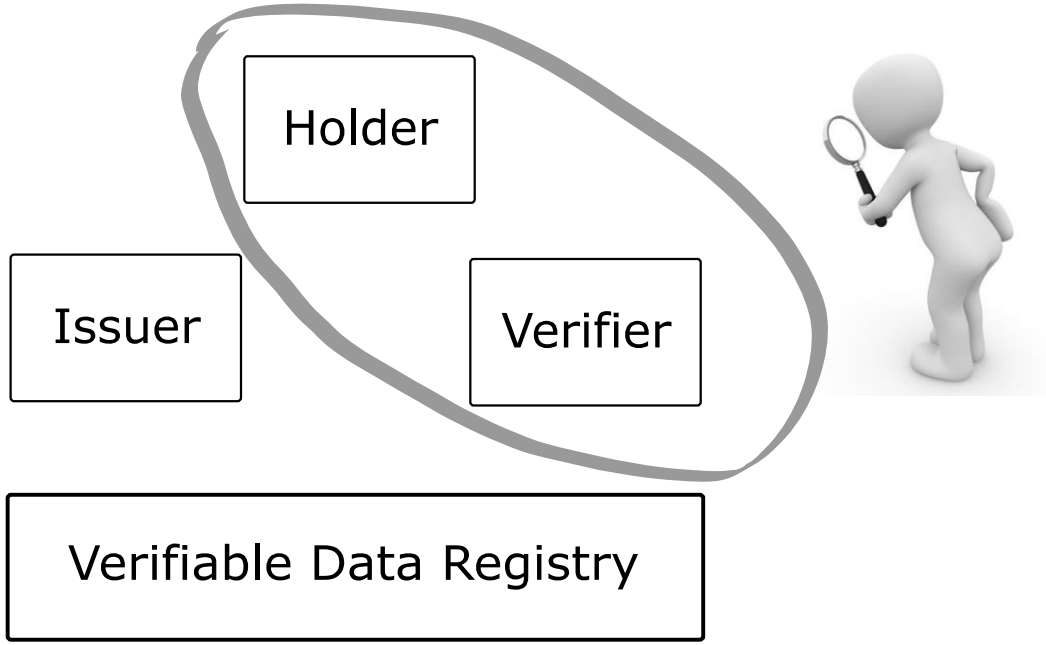
A Lightweight Alternative to DIDcomm and OpenID 4 VPs?

