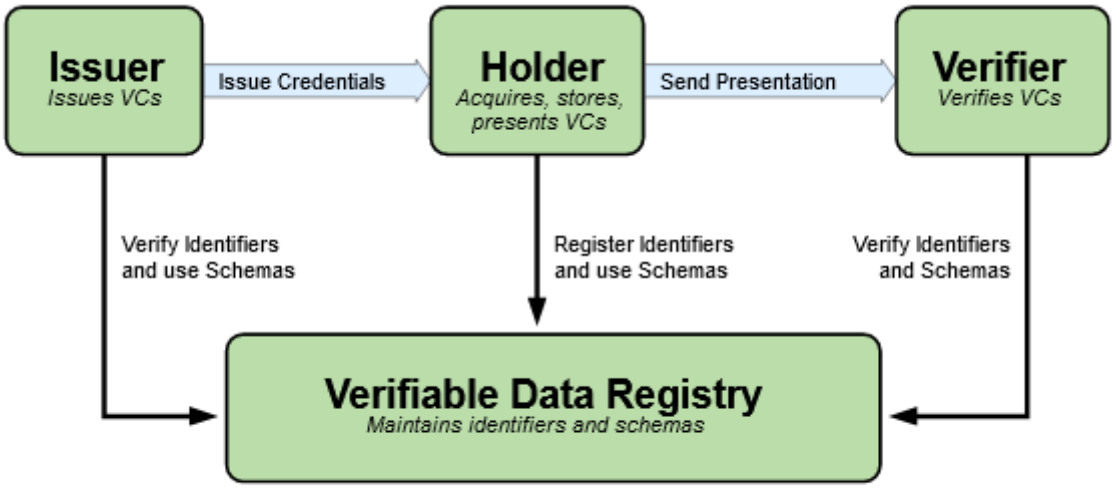


Figure 1 The roles and information flows forming the basis for this specification.

decentralized identifier

A portable URL-based identifier, also known as a ***DID***, associated with an entity. These identifiers are most often used in a verifiable credential and are associated with subjects such that a verifiable credential itself can be easily ported from one repository to another without the need to reissue the credential. An example of a DID is `did:example:123456abcdef`.

decentralized identifier document

Also referred to as a ***DID document***, this is a document that is accessible using a verifiable data registry and contains information related to a specific decentralized identifier, such as the associated repository and public key information.

Source of images above: https://www.w3.org/TR/vc-data-model/#ecosystem-overview

- Verification should not depend on direct interactions between issuers and verifiers.
- Verification should not reveal the identity of the verifier to any issuer.
- Revocation by the issuer should not reveal any identifying information about the subject, the holder, the specific verifiable credential, or the verifier.
- Issuers revoking verifiable credentials should distinguish between revocation for cryptographic integrity (for example, the signing key is compromised) versus revocation for a status change (for example, the driver's license is suspended).
- Issuers can provide a service for refreshing a verifiable credential.

§ 3.2 Credentials

This section is non-normative.

A credential is a set of one or more claims made by the same entity. Credentials might also include an identifier and metadata to describe properties of the credential, such as the issuer, the expiry date and time, a representative image, a public key to use for verification purposes, the revocation mechanism, and so on. The metadata might be signed by the issuer. A verifiable credential is a set of tamper-evident claims and metadata that cryptographically prove who issued it.

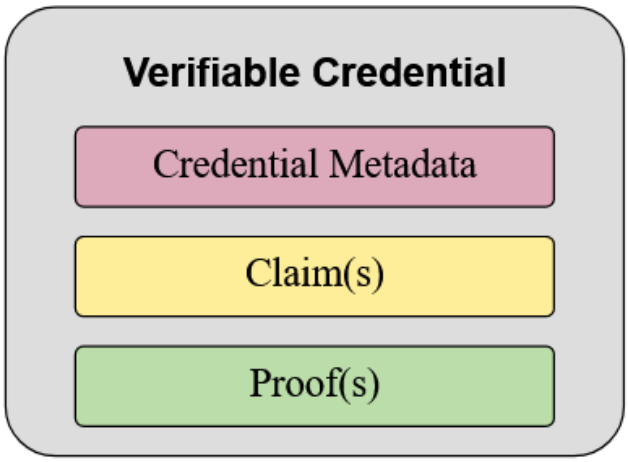


Figure 5 Basic components of a verifiable credential.

