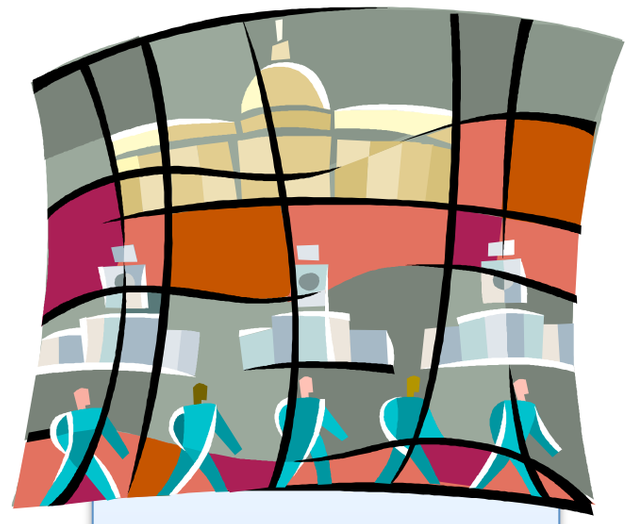
OpenID Connect

Presentation by Roland Hedberg at TNC2012

Fig.1 OpenID Authentication



notary

Google

1. Who are you. Get me a referral letter. Do not forget about Your email!

2. Plz write me a referral letter。

3. Here you are

4. Here is the certificate.

Eve

Alice

Name: Alice de Wonderland
Mail: alice@example.com
Notary: Google.

Official Google Seal

Name: Alice de Wonderland
Mail: alice@example.com
Notary: Google.

