Bachelor Thesis

Unlinkability of Verifiable Credentials in a practical approach

April 23, 2024

Joel Robles | TI

Table of Contents

- **Goal**
- **Self-sovereign Identity**
- **Verifiable Credentials**
- **Verifiable Presentations**
- **Security Considerations**
- **Outlook**

Goal

What is the goal?

Check for unlinkability of Verifiable Credentials in conjuction with the BBS Signature Scheme in Real-World implementations

Self-sovereign Identity

Self-sovereign Identity (SSI)

- Is a concept, where a Person, also known as a **Holder**, decides who gets to know what about them
- Holders cannot choose what to disclose and what not, also known as **selective disclosure**
- First problem:
  - Holder shows Government ID
  - Is a set of data/ set of **attributes**
  - The person verifying sees all the attributes
- Second problem:
  - Holder shows attributes to someone who wants to verify, known as a **verifier**
  - Then shows the same attributes to a second verifier
  - The holder then can be **linked**
- Today's state - Holders have no control over their attributes
- Tomorrow's state thanks to SSI - Holders have full control over their attributes

Trust Triangle

- How does the verifier know that the set of attributes (**credential**), is valid?
- He trusts the issuer!
- Example: Swiss government ID has holograms

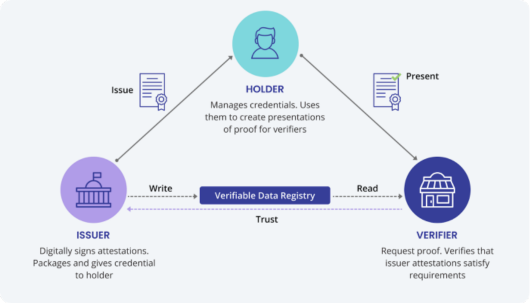
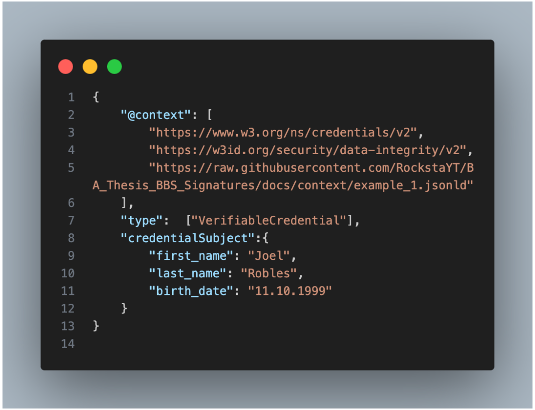

HOLDER
Manages credentials. Uses them to create presentations of proof for verifiers
Issue
Present
ISSUER
Digitally signs attestations. Packages and gives credential to holder
Write
Verifiable Data Registry
Read
Trust
VERIFIER
Request proof. Verifies that issuer attestations satisfy requirements

Figure: Trust triangle

Verifiable Credentials

Verifiable Credentials (VC)

- Verifiable Credentials are digital representations of physical credentials
- JSON-LD represent attributes as **key-value pairs**
- Example:
  - First name on Government ID
  - Represented as {"first_name": "Joel"}
  - "first_name" is the key and "Joel" is the value



```
{
    "@context": [
        "https://www.w3.org/ns/credentials/v2",
        "https://w3id.org/security/data-integrity/v2",
        "https://raw.githubusercontent.com/RockstaYT/BA_Thesis_BBS_Signatures/docs/context/example_1.jsonld"
    ],
    "type":  ["VerifiableCredential"],
    "credentialSubject":{
        "first_name": "Joel",
        "last_name": "Robles",
        "birth_date": "11.10.1999"
    }
}
```

Figure: Example VC

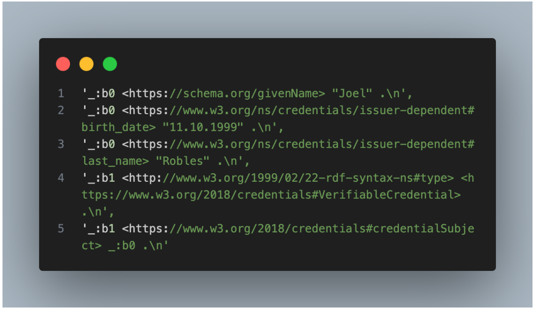
VCs and BBS

- Why are they called **Verifiable** Credentials?
- The verifier is able to verify a VC that was presented to him (**Verifiable Presentation**), because of cryptographic signatures
- These show that a credential has not been altered since issuance
- We use the BBS Signature Scheme (**BBS**)
- This scheme provides **selective disclosure** and **unlinkability**
- How unlinkability? - Verifier needs the signature
- BBS can generate **proofs**
- These proof that a holder knows the signature, without revealing it
- These are also unlinkable between each generation

Verifiable Presentations

Verifiable Presentation (VP)

- A holder would like to present a VC
- For that, Verifiable Presentations are used
- BBS can only sign statements
- The **RDF** canonicalization algorithm creates statements out of key-value pairs



```
'_:b0 <https://schema.org/givenName> "Joel" .\n',
'_:b0 <https://www.w3.org/ns/credentials/issuer-dependent#birth_date> "11.10.1999" .\n',
'_:b0 <https://www.w3.org/ns/credentials/issuer-dependent#last_name> "Robles" .\n',
'_:b1 <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <https://www.w3.org/2018/credentials#VerifiableCredential> .\n',
'_:b1 <https://www.w3.org/2018/credentials#credentialSubject> _:b0 .\n'
```

Figure: Example Canonicalized VP

Security Considerations

Shuffling of statements

- The verifier now gets a VP from the holder, which can be verified against the public key of the issuer
- The holder gets a new version of a credential, where an attribute has been changed
- In a very specific case, the RDF canonicalization algorithm can lead to data leakage
- Each time this algorithm is used, the canonical statements must be shuffled using a hash function

Outlook

Outlook

- Analyze OpenID Connect for Verifiable Presentations (**OIDC4VP**) for unlinkability and data leakage
- Analyze Link Secrets and Blind Signatures for unlinkability and data leakage