Examples of verifiable credentials include digital employee identification cards, digital birth certificates, and digital educational certificates.

> NOTE
>
> Credential identifiers are often used to identify specific instances of a credential. These identifiers can also be used for correlation. A holder wanting to minimize correlation is advised to use a selective disclosure scheme that does not reveal the credential identifier.

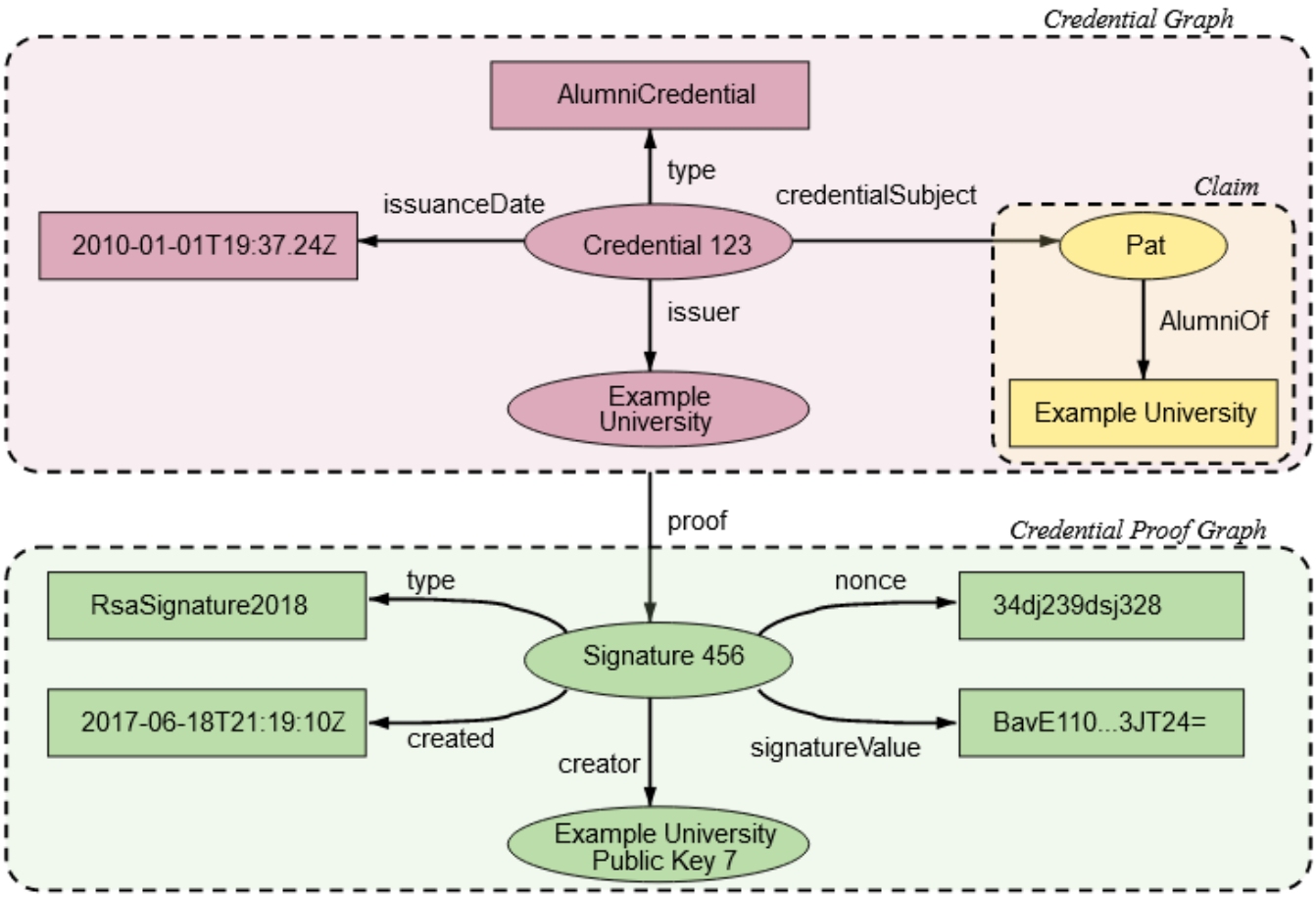


Figure 6 Information graphs associated with a basic verifiable credential.

Source: https://www.w3.org/TR/vc-data-model/#credentials

§ 3.3 Presentations

This section is non-normative.

Enhancing privacy is a key design feature of this specification. Therefore, it is important for entities using this technology to be able to express only the portions of their persona that are appropriate for a given situation. The expression of a subset of one's persona