Official Google Seal

Fig.2 Pseudo-Authentication using OAuth

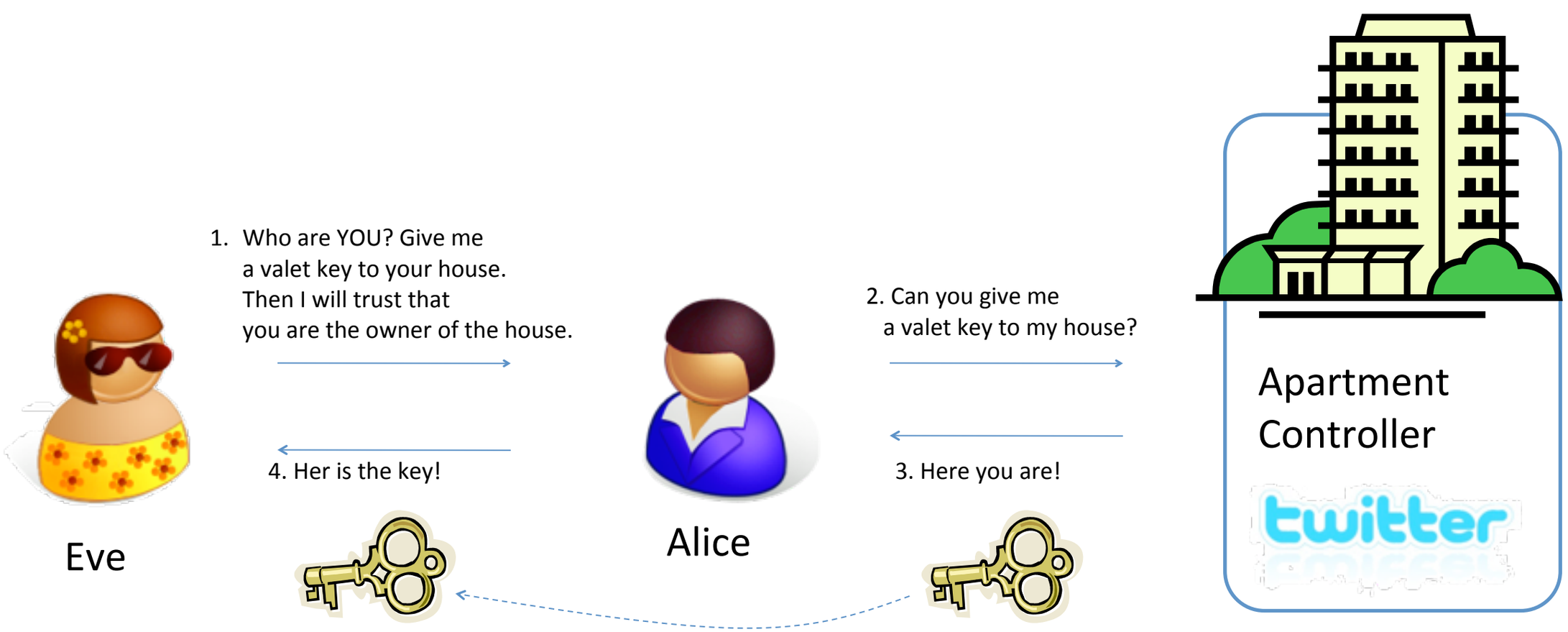


Fig.3 OpenID Connect Authentication

