OAuth2 testing

Roland Hedberg @ IETF86

- Why
- What
- How

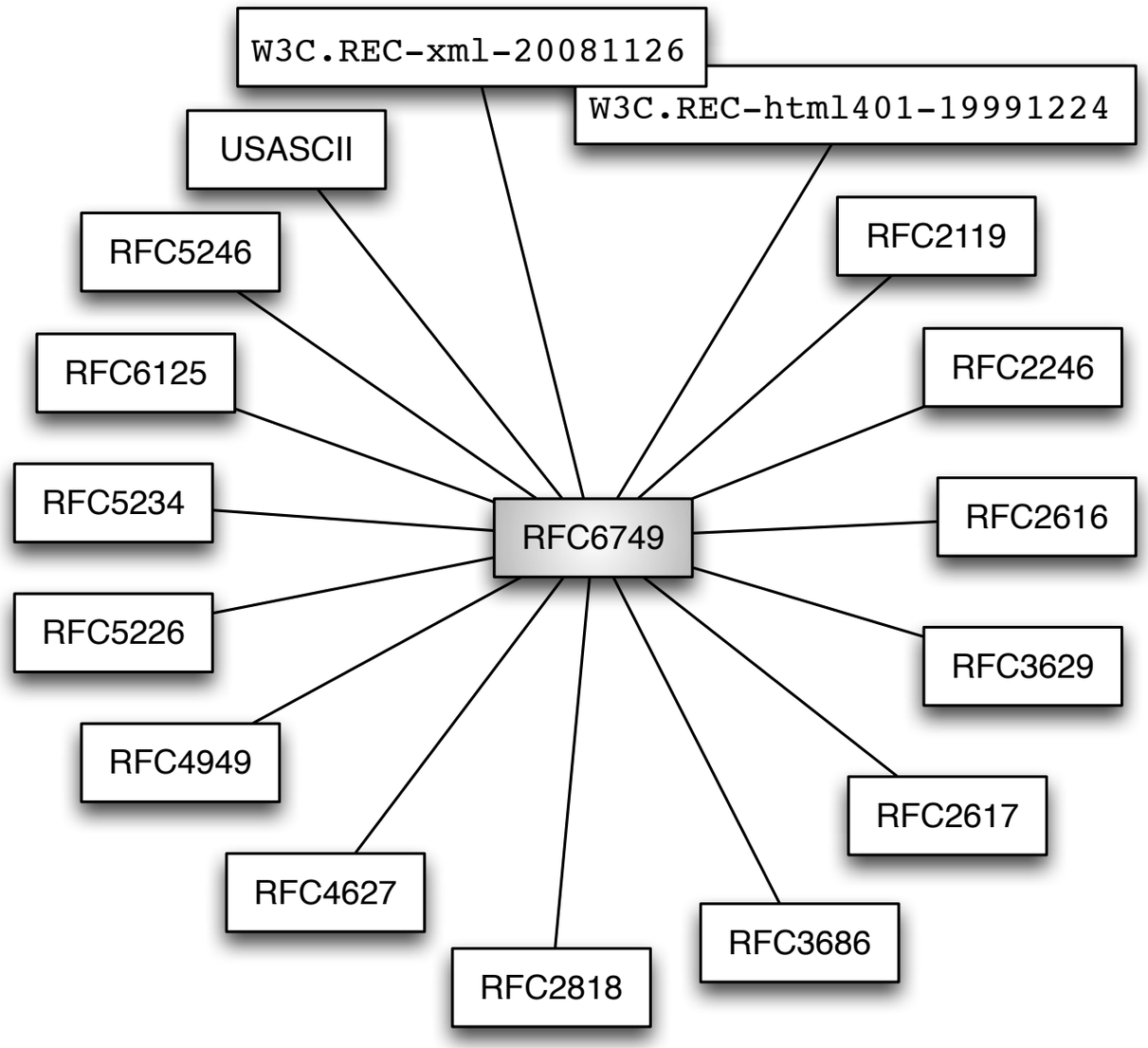
Why me ?

- Worked on a test tool for OpenID Connect imlementation the last year and a half.
- Since a couple of month back working on similar test tool for SAML2

Why ?

- It's hard to get it right
  - Standard is written in english
    - Open for interpretation
- Dependency on other standards
  - You have to be quite knowledgeable

Normative references



What to test ?

RFC2119 keyword usage in RFC6749

MUST	106
MUST NOT	29
REQUIRED	28
SHALL	3
SHALL NOT	0
SHOULD	43
SHOULD NOT	6
RECOMMENDED	6
MAY	27
OPTIONAL	14
	262

Testable ?