is called a verifiable presentation. Examples of different personas include a person's professional persona, their online gaming persona, their family persona, or an incognito persona.

A verifiable presentation expresses data from one or more verifiable credentials, and is packaged in such a way that the authorship of the data is verifiable. If verifiable credentials are presented directly, they become verifiable presentations. Data formats derived from verifiable credentials that are cryptographically verifiable, but do not of themselves contain verifiable credentials, might also be verifiable presentations.

The data in a presentation is often about the same subject, but might have been issued by multiple issuers. The aggregation of this information typically expresses an aspect of a person, organization, or entity.

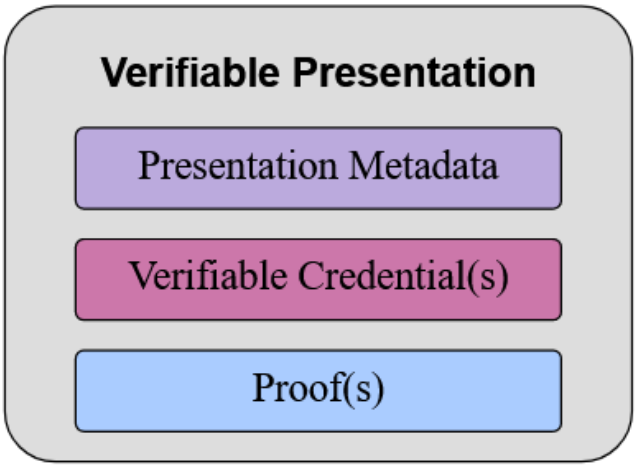


Figure 7 Basic components of a verifiable presentation.