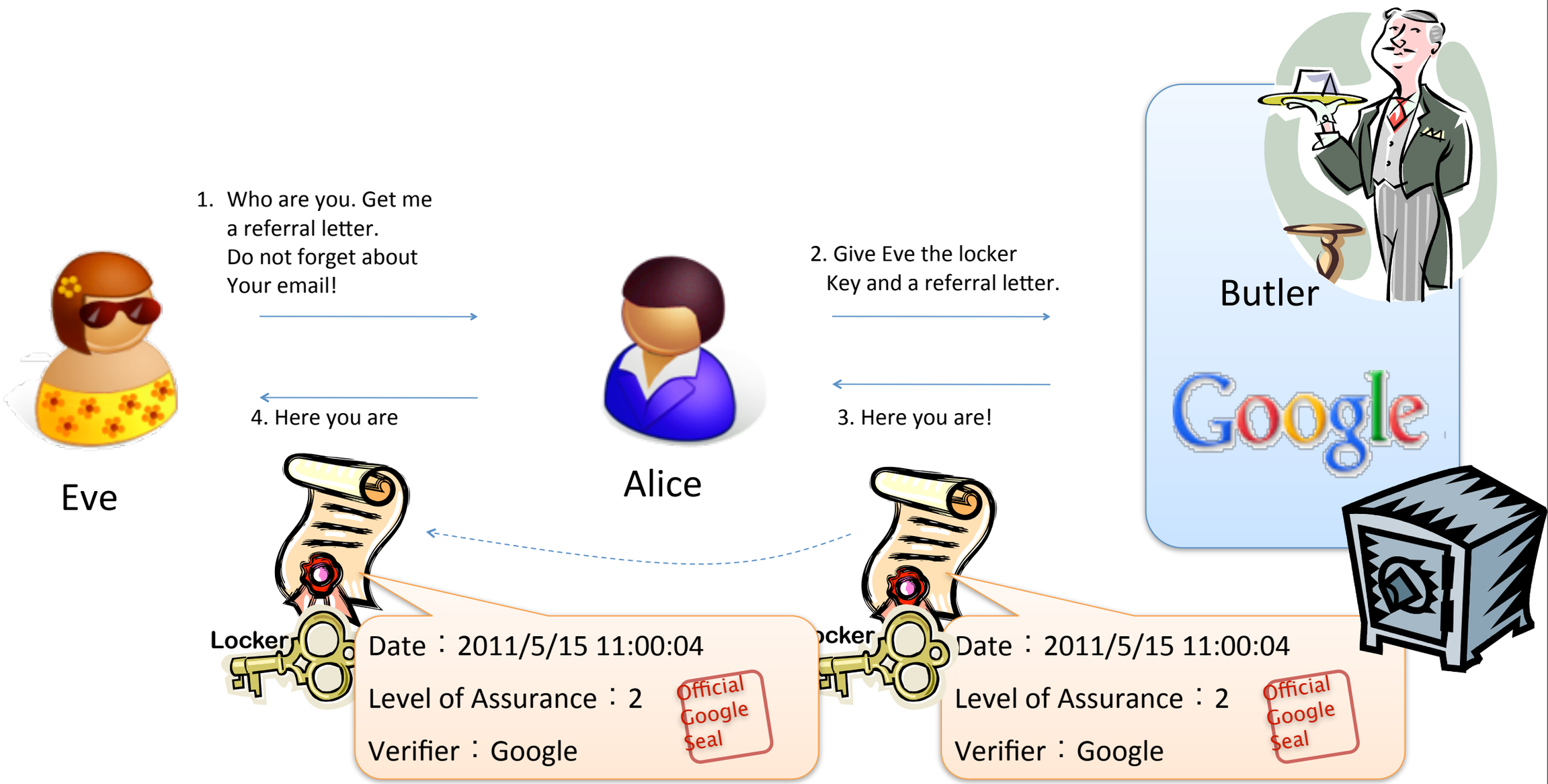
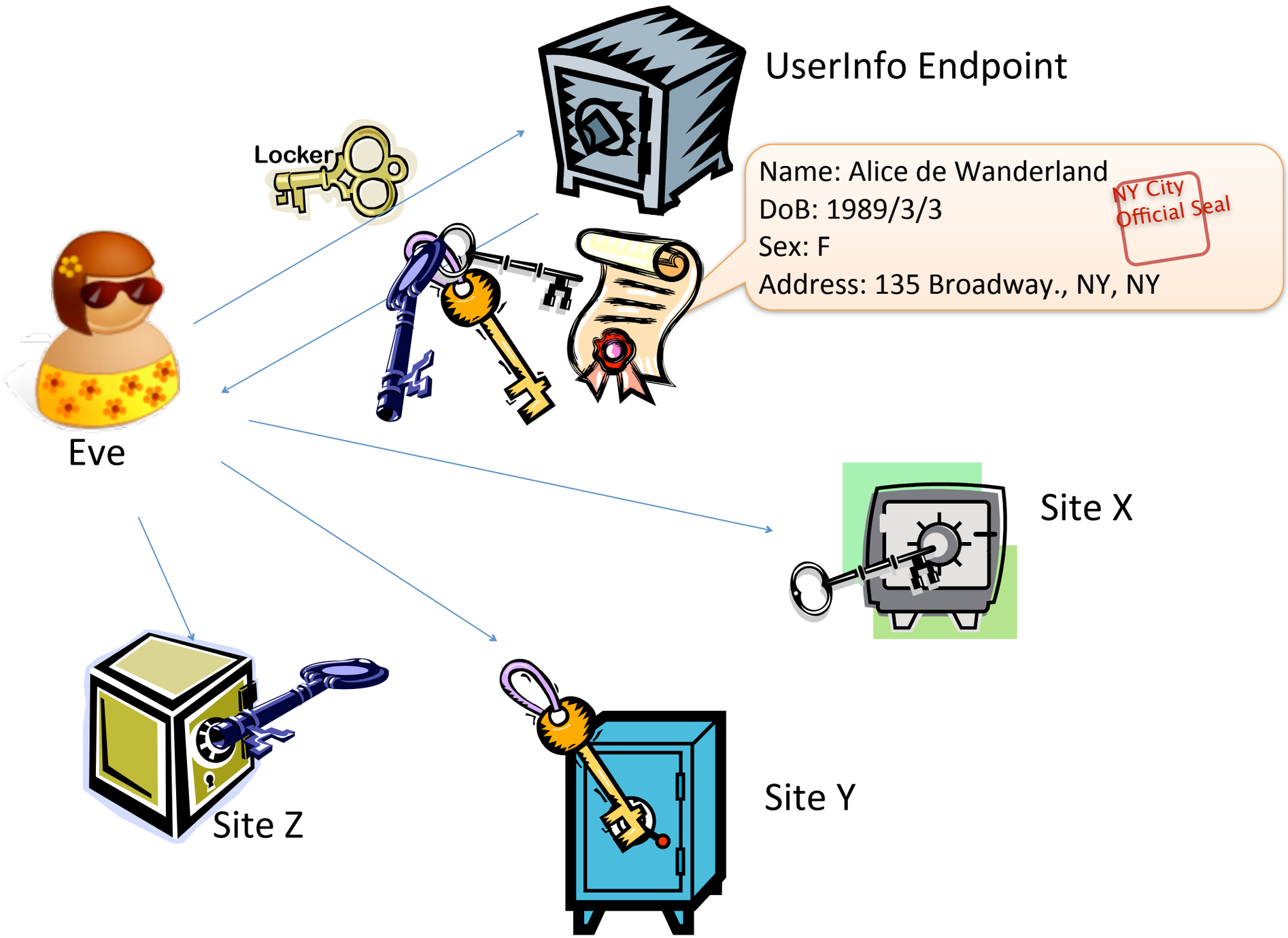
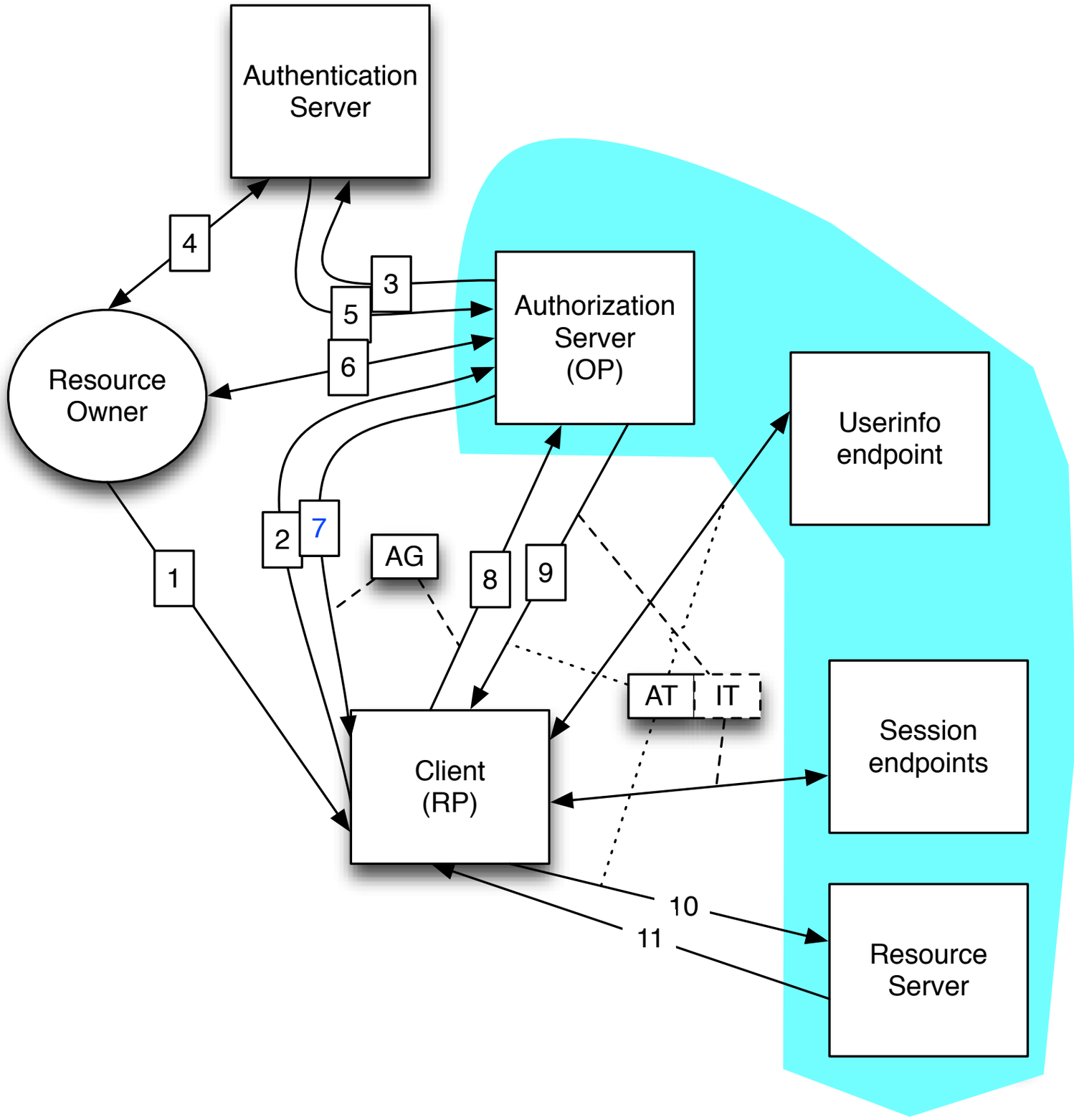


