Decentralized Identity + Interoperability Workshop:
Connecting Credo (Aries Framework JavaScript) with Hyperledger Besu, Cardano, Cheqd, Hyperledger AnonCreds and OID4VC

Alexander Shcherbakov
Renata Toktar
Artem Ivanov

DSR Corporation
Goals

• SSI is not a framework/tool/library
• SSI is a concept/model for digital identity
• There are multiple specifications/protocols in SSI
• There are multiple frameworks/tools/libraries implementing SSI principles and protocols

• Goal 1: Summarize main approaches/specifications/profiles in SSI
• Goal 2: Show interoperability between some approaches, tools, frameworks
Agenda

1. About Self-sovereign Identity (SSI)
2. Interoperability Variables (Profiles)
   • VC Formats
   • VC Exchange Protocols
   • DID method / Verifiable Data Registry (VDR)
3. Interoperability Variables Values for the Demo Part
   • Credo (Aries Framework JavaScript)
   • VC Formats: Hyperledger AnonCreds, W3C VC
   • VC Exchange Protocols: Hyperledger Aries, OID4VC
   • VDR: Cardano, Cheqd, Hyperledger Besu
4. Demo
5. Hands-On
About the Speakers

Alexander Shcherbakov

Head of Decentralized Systems Department
DSR Corporation

• Ph.D. degree in Mathematics
• 13+ years of experience in software engineering and team management
• 7+ years of experience in Digital ID, Self-Sovereign Identity (SSI), Blockchain, Distributed Ledger Technology (DLT), Consensus protocols, and cryptography.
• One of the core maintainers and contributors of such projects as Hyperledger Indy, Hyperledger Aries, Hyperledger Ursa, etc.
• International conference speaker: Hyperledger Global Forum, Hyperledger Webinars/Workshops, IIW, CSA member meeting, etc.
About the Speakers

Renata Toktar

Lead Software Engineer
DSR Corporation

• Master degree in Mathematics

• 9+ years of experience in software engineering

• 6 years of experience in Digital ID, Self-Sovereign Identity (SSI), Blockchain, Distributed Ledger Technology (DLT), Consensus protocols, and cryptography.

• One of the core maintainers and contributors of such projects as Hyperledger Indy and Cheqd

• Used to lead blockchain startup

• International conference speaker on Hyperledger Global Forum
About DSR

- Over 7 years of experience in Blockchain and Self-sovereign Identity
- Contributed to more than 50 open source projects
- One of the main contributors of:
  - Hyperledger Indy
  - Hyperledger Aries
  - Hyperledger Ursa
  - Sovrin
  - Hyperledger Anoncreds
  - Cheqd
- A member of:
  - Hyperledger Foundation
  - DIF
  - OpenWallet Foundation
  - Trust Over IP Foundation
  - The Linux Foundation
Digital Identity Models

#1: Siloed (Centralized) Identity

#2: Third-Party IDP (Federated) Identity

#3 Self-Sovereign Identity
Non-SSI vs SSI
SSI Concepts

1. Verifiable Credentials (VC)
Verifiable Credentials Data Model v1.1, W3C Recommendation 2022

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

2. Decentralized Identifiers (DID)
Decentralized Identifiers (DIDs) v1.0, W3C Recommendation 2022

From https://www.w3.org/TR/did-core/
SSI Concepts: Verifiable Credential

- A credential is a set of one or more claims made by an issuer. Typically the claims describe some properties of the credential holder.

- A verifiable credential is a tamper-evident credential that has authorship that can be cryptographically verified. Verifiable credentials can be used to build verifiable presentations, which can also be cryptographically verified. The claims in a credential can be about different subjects.

Verifiable Credentials Data Model v1.1, W3C Recommendation 2022
SSI Concepts: DID

- A Decentralized Identifier (DID) refers to any subject (e.g., a person, organization, thing, data model, abstract entity, etc.).

- In contrast to typical, federated identifiers, DIDs may be decoupled from centralized registries, identity providers, and certificate authorities.