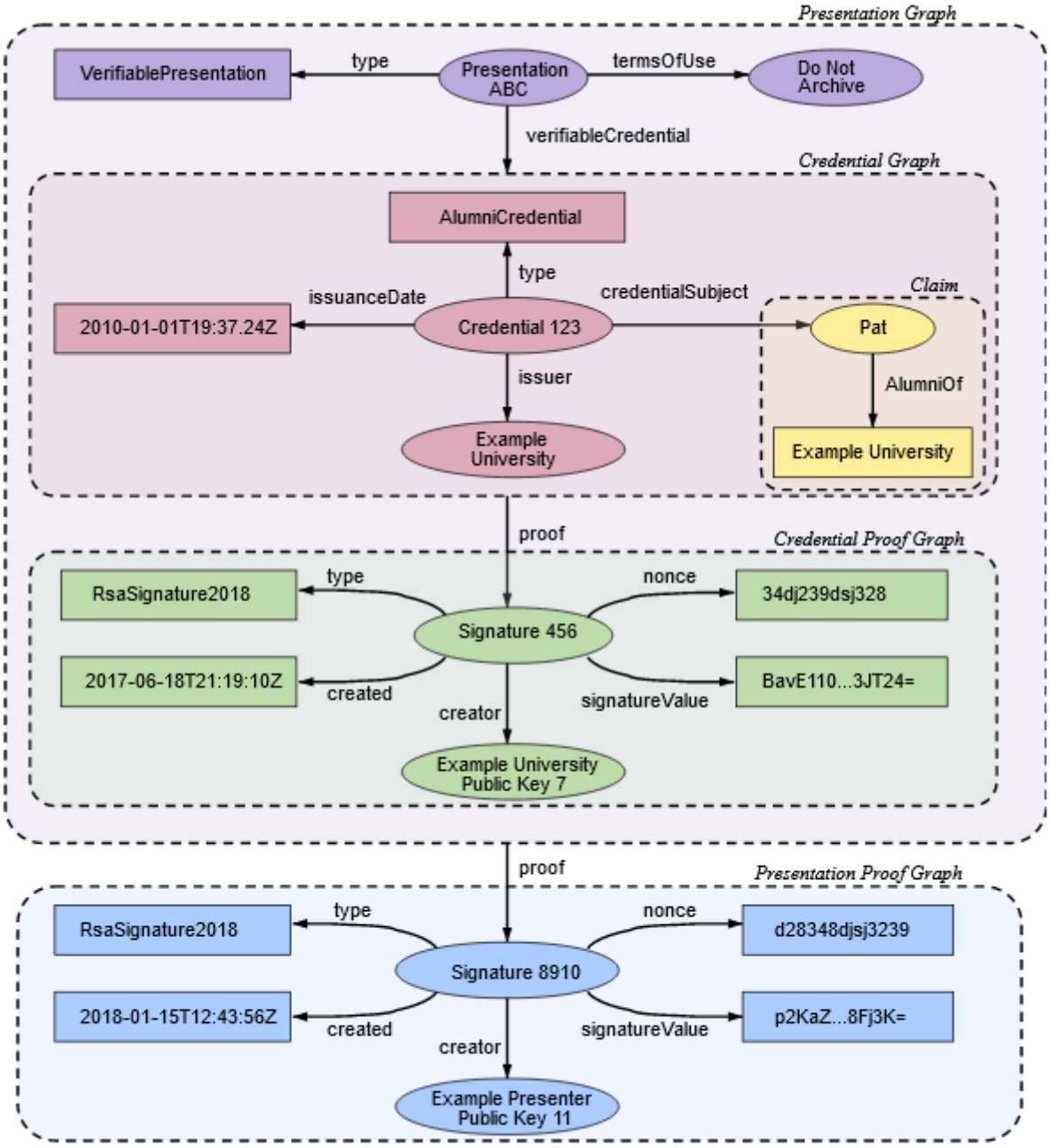


Figure 8 Information graphs associated with a basic verifiable presentation.

Source: https://www.w3.org/TR/vc-data-model/#presentations

A simple example of a verifiable credential:

```
// set the context, which establishes the special terms we will be using
// such as 'issuer' and 'alumniOf'.
"@context": [
"https://www.w3.org/2018/credentials/v1",
"https://www.w3.org/2018/credentials/examples/v1"
],
// specify the identifier for the credential
"id": "http://example.edu/credentials/1872",
// the credential types, which declare what data to expect in the credential
"type": ["VerifiableCredential", "AlumniCredential"],
```

```
// the entity that issued the credential
"issuer": "https://example.edu/issuers/565049",
// when the credential was issued
"issuanceDate": "2010-01-01T19:23:24Z",
// claims about the subjects of the credential
"credentialSubject": {
// identifier for the only subject of the credential
"id": "did:example:ebfeb1f712ebc6f1c276e12ec21",
// assertion about the only subject of the credential
"alumniOf": {
  "id": "did:example:c276e12ec21ebfeb1f712ebc6f1",
  "name": [{
    "value": "Example University",
    "lang": "en"
  }, {
    "value": "Exemple d'Université",
    "lang": "fr"
  }]
}
},
// digital proof that makes the credential tamper-evident
// see the NOTE at end of this section for more detail
"proof": {
// the cryptographic signature suite that was used to generate the signature
"type": "RsaSignature2018",
// the date the signature was created
"created": "2017-06-18T21:19:10Z",
// purpose of this proof
"proofPurpose": "assertionMethod",
// the identifier of the public key that can verify the signature
"verificationMethod": "https://example.edu/issuers/565049#key-1",
// the digital signature value
"jws": "eyJhbGciOiJSUzI1NiIsImI2NCI6ZmFsc2UsImNyaXQiOlsiYjY0Il19..TCYt5X
  sITJX1CxPCT8yAV-TVkIEq_PbChOMqsLfRoPsnsgw5WEuts01mq-pQy7UJiN5mgRxD-WUc
  X16dUEMGlv50aqzpqh4Qktb3rk-BuQy72IFLOqV0G_zS245-kronKb78cPN25DGlcTwLtj
  PAYuNzVBAh4vGHSrQyHUdBBPM"
}
```