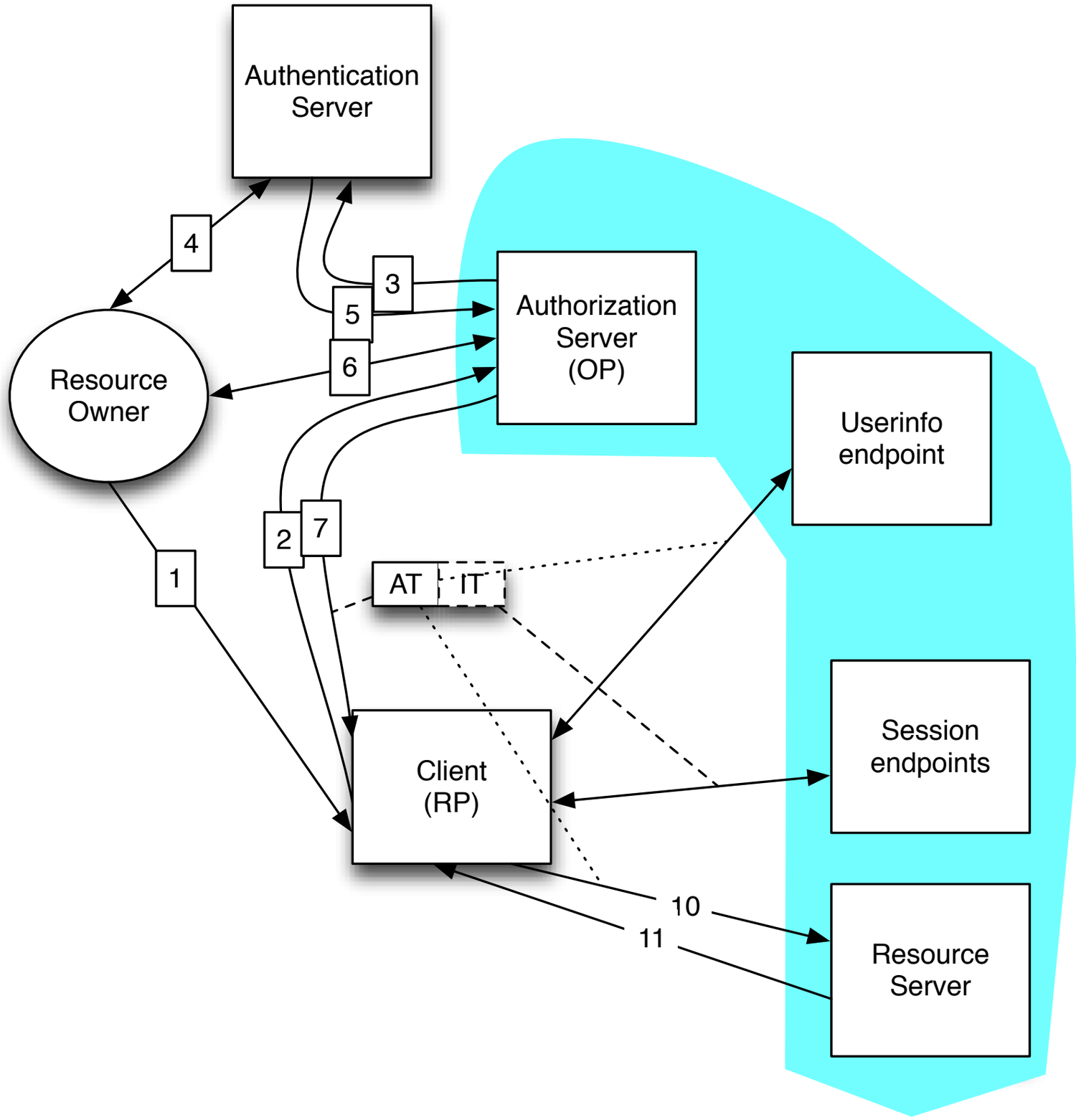
Fig.4 OpenID Connect's Clams aggregation and distributed claims.