Decentralized Identifiers (DIDs) v1.0, W3C Recommendation 2022

From https://www.w3.org/TR/did-core/
SSI Concepts: DID and DID DOC

```json
{
  "@context": ["https://www.w3.org/ns/did/v1", "https://identity.foundation/.well-known/did-ld-context.json"],
  "id": "did:example:123",
  "verificationMethod": [{
    "id": "did:example:123#verification",
    "type": "JsonWebKey2020",
    "controller": "did:example:123",
    "publicKeyJwk": {
      "kty": "OKP",
      "crv": "Ed25519",
      "x": "VCpo2LMLhnh6jWku8MKvS9g2ZAoC-nL0yPVq03FxYe0"
    }
  }],
  "service": [
    {
      "id": "did:example:123#foo",
      "type": "LinkedDomains",
      "serviceEndpoint": {
        "origins": ["https://foo.example.com", "https://identity.foundation"]
      }
    },
    {
      "id": "did:example:123#bar",
      "type": "LinkedDomains",
      "serviceEndpoint": "https://bar.example.com"
    }
  ]
}
```

From https://www.w3.org/TR/did-core/
SSI Concepts In Action

- **Issuer’s DID and DIDDoc (Public Key)**
  - Sign VC
  - Public, usually on a Blockchain or a trusted Web service (did:web, did:indy, did:cheqd, etc.)

- **Holder’s DID**
  - Associates a VC with a DID
  - Prove ownership (signature) of the Holder during presentation
  - Either Public or Private (did:key, did:peer, long form of did:ion, etc.)

By Daniel Hardman licenced under CC BY-SA 4.0
**Trust Over IP Stack**

<table>
<thead>
<tr>
<th>Layer 1</th>
<th>Layer 2</th>
<th>Layer 3</th>
<th>Layer 4</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Public Utilities</strong></td>
<td><strong>DID method 1</strong></td>
<td><strong>Example: Credential Exchange</strong></td>
<td><strong>Trust Task Protocols</strong></td>
</tr>
<tr>
<td>DID method 1</td>
<td>Interoperable</td>
<td>Issuer</td>
<td>Holder</td>
</tr>
<tr>
<td>DID method 2</td>
<td>Utility 1</td>
<td></td>
<td>Trust</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Verifier</td>
</tr>
<tr>
<td><strong>Utility Governance Frameworks</strong></td>
<td><strong>Agent/Wallet Governance Frameworks</strong></td>
<td><strong>Ecosystem Governance Frameworks</strong></td>
<td><strong>ToIP Technology Stack</strong></td>
</tr>
<tr>
<td>Governing Authority</td>
<td>Agent/Wallet Framework</td>
<td>Governing Authority</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Credential Framework</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Ecosystem Framework</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Governa/Certifies</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Ecosystem Roles</td>
<td></td>
</tr>
<tr>
<td><strong>Agent/Wallet</strong></td>
<td><strong>Connection</strong></td>
<td><strong>Issuer</strong></td>
<td><strong>DID</strong></td>
</tr>
<tr>
<td>Agent/Wallet</td>
<td></td>
<td>Holder</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Utility 1</strong></td>
<td><strong>Interoperable</strong></td>
<td><strong>DID</strong></td>
<td><strong>Layer 3:</strong></td>
</tr>
<tr>
<td>Utility 2</td>
<td></td>
<td></td>
<td>- Verifiable Credential issued and signed by the Issuer</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Verifiable Presentation created by the Holder and sent to the Verifier</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- The exact mechanism (protocol) how to exchange credentials and presentations</td>
</tr>
</tbody>
</table>

**Layer 2:**
- Secure “connection” to share the credentials and presentations (such as DIDComm)

**Layer 1:**
- Issue/Sign: Issuer puts an Identifier (DID) and associated Public Key and Metadata (DIDDoc) on the VDR (such as Blockchain).
- Hold/Present: Holder may put an Identifier (DID) to be associated with a credential on the VDR (such as Blockchain). Proof of possession.

*From https://trustoverip.org/toip-model/*
SSI and Blockchain

- Blockchain is optional in SSI cases
- Blockchain may appear on Layer 1 only as one of the options for DID's Verifiable Data Registry (VDR)

What is **usually** stored on the Blockchain:
- Issuer’s DID/DIDDoc and Public Keys
- Revocation registries
- Credential Schemas

What **may be** stored on the Blockchain:
- Holder’s DID/DIDDoc and Public Keys

What is **never** stored on the Blockchain:
- Verifiable Credentials
- Private keys

From https://trustoverip.org/toip-model/
Interoperability Variables (Profiles)
Interoperability Variables

1. **What: VC Format**
   - Hyperledger AnonCreds
   - W3C VC (JSON-LD, JWT, SD-JWT)
   - ISO mDL
   - etc.