A simple example of a verifiable presentation:

```
{
"@context": [
"https://www.w3.org/2018/credentials/v1",
"https://www.w3.org/2018/credentials/examples/v1"
```

```
],
"type": "VerifiablePresentation",
// the verifiable credential issued in the previous example
"verifiableCredential": [{
"@context": [
  "https://www.w3.org/2018/credentials/v1",
  "https://www.w3.org/2018/credentials/examples/v1"
],
"id": "http://example.edu/credentials/1872",
"type": ["VerifiableCredential", "AlumniCredential"],
"issuer": "https://example.edu/issuers/565049",
"issuanceDate": "2010-01-01T19:23:24Z",
"credentialSubject": {
  "id": "did:example:ebfeb1f712ebc6f1c276e12ec21",
  "alumniOf": {
    "id": "did:example:c276e12ec21ebfeb1f712ebc6f1",
    "name": [{
      "value": "Example University",
      "lang": "en"
    }, {
      "value": "Exemple d'Université",
      "lang": "fr"
    }]
  }
},
"proof": {
  "type": "RsaSignature2018",
  "created": "2017-06-18T21:19:10Z",
  "proofPurpose": "assertionMethod",
  "verificationMethod": "https://example.edu/issuers/565049#key-1",
  "jws": "eyJhbGciOiJSUzI1NiIsImI2NCI6ZmFsc2UsImNyaXQiOlsiYjY0Il19..TCYt5X
    sITJX1CxPCT8yAV-TVkIEq_PbChOMqsLfRoPsnsgw5WEuts01mq-pQy7UJiN5mgRxD-WUc
    X16dUEMGlv50aqzpqh4Qktb3rk-BuQy72IFLOqV0G_zS245-kronKb78cPN25DGlcTwLtj
    PAYuNzVBAh4vGHSrQyHUdBBPM"
}
}],
// digital signature by Pat on the presentation
// protects against replay attacks
"proof": {
"type": "RsaSignature2018",
"created": "2018-09-14T21:19:10Z",
"proofPurpose": "authentication",
"verificationMethod": "did:example:ebfeb1f712ebc6f1c276e12ec21#keys-1",
```

```
// 'challenge' and 'domain' protect against replay attacks
"challenge": "1f44d55f-f161-4938-a659-f8026467f126",
"domain": "4jt78h47fh47",
"jws": "eyJhbGciOiJSUzI1NiIsImI2NCI6ZmFsc2UsImNyaXQiOlsiYjY0Il19..kTCYt5
  XsITJX1CxPCT8yAV-TVIw5WEuts01mq-pQy7UJiN5mgREEMGlv50aqzpqh4Qq_PbChOMqs
  LfRoPsnsgxD-WUcX16dUOqV0G_zS245-kronKb78cPktb3rk-BuQy72IFLN25DYuNzVBAh
  4vGHSrQyHUGlcTwLtjPAnKb78"
}
```