-

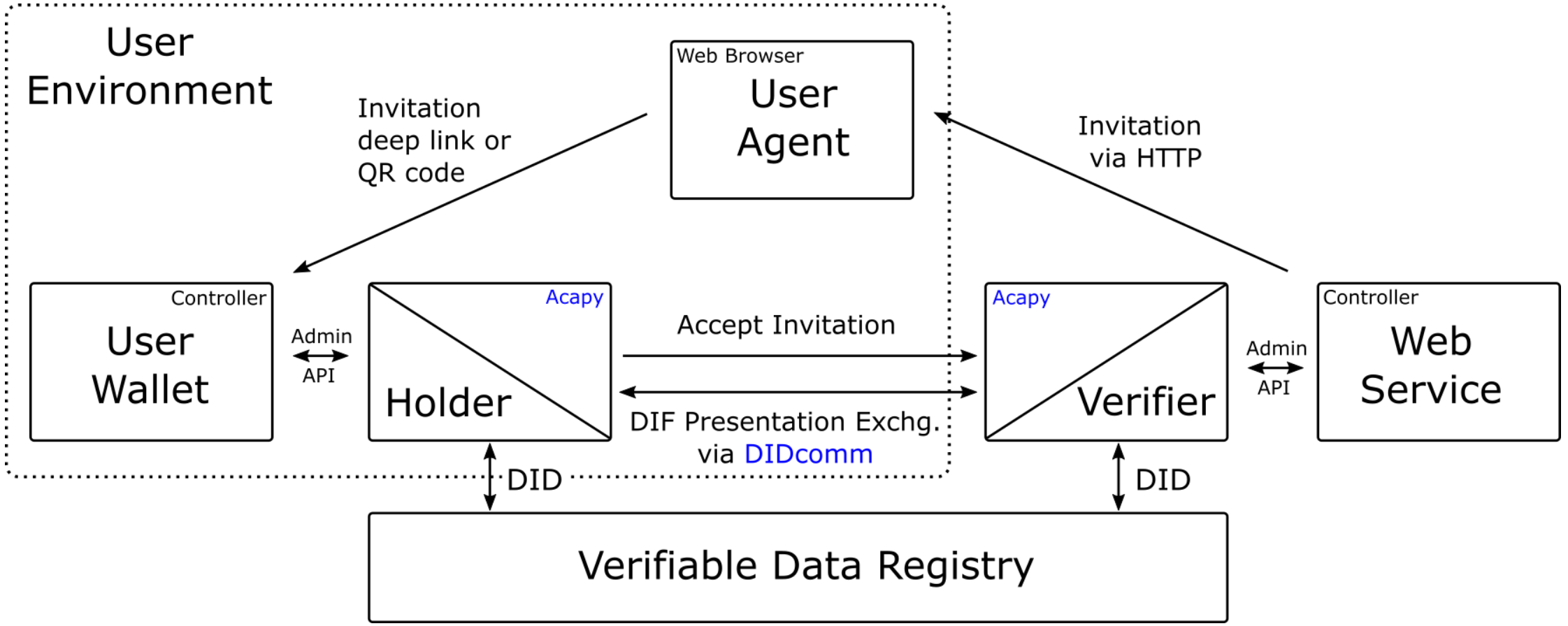
lowering implementation complexity for existing OIDC relying parties

Scope and Motivation

- DIDcomm is a great protocol, but brings some implementation complexity for existing (OIDC) relying parties that utilize **JOSE/JWT** processing capabilities for ID token/access token verification
- Relying Parties / Web Services want to enable resource access for (wallet-) holders based on verifiable credentials and verifiable presentations
- in scenarios where no persistent connection is needed (session-based approach)
- out-of-band initiated (QR code), cross device user interaction