Authorization Grant



Implicit


Authentication Server
4
3
5
Authorization Server (OP)
Resource Owner
6
Userinfo endpoint
2
7
1
AT
IT
Client (RP)
Session endpoints
10
11
Resource Server

A communication flow

(1) OP discovery

- *The user provides an identifier (for instance an email address)*
- *Using webfinger the OP is found*

$ curl -G https://connect-op.heroku.com/.well-known/host-meta.jso

{"links":[{"rel":"http://openid.net/specs/connect/1.0/issuer",
"href":"https://connect-op.heroku.com"}]}

(2) OP functionality discovery

$ curl -G -k https://localhost:8088/.well-known/openid-configuration

{"registration_endpoint": "https://localhost:8088/registration",
"userinfo_endpoint": "https://localhost:8088/userinfo",
"token_endpoint": "https://localhost:8088/token",
"authorization_endpoint": "https://localhost:8088/authorization",
"check_id_endpoint": "https://localhost:8088/check_id",
"token_endpoint_auth_types_supported": ["client_secret_post",
"client_secret_basic", "client_secret_jwt"],
"jwk_url": "https://localhost:8088/static/jwk.json",
"user_id_types_supported": ["public"],
"scopes_supported": ["openid"],
"version": "3.0",
"response_types_supported": ["code", "token", "id_token", "code token",
"code id_token", "token id_token", "code token id_token"],
"issuer": "https://localhost:8088/"}

(3) Dynamic registration

POST

https://localhost:8088/registration

```
application_name=OIC+test+tool
&application_type=web
&redirect_uris=https://smultron.catalogix.se/authz_cb
&type=client_associate
&contact=roland@example.com
```

(4) Authorization Request

https://server.example.com/op/authorize?
response_type='code id_token'
&client_id=s6BhdRkqt3
&redirect_uri=https://client.example.com/cb
&scope=openid
&nonce=n-0S6_WzA2Mj
&state=af0ifjsldkj
&**request**=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJyZXNwb25zZV90eXBlIjoiY2
9kZSBpZF90b2tlbiIsImNsaWVudF9pZCI6InM2QmhkUmtxdDMiLCJyZWRpcmVjdF91cmkiOi
JodHRwczpcL1wvY2xpZW50LmV4YW1wbGUuY29tXC9jYiIsInNjb3BlIjoib3BlbmlkIHByb2
ZpbGUiLCJzdGF0ZSI6ImFmMGlmanNsZGtqIiwidXNlcmluZm8iOnsiY2xhaW1zIjp7Im5hbW
UiOm51bGwsIm5pY2tuYW1lIjp7Im9wdGlvbmFsIjp0cnVlfSwiZW1haWwiOm51bGwsInZlcm
lmaWVkIjpudWxsLCJwaWN0dXJlIjp7Im9wdGlvbmFsIjp0cnVlfX0sImZvcm1hdCI6InNpZ2
5lZCJ9LCJpZF90b2tlbiI6eyJtYXhfYWdlIjo4NjQwMCwiaXNvMjkxMTUiOiIyIn19.2OiqR
grbrHkA1FZ5p_7bc_RSdTbH-wo_Agk-ZRpD3wY

SAML AuthnRequest

```
<?xml version='1.0' encoding='UTF-8'?>
<ns0:AuthnRequest xmlns:ns0="urn:oasis:names:tc:SAML:2.0:protocol"
                  xmlns:ns1="urn:oasis:names:tc:SAML:2.0:assertion"
                  AssertionConsumerServiceURL="http://www.example.org/service"
                  Destination="http://www.example.com/sso" ID="id1234567890"
                  IssueInstant="2011-11-14T08:16:15Z"
                  ProtocolBinding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
                  ProviderName="My Name" Version="2.0">
    <ns1:Issuer Format="urn:oasis:names:tc:SAML:2.0:nameid-format:entity">
        http://example.org/saml/sp
    </ns1:Issuer>
    <ns0:NameIDPolicy AllowCreate="true"
                      Format="urn:oasis:names:tc:SAML:2.0:nameid-format:transient"/>
</ns0:AuthnRequest>
```