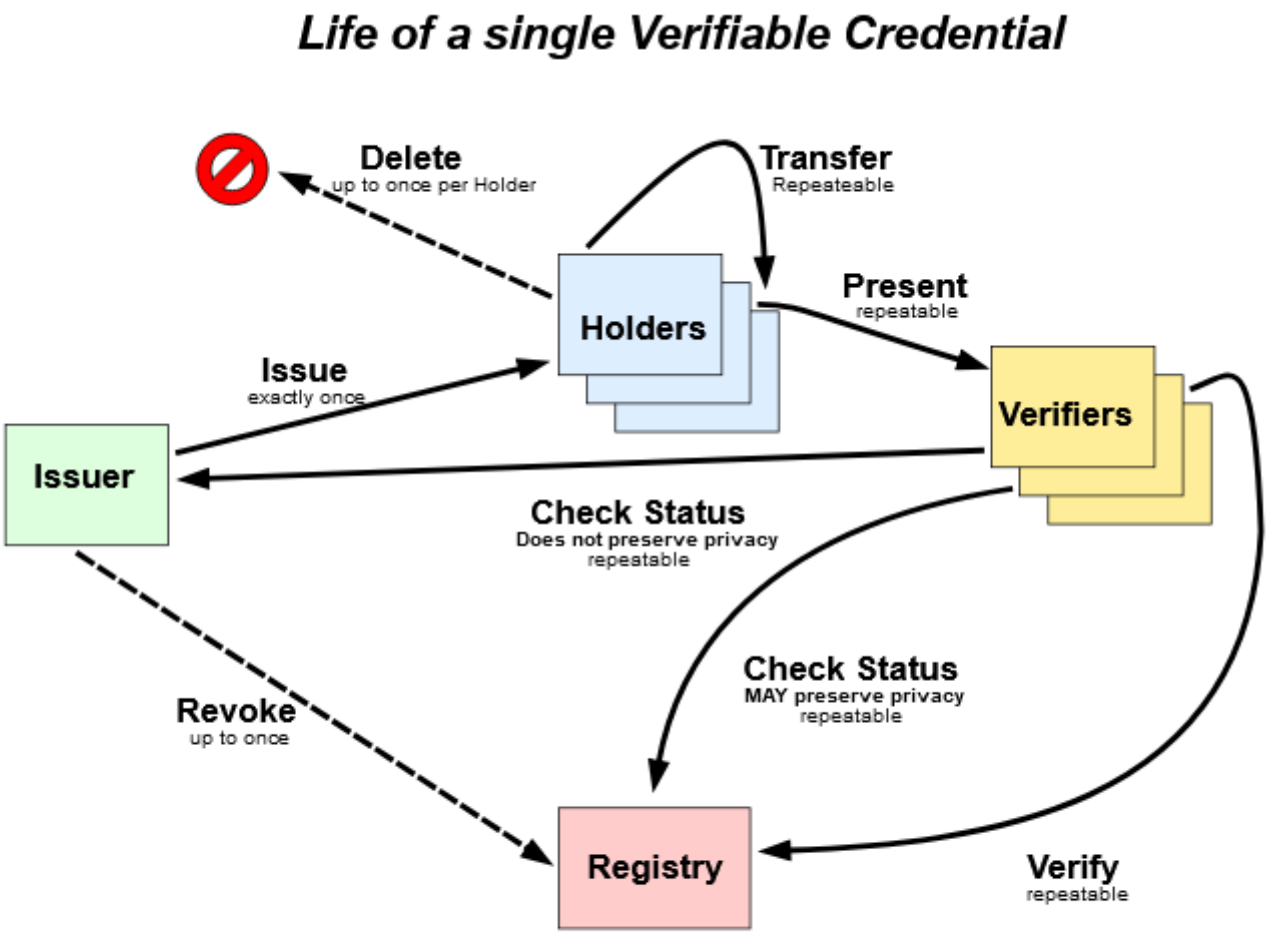


Figure 10 The roles and information flows for this specification.

