Client/Trust Management

Kristina Yasuda
Tobias Looker, Mattr
Torsten Lodderstedt, yes
OpenID for Verifiable Credentials

Issuer (Website)
Issue Credentials

Wallet
(user’s device, cloud or hybrid)
Present Credentials

Verifier (Website)

OpenID for Verifiable Credential Issuance
OpenID for Verifiable Presentations
Self-Issued OP v2

Protocols based on OAuth / OpenID Connect
Recap from IETF-115

- Verifiers need to work with multiple wallets, Wallets need to work with multiple Issuers
- Pre-registration with every Wallet or Issuer doesn’t scale
- Conclusion: client ids can be managed by trusted 3rd parties (OAuth 2.1)
Next challenge: so many different methods

- Dynamic Client registration with Software Statements
- x.509 certificates / PKI
- OpenID Connect Federation
- DIDs
- TRAIN
- client_id == redirect_uri
- IndieAuth
- Proprietary mechanisms (e.g. yes Ecosystem)

Multi homing wallets need to support many of them
How to determine actual method?

- Every method requires special treatment (signing, metadata source, additional request parameters)
- Previously, method was determined from existing authorization request parameters, e.g.:
  “When Verifier's “client_id” is expressed as an https URI, and does not equal to a “redirect_uri” value ...“
- Decided to go with a new Authorization Request parameter to let the client indicate to the AS what method to use
“client_id_scheme” Authorization Request Parameter

- A string identifying the scheme of the value in the `client_id` Authorization Request parameter
- Determines how the Wallet (acting as OAuth AS)
  - interprets the `client_id` value,
  - whether the request needs to be signed/authenticated,
  - what additional parameters need to be present,
  - where the client metadata can be obtained and how
- Supported client id schemes can be published in the Wallet’s (AS) metadata
- Supported by OpenID for Verifiable Presentations, plan to extend to OpenID for Verifiable Credential Issuance and SIOP v2 as well
Pre-Defined Options in OpenID for Verifiable Presentations

- pre-registered
- redirect_uri
- entity_id
- did

Under consideration

- x509
- train
Example authorization requests and processing rules

```
{  
  "client_id": "https://client.example.org/callback",
  "client_id_scheme": "redirect_uri",
  "response_types": "vp_token",
  "redirect_uri": "https://client.example.org/callback",
  "nonce": "n-0S6_WzA2Mj",
  "presentation_definition": "...",
  "client_metadata": "...
}
```

- Request must not be signed
- redirect_uri must equal client_id
- additional untrusted client metadata through parameter

```
{  
  "client_id": "https://client.example.org",
  "client_id_scheme": "entity_id",
  "response_types": "vp_token",
  "redirect_uri": "https://client.example.org/callback",
  "nonce": "n-0S6_WzA2Mj",
  "presentation_definition": "...
}
```

- Request must be signed
- any redirect_uri allowed
- additional client metadata through entity statements
Applicable to other OAuth-based applications?