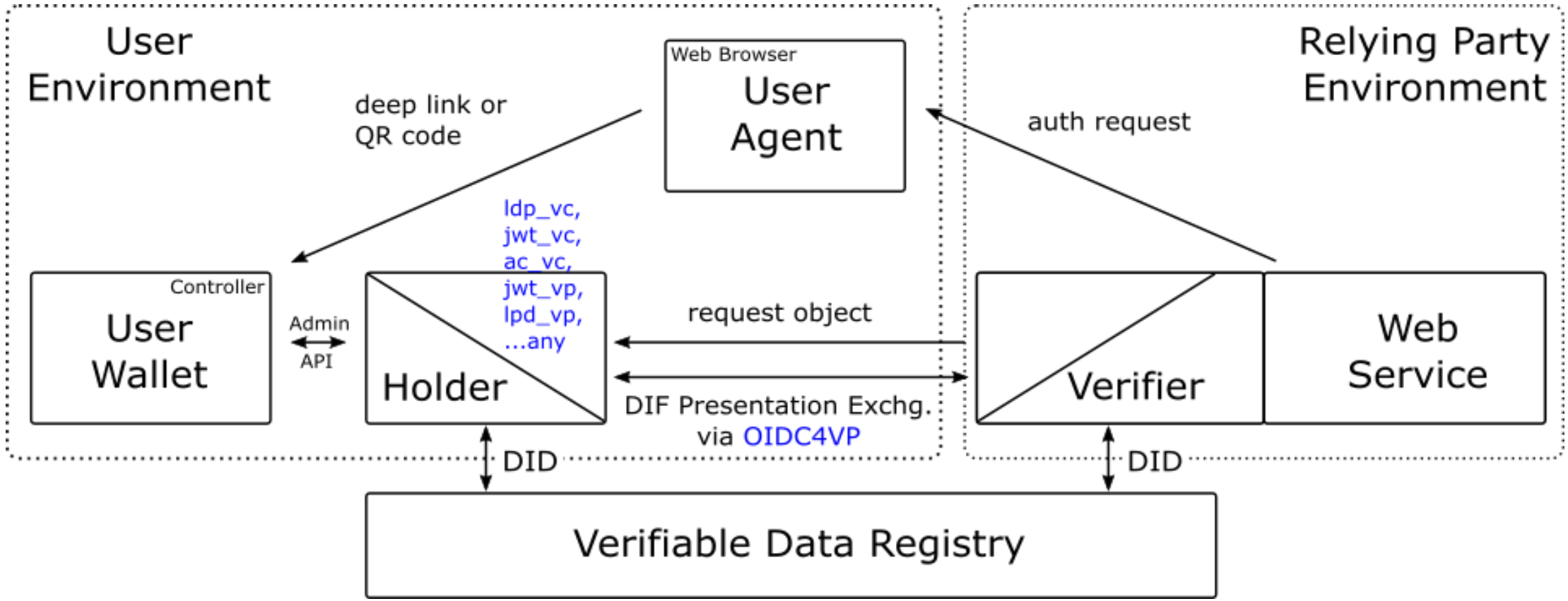


Starting Point: DIDcomm



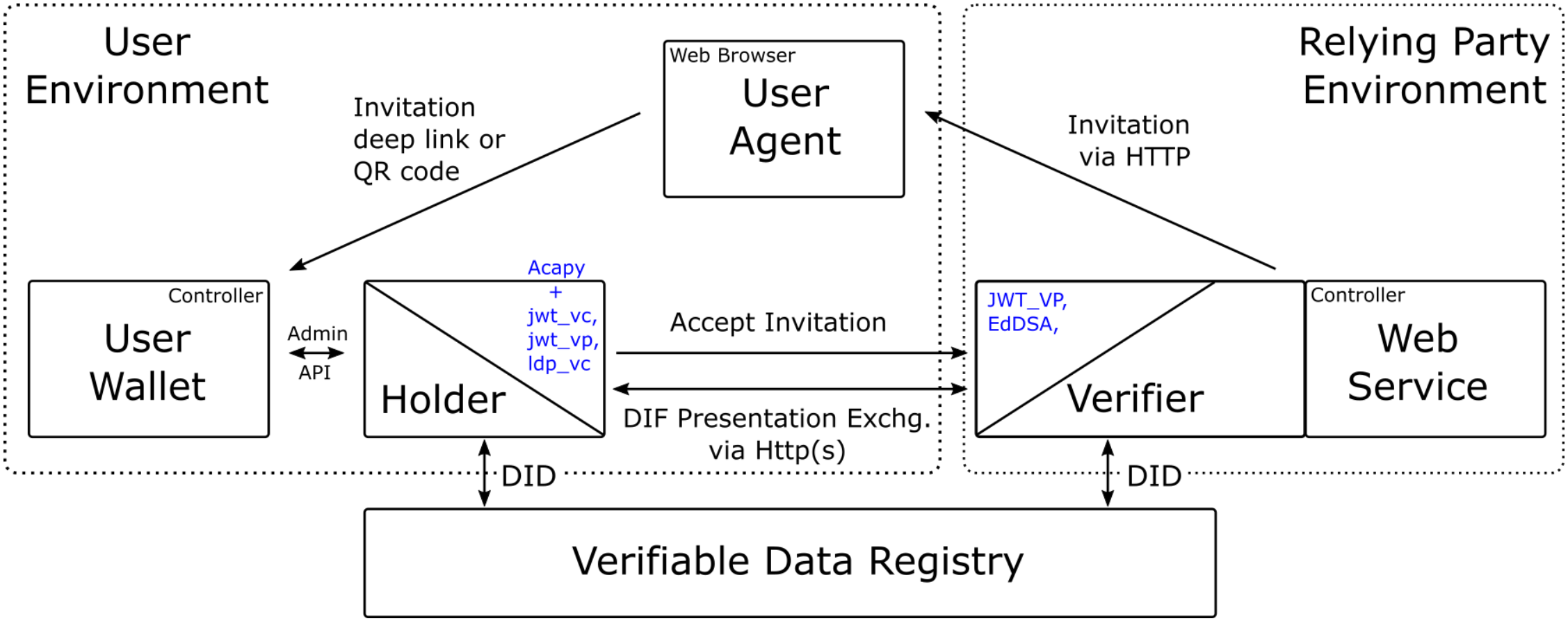
See https://github.com/bcgov/vc-authn-oidc/blob/main/docs/README.md for a similiar approach