2. **How: VC Exchange Protocols**
   - Hyperledger Aries (DIDComm based)
   - DIF WACI (DIDComm based)
   - OID4VC
   - W3C CHAPI
   - W3C VC API
   - ISO mDL
   - etc.

3. **Where: DID methods / VDR**
   - Hyperledger Indy Ledger (did:indy, did:sov)
   - Cheqd blockchain (did:cheqd)
   - Cardano blockchain (did:prism)
   - Hyperledger Besu, Indy-Besu (did:indy, did:indy2)
   - Self-resolving (did:key)
   - DNS (did:web)
   - etc.

*The combination is sometimes called a Profile*
*More variables can be defined (such as Revocation, etc.)*

From https://trustoverip.org/toip-model/
Interoperability Variables: More than Indy

1. What: VC Format
   - Hyperledger AnonCreds
   - W3C VC (JSON-LD, JWT, SD-JWT)
   - ISO mDL
   - etc.

2. How: VC Exchange Protocols
   - Hyperledger Aries (DIDComm based)
   - DIF WACI (DIDComm based)
   - OID4VC
   - W3C CHAPI
   - W3C VC API
   - ISO mDL
   - etc.

3. Where: DID methods / VDR
   - Hyperledger Indy Ledger (did:indy, did:sov)
   - Cheqd blockchain (did:cheqd)
   - Cardano blockchain (did:prism)
   - Hyperledger Besu, Indy-Besu (did:indy, did:indy2)
   - Self-resolving (did:key)
   - DNS (did:web)
   - etc.

Traditional / First Versions
Interoperability Variables: More than Indy

1. What: VC Format
   - Hyperledger AnonCreds
   - W3C VC (JSON-LD, JWT, SD-JWT)
   - ISO mDL
   - etc.

2. How: VC Exchange Protocols
   - Hyperledger Aries (DIDComm based)
   - DIF WACI (DIDComm based)
   - OID4VC
   - W3C CHAPI
   - W3C VC API
   - ISO mDL
   - etc.

3. Where: DID methods / VDR
   - Hyperledger Indy Ledger (did:indy, did:sov)
   - Cheqd blockchain (did:cheqd)
   - Cardano blockchain (did:prism)
   - Hyperledger Besu, Indy-Besu (did:indy, did:indy2)
   - Self-resolving (did:key)
   - DNS (did:web)
   - etc.

There are much more options / profiles
Interoperability Variables: More than Indy

1. What: VC Format
   - Hyperledger AnonCreds
   - W3C VC (JSON-LD, JWT, SD-JWT)
   - ISO mDL
   - etc.

2. How: VC Exchange Protocols
   - Hyperledger Aries (DIDComm based)
   - DIF WACI (DIDComm based)
   - OID4VC
   - W3C CHAPI
   - W3C VC API
   - ISO mDL
   - etc.

3. Where: DID methods / VDR
   - Hyperledger Indy Ledger (did:indy, did:sov)
   - Cheqd blockchain (did:cheqd)
   - Cardano blockchain (did:prism)
   - Hyperledger Besu, Indy-Basu (did:indy, did:indy2)
   - Self-resolving (did:key)
   - DNS (did:web)
   - etc.

There are much more options / profiles
What: Verifiable Credential Formats
Selective Disclosure and Predicates

Holder
- Name: John Smith
- Nickname: Johnny
- Address: Bangkok
- Date of Birth: 24/11/91
- ID Number: 3638291

Verifier
- Nickname: Johnny
- Predicate: Age ≥ 20
## What: Verifiable Credential Formats