Json Web Token (JWT)

- Header
  - {"typ":"JWT","alg":"HS256"}
  - Base64 encoding of the UTF-8 representation
- Second part
  - When the JWT is signed, the JWT Second Part is the Encoded JWS Payload. When the JWT is encrypted, the JWT Second Part is the Encoded JWE Encrypted Key.
- Third part
  - When the JWT is signed, the JWT Third Part is the Encoded JWS Signature. When the JWT is encrypted, the JWT Third Part is the Encoded JWE Ciphertext.

Authorization -*request* object unpacked

```
{
  "response_type": "code id_token",
  "client_id": "s6BhdRkqt3",
  "redirect_uri": "https://client.example.com/cb",
  "scope": "openid profile",
  "state": "n-0S6_WzA2Mj",
  "nonce": "af0ifjsldkj",
  "userinfo"{
    "claims":{
      "name": null,
      "given_name": null,
      "family_name": null,
      "nickname": {"optional": true},
      "email": null,
      "verified": null,
      "picture": {"optional": true},
      },
    "format": "signed
    }
  "id_token":{
    "max_age": 86400,
    "acr": "2"
  }
}
```

Authorization Response

```
HTTP/1.1 302 Found
Location: https://client.example.com/cb?
code=SplxlOBeZQQYbYS6WxSbIA
&state=af0ifjsldkj
&nonce=n-0S6_WzA2Mj
&id_token=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI
1NiJ9.eyJpc3MiOiJodHRwOlwvXC9zZXJ2ZXIuZXh
hbXBsZS5jb20iLCJ1c2VyX2lkIjoiMjQ4Mjg5NzYx
MDAxIiwiYXVkIjoiaHR0cDpcL1wvY2xpZW50LmV4Y
W1wbGUuY29tIiwiZXhwIjoxMzExMjgxOTcwfQ.eDe
sUD0vzDH3T1G3liaTNOrfaeWYjuRCEPNXVtaazNQ
```

IDToken unpacked

```
{
 "iss": "https://server.example.com/op",
 "user_id":"24400320",
 "aud":"s6BhdRkqt3",
 "exp":1320502962,
 "acr": 2,
 "nonce":"af0ifjsldkj",
 "auth_time":1320502000
}
```

(5) UserInfoRequest

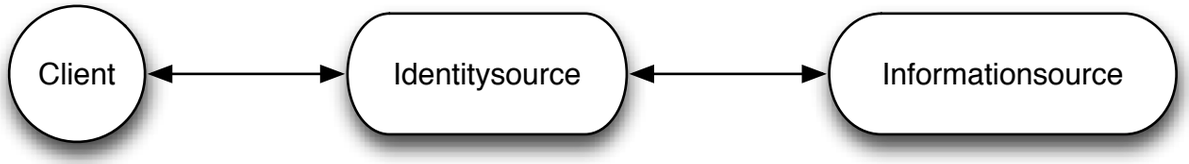
```
POST
https://server.example.com/op/userinfo

access_code=SlAV32hkKG&schema=openid
```

UserInfoResponse

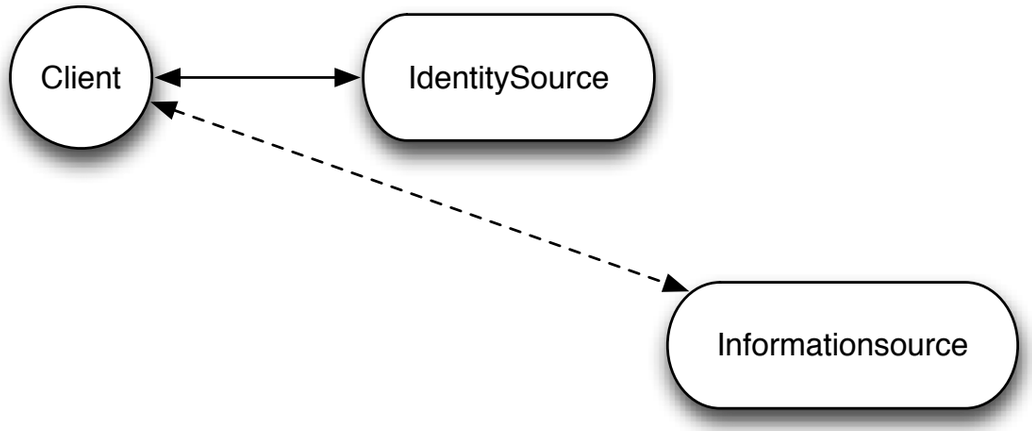
```
{
   "name#sv-se": "Jane Doe"
   "given_name": "Jane",
   "family_name": "Doe",
   "email": "janedoe@example.com",
   ”verified”: true,
   "picture": "http://example.com/janedoe/me.jpg"
}
```