Source: https://www.w3.org/TR/vc-data-model/#lifecycle-details

Networks

- The networks are different distributed blockchains with verifiable data registries using DIDs.
- You need to know how to resolve each DID, issuer DID to verify or use a credential and so you need to know where to find the network on which the DID is persisted.
- **The networks are really just persisted distributed databases.**
- The blockchain holds public key DIDs, DID documents, ie credentials and schemas.

JWT and JWS considerations regarding Credentials syntaxes

> Ref in spec: https://www.w3.org/TR/vc-data-model/#json-web-token

Statements

- If a JWS is present, the digital signature refers either to the issuer of the verifiable credential, or in the case of a verifiable presentation, to the holder of the verifiable credential. The JWS proves that the `iss` of the JWT signed the contained JWT payload and therefore, the `proof` property can be omitted.
- If no JWS is present, a proof property MUST be provided. The `proof` property can be used to represent a more complex proof, as may be necessary if the creator is different from the issuer, or a proof not based on digital signatures, such as Proof of Work. The issuer MAY include both a JWS and a `proof` property. For backward compatibility reasons, the issuer MUST use JWS to represent proofs based on a digital signature.

Claims

> Other JOSE header parameters and JWT claim names not specified herein can be used if their use is not explicitly discouraged. Additional verifiable credential claims MUST be added to the credentialSubject property of the JWT.

- `vc`: JSON object, which MUST be present in a JWT verifiable credential. The object contains the credential according to this specification.
- `vp`: JSON object, which MUST be present in a JWT verifiable presentation. The object contains the presentation according to this specification.
- `alg`: MUST be set for digital signatures. If only the `proof` property is needed for the chosen signature method (that is, if there is no choice of algorithm within that method), the `alg` header MUST be set to `none`.
- `kid`: MAY be used if there are multiple keys associated with the issuer of the JWT. The key discovery is out of the scope of this specification. For example, the `kid` can refer to a key in a DID document, or can be the identifier of a key inside a JWKS.
- `typ`: if present, MUST be set to **JWT**.
- `exp`: MUST represent the **expirationDate** property, encoded as a UNIX timestamp (NumericDate).
- `iss`: MUST represent the issuer property of a verifiable credential or the holder property of a verifiable presentation.
- `nbf`: MUST represent **issuanceDate**, encoded as a UNIX timestamp (NumericDate).
- `jti`: MUST represent the **id** property of the verifiable credential or verifiable presentation.
- `sub`: MUST represent the **id** property contained in the **credentialSubject**.
- `aud`: MUST represent (i.e., identify) the intended audience of the verifiable presentation (i.e., the verifier intended by the presenting holder to receive and verify the verifiable presentation).

JWT **header** of a JWT based verifiable presentation:

```
{
  "alg": "RS256",
  "typ": "JWT",
  "kid": "did:example:ebfeb1f712ebc6f1c276e12ec21#keys-1"
}
```

In the example above, the verifiable presentation uses a proof based on JWS digital signatures, and the corresponding verification key can be obtained using the **kid** header parameter.

JWT **payload** of a JWT-based verifiable credential using JWS as a proof:

```
{
  "sub": "did:example:ebfeb1f712ebc6f1c276e12ec21",
  "jti": "http://example.edu/credentials/3732",
  "iss": "https://example.com/keys/foo.jwk",
  "nbf": 1541493724,
  "iat": 1541493724,
  "exp": 1573029723,
  "nonce": "660!6345FSer",
  "vc": {
    "@context": [
      "https://www.w3.org/2018/credentials/v1",
      "https://www.w3.org/2018/credentials/examples/v1"
    ],
    "type": ["VerifiableCredential", "UniversityDegreeCredential"],
    "credentialSubject": {
      "degree": {
        "type": "BachelorDegree",
        "name": "Bachelor of Science and Arts"
      }
    }
  }
}
```

In the example above, **vc** does not contain the **id** property because the JWT encoding uses the **jti** attribute to represent a unique identifier. The **sub** attribute encodes the information represented by the **id** property of **credentialSubject**. The **nonce** has been added to stop a replay attack.

Other examples can be found here.