oidc4vp Approach: Verifier-Initiated Cross Device Flow



See https://hackmd.io/@romanr/H1WXfAcQj for detailed messages and their sequence flow chart

Simplified Approach: px-over-http



See https://github.com/windley/IIW_homepage/blob/gh-pages/assets/proceedings/IIW_34_Book_of_Proceedings.pdf pages 205-206 for detailed messages and their sequence flow chart

Connection Establishment

DIDcomm vs. px-over-http

OOB Invitation Message

```
{
 "@id": "90d3878c-e58d-4111-a02c-8409717344f7 ",
 "@type": "https://didcomm.org/out-of-band/1.0/invitation",
 "handshake_protocols": [
   "https://didcomm.org/didexchange/1.0",
   "https://example.org/px-over-http-handshake/0.1",    // non-didcomm protocol
],
  "services": [
   {
    "id": „#inline",
    "id": "https://verifier.org",
    "type": "did-communication“,
              "px-over-http“,
    "serviceEndpoint": "https://verifier.org"
   }
  ]
}
```

OIDC4VP

Auth Request

```
https://wallet.verifier.org?
   client_id=https%3A%2F%2Fclient.verifier.org%2Fcb
   &request_uri=https%3A%2F%2Fclient.verifier.org%2F567545564
```

OOB Invitation Message

```
{
  "@id": "ba80c9a4-a087-42f3-97df-2612b21ba446",    // UUID generated by controller
  "@type": "https://didcomm.org/out-of-band/1.0/invitation",
  "handshake_protocols": [
    "https://example.org/oidc4vp-handshake/0.1"              // non-didcomm protocol
  ],
  "services": [
   {
    "id": "https://client.example.org",
    "serviceEndpoint":  "https://client.example.org/567545564",  // request_uri
    "type":  "oidc_request_uri"                                  // custom type
   }
  ]
}
```

Controller derives OOB invitation from auth request to make use of OOB protocol as single mechanism for connection establishment.
Alternatively, add API endpoint which accepts auth requests

All protocols start off by presenting a (dynamically generated) QR-Code that is scanned with a mobile device.

px-over-http at a glance

- RP creates OOB invitation for px-over-http
- Holder fetches presentation request from serviceEndpoint, providing the `invitation_msg_id`
- Presentation request contains only 3 parameters:
  `presentation_definition`, `nonce` and `session`
- Holder creates ID token: JWT_VP + OpenID attributes
- Holder POSTs response (ID token + session param) to serviceEndpoint

Bonus Feature:
To authenticate a previously registered holder, the RP can send an empty presentation definition, which results in a signed ID token which contains an empty presentation. –> Very fast verification.