<table>
<thead>
<tr>
<th>VC Format</th>
<th>Standard</th>
<th>Selective Disclosure</th>
<th>Predicates (Ex.: proof over 18)</th>
<th>Serialization</th>
<th>Proof Format, Signing Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>W3C VC / JSON-LD</td>
<td></td>
<td>No</td>
<td>No</td>
<td>JSON-LD</td>
<td>Data Integrity Proofs, ECDSA, RSA, EdDSA (Ed25519)</td>
</tr>
<tr>
<td>W3C VC / JSON-LD, BBS+</td>
<td>W3C VC</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td>Data Integrity Proofs, BBS+</td>
</tr>
<tr>
<td>W3C VC / JWT</td>
<td></td>
<td>No</td>
<td>No</td>
<td>JSON</td>
<td>JWT/JWS: ECDSA, RSA, EdDSA (Ed25519)</td>
</tr>
<tr>
<td>W3C VC / SD-JWT</td>
<td></td>
<td>Yes</td>
<td>No</td>
<td>JSON</td>
<td>JWT/JWS: ECDSA, RSA, EdDSA (Ed25519)</td>
</tr>
<tr>
<td>Hyperledger AnonCreds (Indy)</td>
<td>Hyperledger AnonCreds</td>
<td>Yes</td>
<td>Yes</td>
<td>JSON</td>
<td>CL AnonCreds (ZKP)</td>
</tr>
<tr>
<td>ISO mDL</td>
<td>ISO 18013-5</td>
<td>Yes</td>
<td>No</td>
<td>CBOR</td>
<td>COSE (ECDSA)</td>
</tr>
</tbody>
</table>

Selective disclosure JWT (SD-JWT)
### How: Verifiable Credential Exchange Protocols

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Specification</th>
<th>VC Format</th>
<th>Transport</th>
<th>Offline/Online</th>
</tr>
</thead>
<tbody>
<tr>
<td>OpenID for VC</td>
<td>• OpenID for Verifiable Credential Issuance (OID4VCI)</td>
<td>Any</td>
<td>HTTP(s), Bluetooth (work in progress)</td>
<td>Online, work in progress for offline</td>
</tr>
<tr>
<td></td>
<td>• OpenID for Verifiable Presentations (OID4VP)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Self-Issued OpenID Provider v2 (SIOPv2)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DIDComm based</td>
<td>• Hyperledger Aries</td>
<td>Any, but usually Hyperledger AnonCreds</td>
<td>Any</td>
<td>Offline and online</td>
</tr>
<tr>
<td></td>
<td>• DIF WACI</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ISO mDL Protocols</td>
<td>• ISO/IEC WD TS 23220-3</td>
<td>ISO mDL</td>
<td>NFC, Bluetooth, WiFi, HTTP(s)</td>
<td>Offline and online</td>
</tr>
<tr>
<td></td>
<td>• ISO/IEC CD TS 18013-7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>W3C Credential Handler API (Browser specific)</td>
<td>W3C Credential Handler API (CHAPI)</td>
<td>W3C VC</td>
<td>Browser API, HTTP(s)</td>
<td>Online</td>
</tr>
<tr>
<td>W3C Verifiable Credentials API (REST)</td>
<td>W3C VC API</td>
<td>W3C VC</td>
<td>HTTP(s)</td>
<td>Online</td>
</tr>
</tbody>
</table>
Where: DID methods / VDR

https://www.w3.org/TR/did-spec-registries/#did-methods

- **did:key** - Self-resolving and ledger-independent
- **did:peer** - Ledger-independent; partially self-resolving
- **did:web** - Web domain's existing reputation; resolved through DNS (no Ledger)
- **did:webs** - Web based; KERI instead of DNS for trust
- **did:keri** - Ledger agnostic VDR
- **did:ion** - resolved through blockchain-agnostic Sidetree protocol on top of Bitcoin; self-resolving option
- **did:indy / did:sov** – Hyperledger Indy Ledger as VDR
- **did:ethr** - Ethereum as VDR
- **did:cheqd** - Cosmos-sdk based cheqd ledger as VDR
- etc.
Existing Profiles\(^1\)(\(^2\))

https://openwallet-foundation.github.io/credential-format-comparison-sig

- **HAIP** - High Assurance Interoperability Profile (OID4VC, SD-JWT-VC, raw keys)
- **DIIP** - Decentralized Identity Interop Profile (OID4VC, JWT-VC, did:web/did:jwk)
- **ISO mDL**
- **Hyperledger Indy AnonCreds** – Hyperledger AnonCreds, Hyperledger Aries, Hyperledger Indy (did:indy, did:sov)

(1) *The notion of a 'Profile' is not fully established yet and may vary across organizations and working groups*

(2) *Not all profiles have formal specification*
Interoperability Variables Values for the Demo Part
## Demo Scenarios: Interoperability Variables