2.3. Client Authentication

- The authorization server MAY accept any form of client authentication meeting its security requirements.

- The authorization server MAY establish a client authentication method with public clients.

- However, the authorization server MUST NOT rely on public client authentication for the purpose of identifying the client.

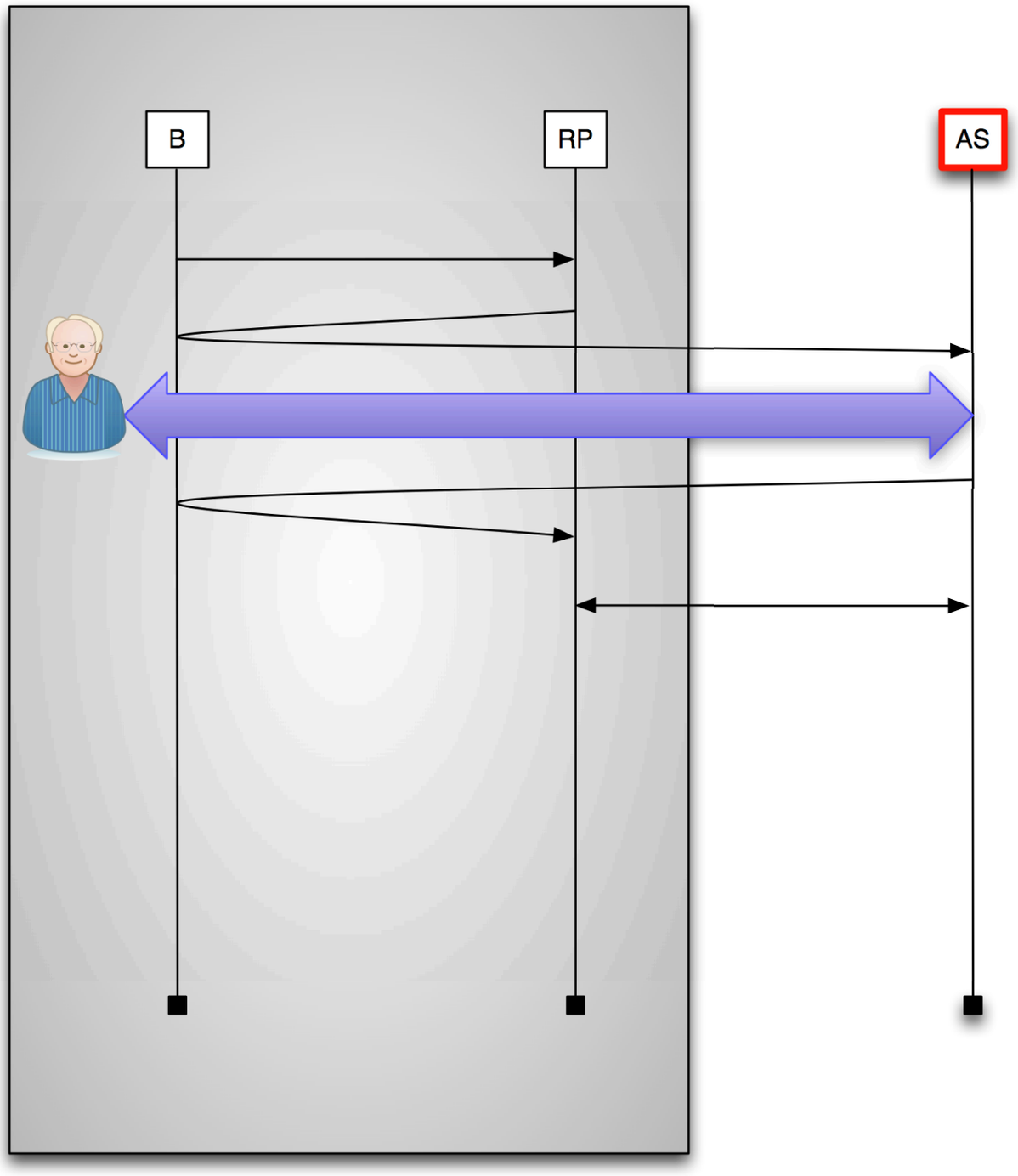
- The client MUST NOT use more than one authentication method in each request.