Aggregated claims




```
{
  "name": "Jane Doe",
  "given_name": "Jane",
  "family_name": "Doe",
  "birthday": "01/01/2001",
  "email": "janedoe@example.com",
  "_claim_names": {
    "address": "src1",
    "phone_number": "src1",
  },
  "_claim_sources": {
    "src1": {"JWT":"jwt_header.jwt_part2.jwt_part3"},
  }
}
```

Distributed claims




```
{
  "name": "Jane Doe",
  "given_name": "Jane",
  "family_name": "Doe",
  "email": "janedoe@example.com",
  "birthday": "01/01/2001",
  "_claim_names": {
    "payment_info": "src1",
    "shipping_address": "src1",
    "credit_score": "src2"
  },
  "_claim_sources": {
    "src1": {"endpoint": "https://bank.example.com/claimsource"}
    "src2": {
      "endpoint": "https://creditagency.example.com/claimshere",
      "access_token": "ksj3n283dke"
    }
  }
}
```

OpenIDConnect vs SAML2

- *Easy to implement !!*
- *JSON instead of XML*
- *Working signature/encryption*
- *Delegated claims*
- *No SOAP/POAS only HTTP GET/POST*
- *Service discovery*
- *Dynamic client registration*
- *No metadata (presently)*
- *Limited identity attribute set*

Implementation status

- *Implementations in Java, Ruby, Python, (PHP)*
- *Interop tests ongoing*
- *Mine and Andreas's conformance test web service*
  - *in operation*

Testing a single flow
$ ./ebay.py | oicc.py -J - 'mj-01'

Discovery

- {"status": 0, "message": {"registration_endpoint": "https://openidconnect.ebay.com/oreo/register.jsp", "userinfo_endpoint": "https://openidconnect.ebay.com/oreo/openidconnect/get-user-info.jsp", "token_endpoint_auth_types_supported": ["client_secret_basic"], "scopes_supported": ["openid", "email", "location"], "refresh_session_endpoint": "https://openidconnect.ebay.com/oreo/openidconnect/refresh-session.jsp", "token_endpoint": "https://openidconnect.ebay.com/oreo/token.jsp", "version": "3.0", "response_types_supported": ["token", "code", "code id_token", "token id_token"], "end_session_endpoint": "https://openidconnect.ebay.com/oreo/openidconnect/end-session.jsp", "authorization_endpoint": "https://openidconnect.ebay.com/oreo/authorize.jsp", "check_id_endpoint": "https://openidconnect.ebay.com/oreo/openidconnect/check-session.jsp", "issuer": "https://openidconnect.ebay.com"}, "id": "check", "name": "Provider Configuration Response"},

Registration

- {"status": 1, "url": "https://openidconnect.ebay.com/oreo/register.jsp", "id": "check-http-response", "name": "Checks that the HTTP response status is within the 200 or 300 range"}
- {"status": 0, "message": "{\"client_id\":\"o1hqe2m52v6925vr68m8jffru6\",\"client_secret\":\"6FEDD5C0642932E7C6185306D5F2D25225EF19A1\",\"expires_at\":900}", "id": "check", "name": "Registration Response"},
- {"status": 1, "id": "check_content_type_header", "name": "Verify that the content-type header is what it should be."},
- {"status": 1, "url": "https://openidconnect.ebay.com/oreo/register.jsp", "response_type": "RegistrationResponse", "id": "response-parse", "name": "Parsing the response"},
- {"status": 1, "id": "check-response-type", "name": "Checks that the asked for response type are among the supported"},