<table>
<thead>
<tr>
<th>Demo Part</th>
<th>VC Format</th>
<th>VC Exchange Protocol</th>
<th>VDR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. AnonCreds + Cardano</td>
<td>Hyperledger AnonCreds</td>
<td>Hyperledger Aries</td>
<td>Cardano (as AnonCreds VDR) + did:key</td>
</tr>
<tr>
<td>2. AnonCreds + cheqd</td>
<td>Hyperledger AnonCreds</td>
<td>Hyperledger Aries</td>
<td>Cheqd (as AnonCreds VDR + did:cheqd)</td>
</tr>
<tr>
<td>3. W3C VC + cheqd</td>
<td>W3C VC (JSON-LD / Ed25519)</td>
<td>Hyperledger Aries</td>
<td>Cheqd (did:cheqd)</td>
</tr>
<tr>
<td>4. W3C VC + OID4VC</td>
<td>W3C VC (JWT / Ed25519)</td>
<td>OID4VC</td>
<td>did:key</td>
</tr>
<tr>
<td>Hands-on Part</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5. AnonCreds + Besu</td>
<td>Hyperledger AnonCreds</td>
<td>Hyperledger Aries</td>
<td>Hyperledger Besu (Indy-Besu)</td>
</tr>
<tr>
<td>6. W3C VC + Besu</td>
<td>W3C VC (JSON-LD / Ed25519)</td>
<td>Hyperledger Aries</td>
<td>Hyperledger Besu (Indy-Besu)</td>
</tr>
</tbody>
</table>
## Demo Scenarios: Frameworks

<table>
<thead>
<tr>
<th>Demo Part</th>
<th>VC Format</th>
<th>VC Exchange Protocol</th>
<th>VDR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. AnonCreds + Cardano</td>
<td></td>
<td></td>
<td>• Cardano (Test Net)</td>
</tr>
<tr>
<td>2. AnonCreds + cheqd</td>
<td></td>
<td></td>
<td>• AFJ extention <a href="https://github.com/roots-id/cardano-anoncreds">https://github.com/roots-id/cardano-anoncreds</a></td>
</tr>
<tr>
<td>3. W3C VC + cheqd</td>
<td>Credo (AFJ)</td>
<td>Credo (AFJ)</td>
<td>• Cheqd (Test Net)</td>
</tr>
<tr>
<td>Hands-on Part</td>
<td></td>
<td></td>
<td>Credo (AFJ) (did:key)</td>
</tr>
<tr>
<td>5. AnonCreds + Besu</td>
<td></td>
<td></td>
<td>• Hyperledger Besu (Indy-Besu)</td>
</tr>
<tr>
<td>6. W3C VC + Besu</td>
<td></td>
<td></td>
<td>• local setup (Docker)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• AFJ extention link</td>
</tr>
</tbody>
</table>
Credo (Aries Framework JavaScript) - 1

https://github.com/openwallet-foundation/credo-ts
(Ex. https://github.com/hyperledger/aries-framework-javascript)

- JavaScript / TypeScript
- BE Web Servers (Node.js) or Mobile Apps (React Native)
  - A basis for Aries Bifold
- SDK/API to be integrated into Web or Mobile apps
  - There are extensions providing REST API on top of it

- Used to be one of Hyperledger Aries projects since 2019
- Moved to Open Wallet Foundation at the end of 2023
- Renamed from Aries Framework JavaScript (AFJ) to Credo
Credo (Aries Framework JavaScript) - 2

https://github.com/openwallet-foundation/credo-ts
(Ex. https://github.com/hyperledger/aries-framework-javascript)

- Wallet:
  - Hyperledger Aries Ascar
  - Hyperledger Indy SDK

- VC Formats:
  - Hyperledger AnonCreds
  - W3C VC JSON-LD, W3C VC JWT,
  - W3C VC SD-JWT
  - W3C VC BBS+

- VC Exchange Protocols:
  - Aries v1/v2
  - OID4VC (WIP)

- DID methods / VDRs
  - did:sov, did:key, did:peer, did:cheqd
  - Extenentions (DID method, AnonCreds registries)
VC Format: Hyperledger AnonCreds

https://github.com/hyperledger/anoncreds-spec
https://github.com/hyperledger/anoncreds-rs

• The most privacy preserving VC Format
  • Predicates, selective disclosure, ZKP, anonymous revocation
• Main adoption historically – Hyperledger Indy, Hyperledger Aries
• Implemented and adopted before W3C VC standard was finalized
• Custom format (JSON), ZKP based signature (CL)
• Recent W3C VC Representation Support (link)
• Entities (published to a VDR, usually a ledger):
  • Credential Schema
  • Credential Definition (Issuer PubKey)
  • Revocation registry
2017: Hyperledger Indy, Sovrin Main Net
  • indy-crypto, indy-sdk, Indy Ledger