Protocol Comparison

	advantages	Disadvantages
DIDcomm + Present Proof 2.0	• well-defined base communication protocol • very flexible, extensible • solid basis for presentation exchange • independent of „untrusted“ transport at lower layers • long lasting (persistent) connections • privacy preserving via mediators • async (offline) protocol via mediator	• all communicating partners need DIDs • implementation complexity applies at all comm-partners • existing application-specific protocols need to be implemented on top of DIDcomm

px-over-http	• simplified presentation exchange tailored to the capabilities of existing RPs:
	  ○ single presentation request/single proof
	  ○ ID token <-> JWT_VP
	  ○ EdDSA (JWT) <-> Ed25519 (LDP_VC)
	• less overhead than DIDcomm: uses transport layer security (HTTPS) instead of encryption envelope
	• self-attested claims in ID token and credentials about the same subject
	• simple migration path for existing RPs
		• only HTTPs, no persistent connections
		• PKI-based transport security (centralized or federated trust based on CAs and trustLists)
		• only W3C credentials
		• only EdDSA
		  ○ no selective disclosure
		  ○ no predicate proofs
		• no multiple proofs in one message
		• not privacy preserving via mediator

Protocol Comparison

	advantages	disadvantages
oidc4vp	• stems from a protocol family that is well defined by the OpenID Foundation and broadly used over the last decade • less overhead than DIDcomm: uses transport layer security (HTTPS) instead of encryption envelope • credential format agnostic, very flexible • self-attested claims in ID token	• only HTTPs, no persistent connections • PKI-based transport security (centralized or federated trust based on CAs and trustLists) • very complex: several communication/ message flows (e.g. on-device vs. cross-device) with many different variants (e.g.: deferred objects / uris for request and presentation_definition) • not privacy preserving via mediator

References

- **Aries RFC 0067: DIDComm DID document conventions**

Tobias Looker; Stephen Curran. 10 June 2019. Hyperledger Foundation. https://github.com/hyperledger/aries-rfcs/tree/main/features/0067-didcomm-diddoc-conventions

- **Aries RFC 0434: Out-of-Band Protocols**

Ryan West; Daniel Bluhm; Matthew Hailstone; Stephen Curran; Sam Curren; George Aristy. 1 March 2020. Hyperledger Foundation. https://github.com/hyperledger/aries-rfcs/tree/2da7fc4ee043effa3a9960150e7ba8c9a4628b68/features/0434-outofband

- **Aries RFC 0454: Present Proof Protocol 2.0**

Nikita Khateev; Stephen Curran. 27 May 2020. Hyperledger Foundation. https://github.com/hyperledger/aries-rfcs/tree/eace815c3e8598d4a8dd7881d8c731fdb2bcc0aa/features/0454-present-proof-v2

- **Decentralized Identifiers (DIDs) v1.0**

Manu Sporny; Amy Guy; Markus Sabadello; Drummond Reed. W3C. 3 August 2021. W3C Proposed Recommendation. https://www.w3.org/TR/did-core/

- **DID Specification Registries**

Orie Steele; Manu Sporny; Michael Prorock. W3C. 02 November 2021. W3C Working Group Note. https://www.w3.org/TR/did-spec-registries/

- **OpenID Connect Core 1.0**

N. Sakimura; J. Bradley; M. Jones; B. de Medeiros; C. Mortimore. The OpenID Foundation. 8 November 2014. Approved Specification. https://openid.net/specs/openid-connect-core-1_0.html

- **Presentation Exchange v1.0.0**

Daniel Buchner; Brent Zundel; Martin Riedel. DIF. Ratified Specification. https://identity.foundation/presentation-exchange/spec/v1.0.0/

- **Self-Issued OpenID Provider v2**

K. Yasuda; M.Jones. 28 January 2022. https://openid.net/specs/openid-connect-self-issued-v2-1_0.html

- **Verifiable Credentials Data Model v1.1**

Manu Sporny; Grant Noble; Dave Longley; Daniel C. Burnett; Brent Zundel; Kyle Den Hartog. W3C. 3 March 2022. W3C Recommendation. https://www.w3.org/TR/vc-data-model/

Thanks for your attention and feedback!