AuthorizationRequest

- {"status": 1, "url": "https://openidconnect.ebay.com/oreo/authorize.jsp?nonce=VgGJC79pwuBP&state=STATE0&redirect_uri=https%3A%2F%2Fsmultron.catalogix.se%2Fauthz_cb&response_type=code&client_id=o1hqe2m52v6925vr68m8jffru6&scope=openid", "id": "check-http-response", "name": "Checks that the HTTP response status is within the 200 or 300 range"},
- {"status": 1, "url": "https://openidconnect.ebay.com/oreo/primary-auth/dummy-signin.jsp?ru=https%3A%2F%2Fopenidconnect.ebay.com%2Foreo%2Fvalidate-auth.jsp&openid.ns=http%3A%2F%2Fspecs.openid.net%2Fauth%2F2.0&openid.mode=checkid_setup&openid.return_to=http%3A%2F%2Fhead2toes.org%2Fsandbox%2Fopenid%2Flightopenid-lightopenid%2Fexample.php%3Fproxy%3Dhttps%3A%2F%2Fopenidconnect.ebay.com%2Foreo%2Fvalidate-auth.jsp&openid.realm=http%3A%2F%2Fhead2toes.org%2Fsandbox%2Fopenid%2Flightopenid-lightopenid%2Fexample.php&openid.ns.sreg=http%3A%2F%2Fopenid.net%2Fextensions%2Fsreg%2F1.1&openid.claimed_id=http%3A%2F%2Fspecs.openid.net%2Fauth%2F2.0%2Fidentifier_select&openid.identity=http%3A%2F%2Fspecs.openid.net%2Fauth%2F2.0%2Fidentifier_select", "id": "check-http-response", "name": "Checks that the HTTP response status is within the 200 or 300 range"},
- {"status": 1, "url": "https://openidconnect.ebay.com/oreo/validate-auth.jsp?openid.identity=test&username=test&password=password&user_id=np01&confirmation=n%2Fa&confirmationExpiry=4049", "id": "check-http-response", "name": "Checks that the HTTP response status is within the 200 or 300 range"},
- {"status": 1, "url": "https://openidconnect.ebay.com/oreo/consent/consent-default.jsp", "id": "check-http-response", "name": "Checks that the HTTP response status is within the 200 or 300 range"},
- {"status": 1, "url": "https://openidconnect.ebay.com/oreo/consent/consent-plain.jsp", "id": "check-http-response", "name": "Checks that the HTTP response status is within the 200 or 300 range"},
- {"status": 1, "url": "https://openidconnect.ebay.com/oreo/validate-consent.jsp?consent_user_response=pending&consent_confirmation_nonce=F3DB97731912AB2AAE93B050ED1D58490CAF397057149AC63473D5123F9120BE&consent_scope=ID", "id": "check-http-response", "name": "Checks that the HTTP response status is within the 200 or 300 range"},
- {"status": 1, "url": "https://openidconnect.ebay.com/oreo/sts/token.jsp", "id": "check-http-response", "name": "Checks that the HTTP response status is within the 200 or 300 range"},
- {"status": 1, "url": "https://openidconnect.ebay.com/oreo/sts/token.jsp", "response_type": "AuthorizationResponse", "id": "response-parse", "name": "Parsing the response"},
- {"status": 1, "id": "check-authorization-response", "name": "Verifies an Authorization response. This is additional constrains besides what is optional or required."}

Doing a sequence of flows
$ oic_flow_tests.py kodtest

---

* (mj-00)Client registration Request - OK
* (mj-01)Request with response_type=code - OK
* (mj-02)Request with response_type=token - OK
* (mj-03)Request with response_type=id_token - OK
* (mj-04)Request with response_type=code token - OK
* (mj-05)Request with response_type=code id_token - OK
* (mj-06)Request with response_type=id_token token - OK
* (mj-07)Request with response_type=code id_token token - OK
* (mj-08)Check ID Endpoint Access with GET and bearer_header - OK
* (mj-09)Check ID Endpoint Access with POST and bearer_header - OK
* (mj-10)Check ID Endpoint Access with POST and bearer_body - OK
* (mj-11)UserInfo Endpoint Access with GET and bearer_header - OK
* (mj-12)UserInfo Endpoint Access with POST and bearer_header - OK
* (mj-13)UserInfo Endpoint Access with POST and bearer_body - OK
* (mj-14)Scope Requesting profile Claims - OK
* (mj-15)Scope Requesting email Claims - OK
* (mj-16)Scope Requesting address Claims - OK
* (mj-17)Scope Requesting phone Claims - OK
* (mj-18)Scope Requesting all Claims - OK
* (mj-19)OpenID Request Object with Required name Claim - OK
* (mj-20)OpenID Request Object with Optional email and picture Claim - OK
* (mj-21)OpenID Request Object with Required name and Optional email and picture Claim - OK
* (mj-22)Requesting ID Token with auth_time Claim - OK
* (mj-23)Requesting ID Token with Required acr Claim - OK
* (mj-24)Requesting ID Token with Optional acr Claim - OK
* (mj-25a)Requesting ID Token with max_age=1 seconds Restriction - OK
* (mj-25b)Requesting ID Token with max_age=10 seconds Restriction - OK
* (mj-26)Request with display=page - OK
* (mj-27)Request with display=popup - OK
* (mj-28)Request with prompt=none - OK
* (mj-29)Request with prompt=login - OK
* (mj-30)Access token request with client_secret_basic authentication - OK
* (mj-31)Request with response_type=code and extra query component - OK
* (mj-32)Request with redirect_uri with query component - OK
* (mj-33)Registration where a redirect_uri has a query component - OK
* (mj-34)Registration where a redirect_uri has a fragment - OK
* (mj-35)Authorization request missing the 'response_type' parameter - OK
* (mj-36)The sent redirect_uri does not match the registered - OK
* (mj-37)Access token request with client_secret_jwt authentication - OK
* (mj-38)Access token request with public_key_jwt authentication - OK
* (mj-39)Trying to use access code twice should result in an error - OK
* (mj-40)Trying to use access code twice should result in revoking previous issued tokens - OK