2018: Hyperledger Ursa
  • replaced indy-crypto

2019: Hyperledger Aries
  • used indy-sdk, Indy Ledger

2022: Hyperledger AnonCreds
  • AnonCreds spec, anoncreds-rs, ledger-agnostic AnonCreds
VC Format: W3C VC

https://www.w3.org/TR/vc-data-model/ (W3C Recommendation 2022)
https://www.w3.org/TR/vc-data-model-2.0 (W3C Working Draft)

- Serialization:
  - JSON
  - JSON-LD (Linked Data, Context, Semantic)

- Proof Format:
  - Data Integrity Proofs
  - JWT / SD-JWT (selective disclosure)

- Crypto/signatures:
  - ECDSA
  - RSA
  - EdDSA (Ed25519)
  - BBS+ (selective disclosure)
Verifiable Credential (VC)

```
{
  "@context": ["https://www.w3.org/2018/credentials/v1", .... ],
  "type": ["VerifiableCredential", "ComcastCredential"],
  // (1) ID of Credential Issuer.
  // Resolved to machine-readable info about the issuer, e.g. Public Keys
  "issuer": "did:example:565049",
  ....

  // (2) claims about the subject of the credential (holder)
  "credentialSubject": {
    // ID of credential subject (holder)
    // Resolved to machine-readable info about the holder, e.g. Public Keys
    "id": "did:example:ebfeb1f712ebc6f1c276e12ec21",
    "name": "John",
    "contractNum": 123456,
    ....
  },

  // (3) digital proof that makes the credential tamper-evident
  "proof": {
    "type": "Ed25519Signature2020",
    "created": "2022-02-25T14:58:42Z",
    "verificationMethod": "did:example:565049#key-1",
    "proofPurpose": "assertionMethod",
    "proofValue": "z3FjecWufY46yg5LhxhueARiKBk9czhsPTFeP...."
  }
}
```

Verifiable Presentation (VP)

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

  // (1) Verifiable Credential, see example at the left.
  "verifiableCredential": [{
    "issuer": "did:example:565049",
    ....
    "credentialSubject": {
      "id": "did:example:ebfeb1f712ebc6f1c276e12ec21",
      "name": "John",
      "contractNum": 123456,
      ....
    },
    "proof": { .... }
  }],

  // (2) digital signature by the credential holder
  // (proof of key possession)
  "proof": {
    "type": "Ed25519Signature2020",
    "created": "2022-03-25T16:37:21Z",
    "verificationMethod": "did:example:ebfeb1f712ebc6f1c276e12ec21#key-1",
    "proofPurpose": "assertionMethod",
    "proofValue": "f3FjecWufY46yg5Lhxued244Jsdfs,PsdflO2d...."
  }
}
```
VC Exchange Protocol: Hyperledger Aries

https://github.com/hyperledger/aries-rfcs

• **DIDComm** based
  • A secure, private communication methodology built atop the decentralized design of DIDs.
  • DIDComm v1: https://github.com/hyperledger/aries-rfcs/tree/main/concepts/0005-didcomm
  • DIDComm v2: https://identity.foundation/didcomm-messaging/spec/v2.0/

• **Protocols** (state machine) running on top of it
  • VC Exchange Protocols
  • Any other (custom) protocol
  • Composition of protocols
  • https://didcomm.org – list of protocols

From https://trustoverip.org/toip-model/
VC Exchange Protocol: Hyperledger Aries

https://github.com/hyperledger/aries-rfcs

• Aries Interop Profile v1
  • Establish connections (Connection protocol)
  • Exchange credentials and presentations via connections (Issuance and Presentation protocols)
  • Complete a connection-less proof-request/proof transaction

• Aries Interop Profile v2
  • Establish connections (DID Exchange protocol)
  • Exchange credentials and presentations via connections (Issuance and Presentation protocols)
  • Complete a connection-less proof-request/proof transaction
  • Reuse connections (out-of-band protocol)
  • Improved UX
  • Transition to DIDComm v2
  • Multiple ledger types and verifiable credential formats
  • Standard mediator coordination capabilities for mobile agents and multi-tenant agencies
VC Exchange Protocol: OID4VC

OAuth and OpenID Connect (OIDC)

Protocol that support authorization-based credential exchange where the holder authorizes a verifier (client) to access information on her behalf.

OpenID for VCs (OID4VC)

Protocol that supports self-sovereign credential exchange where the holder can autonomously control the exchange of credentials with any verifier she wants.
VC Exchange Protocol: OID4VC Standards

1.1 Authentication: SIOPv2

Defines how holders can authenticate in a self-sovereign way with any actor (Self-issued ID Token)

1.2 Presentation: OID4VP

Defines mechanisms on top of SIOPv2 to allow the presentation of claims in the form of Verifiable Credentials (complements self-issued ID token with cryptographically verifiable claims – VP token)
2. Issuance: **OID4VCI**

Defines APIs and the corresponding OAuth2-based authorisation mechanisms for the issuance of Verifiable Credentials

VDR: Cardano

https://github.com/IntersectMBO/cardano-node

Public Permissionless Blockchain
• Proof-of-stake consensus protocol Ouroboros
  • Peer-reviewed, verifiably secure
  • One of the pioneers of the proof-of-stake approach
• Applications: Smart Contracts, DApps, DeFi, DAO, NFT, SSI
• Main Net since 2017

Cardano and SSI
• AFJ extension: Cardano as AnonCreds Registry to publish Schema, CredDef, etc.
  https://github.com/roots-id/cardano-anoncreds
• Atala Prism, now open sourced as Hyperledger Labs Open Enterprise Agent
• VC Formats: W3C VC, Hyperledger AnonCreds
VDR: cheqd

https://github.com/cheqd/cheqd-node

- Public Permissionless Blockchain
- Application specific Blockchain: Decentralized Identity / SSI
- Built using the Cosmos SDK blockchain framework
- Proof-of-stake consensus protocol(Tendermint / Cosmos)
- Main Net since 2021
- SSI Features:
  - did:cheqd
  - Cheqd as AnonCreds registry
  - VC Formats: W3C VC, Hyperledger AnonCreds
  - DID-linked resources
  - Universal Resolver
VDR: Hyperledger Besu

https://github.com/hyperledger/besu

- Graduated/Active Hyperledger project since 2020
- Ethereum client written in Java
- Two use cases:
  - Public networks (such as public Ethereum nets)
  - Private Permissioned networks (such as enterprise or supply chain ledgers)
- Different approach for public and private cases (consensus, features, etc)
- Includes several consensus algorithms: PoS, PoW, PoA (IBFT 2.0, QBFT, Clique)
- Business logic as Solidity/Ethereum Smart Contracts
- Pluggable Architecture
- Private Permissioned Use Cases: CBDC, Supply Chain, Enterprise ledgers, SSI
**VDR: Indy-Besu**

[https://github.com/hyperledger/indy-besu](https://github.com/hyperledger/indy-besu)

- Public **permissioned** Ledger for SSI
- Indy (did:sov, did:indy) compatible
- Based on **Hyperledger Besu**, replaces Indy Plenum
  - QBFT consensus protocol instead of RBFT
- SSI business logic (DID and AnonCreds registry) as **Solidity/Ethereum smart contracts**
- Client library; integration with Credo (Aries frameworks JavaScript)
- SSI Features:
  - did:ethr
  - did:indybesu (WIP)
  - AnonCreds registry
  - VC Formats: W3C VC, Hyperledger AnonCreds
- Status:
  - Sep 2023: DSR proposes initiative at Indy Summit meeting
  - Nov 2023: DSR delivers PoC
  - Jan 2023: Code moved to a separate Hyperledger Indy (indy-besu)
  - Ongoing: MVP work
Demo
## Demo Scenarios

<table>
<thead>
<tr>
<th>Scenario</th>
<th>VC Format</th>
<th>VC Exchange Protocol</th>
<th>VDR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. AnonCreds + Cardano</td>
<td>Hyperledger AnonCreds</td>
<td>Hyperledger Aries</td>
<td>Cardano (as AnonCreds VDR) + did:key</td>
</tr>
<tr>
<td>2. AnonCreds + cheqd</td>
<td>Hyperledger AnonCreds</td>
<td>Hyperledger Aries</td>
<td>Cheqd (as AnonCreds VDR + did:cheqd)</td>
</tr>
<tr>
<td>3. W3C VC + cheqd</td>
<td>W3C VC (JSON-LD / Ed25519)</td>
<td>Hyperledger Aries</td>
<td>Cheqd (did:cheqd)</td>
</tr>
<tr>
<td>4. W3C VC + OID4VC</td>
<td>W3C VC (JWT / Ed25519)</td>
<td>OID4VC</td>
<td>did:key</td>
</tr>
</tbody>
</table>
Hands-On
### Demo Scenarios

<table>
<thead>
<tr>
<th>VC Formats</th>
<th>VC Exchange Protocol</th>
<th>VDR</th>
</tr>
</thead>
<tbody>
<tr>
<td>5. AnonCreds + Besu</td>
<td>Hyperledger AnonCreds</td>
<td>Hyperledger Besu (Indy-Besu)</td>
</tr>
<tr>
<td>6. W3C VC + Besu</td>
<td>W3C VC (JSON-LD / Ed25519)</td>
<td>Hyperledger Besu (Indy-Besu)</td>
</tr>
</tbody>
</table>
For the hands-on, we are going to use a Gitpod profile, which you can log into through your GitHub account.

• We ask all participants to make sure that they have at least a few hours of workspace usage in their Gitpod profile to participate in the hands-on. You can check this using the following link.

• Recommended browsers: Chrome or Firefox.
const assertKey = await this.agent.wallet.createKey({ keyType: KeyType.Ed25519 })

const createdDid = await this.agent.dids.create<IndyBesuDidCreateOptions>({ method: 'ethr',
  options: {
    verificationKeys: [
      {
        type: VerificationKeyType.Ed25519VerificationKey2018,
        key: assertKey,
        purpose: VerificationKeyPurpose.AssertionMethod,
      },
    ],
  },
})
Part 2: Write the Code
Issue AnonCreds Credential

Create credential

```javascript
const credential = {
  attributes: [
    { name: 'name', value: 'Alice Smith' },
    { name: 'degree', value: 'Computer Science' },
    { name: 'date', value: '01/01/2022' },
  ],
  credentialDefinitionId: this.credentialDefinition.credentialDefinitionId,
}
```

Offer credential

```javascript
const record = await this.agent.credentials.offerCredential({
  connectionId: connectionRecord.id,
  protocolVersion: 'v2',
  credentialFormats: {
    anoncreds: credential,
  },
})
```

Wait for accept

```javascript
await this.waitForAcceptCredential(record.id)
```
Part 3: Write the Code
Issue AnoncCreds Credential

Create credential

```javascript
const credential = {
  '@context': [CREDENTIALS_CONTEXT_V1_URL, 'https://www.w3.org/2018/credentials/examples/v1'],
  type: ['VerifiableCredential', 'FaberCollege'],
  issuer: this.issuerId,
  issuanceDate: '2023-12-07T12:23:48Z',
  credentialSubject: {
    name: 'Alice Smith',
    degree: 'Computer Science',
  },
}
```

Offer credential

```javascript
const record = await this.agent.credentials.offerCredential({
  connectionId: connectionRecord.id,
  protocolVersion: 'v2',
  credentialFormats: {
    jsonld: {
      credential: credential,
      options: {
        proofType: 'Ed25519Signature2018',
        proofPurpose: 'assertionMethod',
      },
    },
  },
})

await this.waitForAcceptCredential(record.id)
```

Wait for accept

```javascript
await this.waitForAcceptCredential(record.id)
```

issueJsonLdCredential
Run the Demo
AnonCreds + Besu

1. Select credential type: Hyperledger AnonCreds
2. Create DID
3. Register Schema
4. Register Credential Definition

5. Establish connection
   1. Issuer (Faber): Create connection invitation
   2. Holder (Alice): Receive connection invitation

6. Offer credential
   1. Issuer (Faber): Offer credential
   2. Holder (Alice): Accept credential

7. Proof credential
   1. Issuer (Faber): Request proof
   2. Holder (Alice): Accept Credential
Run the Demo
W3C JSON-LD + Besu

1. Select credential type: W3C JSON-LD
2. Create DID
3. Register Schema
4. Establish connection
   1. Issuer (Faber): Create connection invitation
   2. Holder (Alice): Receive connection invitation
5. Offer credential
   1. Issuer (Faber): Offer credential
   2. Holder (Alice): Accept credential
DOING SOFTWARE RIGHT