3.2 Token endpoint

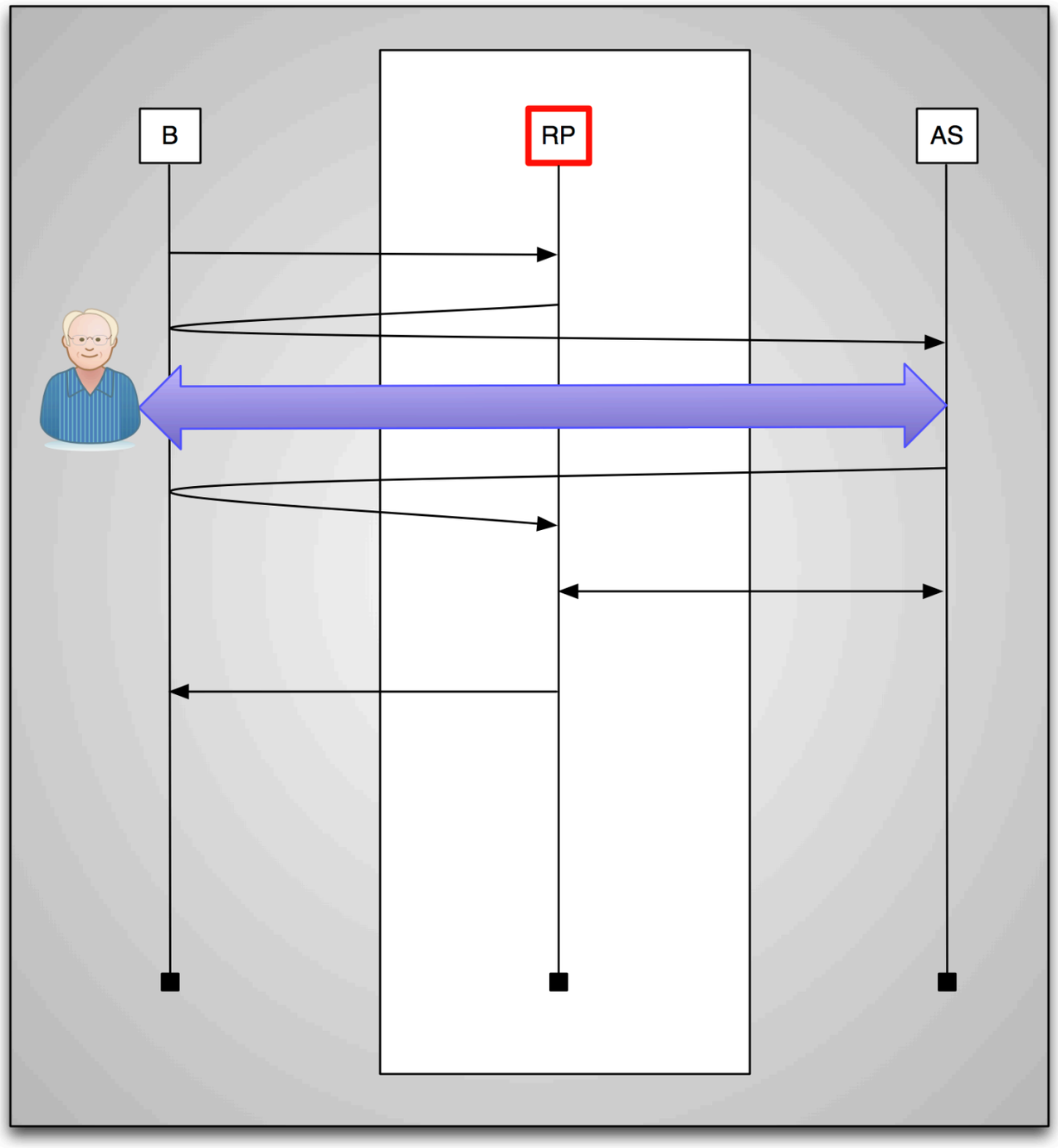
- The endpoint URI MAY include an "application/x-www-form-urlencoded" formatted (per Appendix B) query component ([RFC3986] Section 3.4).
- If the endpoint URI includes a query component, this MUST be retained when adding additional query parameters.
- The endpoint URI MUST NOT include a fragment component.
- The authorization server MUST require the use of TLS as described in Section 1.6 when sending requests to the token endpoint (since requests result in user authentication and the transmission of clear-text credentials).
- The client MUST support the use of the HTTP "POST" method when making access token requests.
- Parameters sent without a value MUST be treated as if they were omitted from the request.
- The authorization server MUST ignore unrecognized request parameters.
- Request and response parameters MUST NOT be included more than once.

How ?

Testing the AS



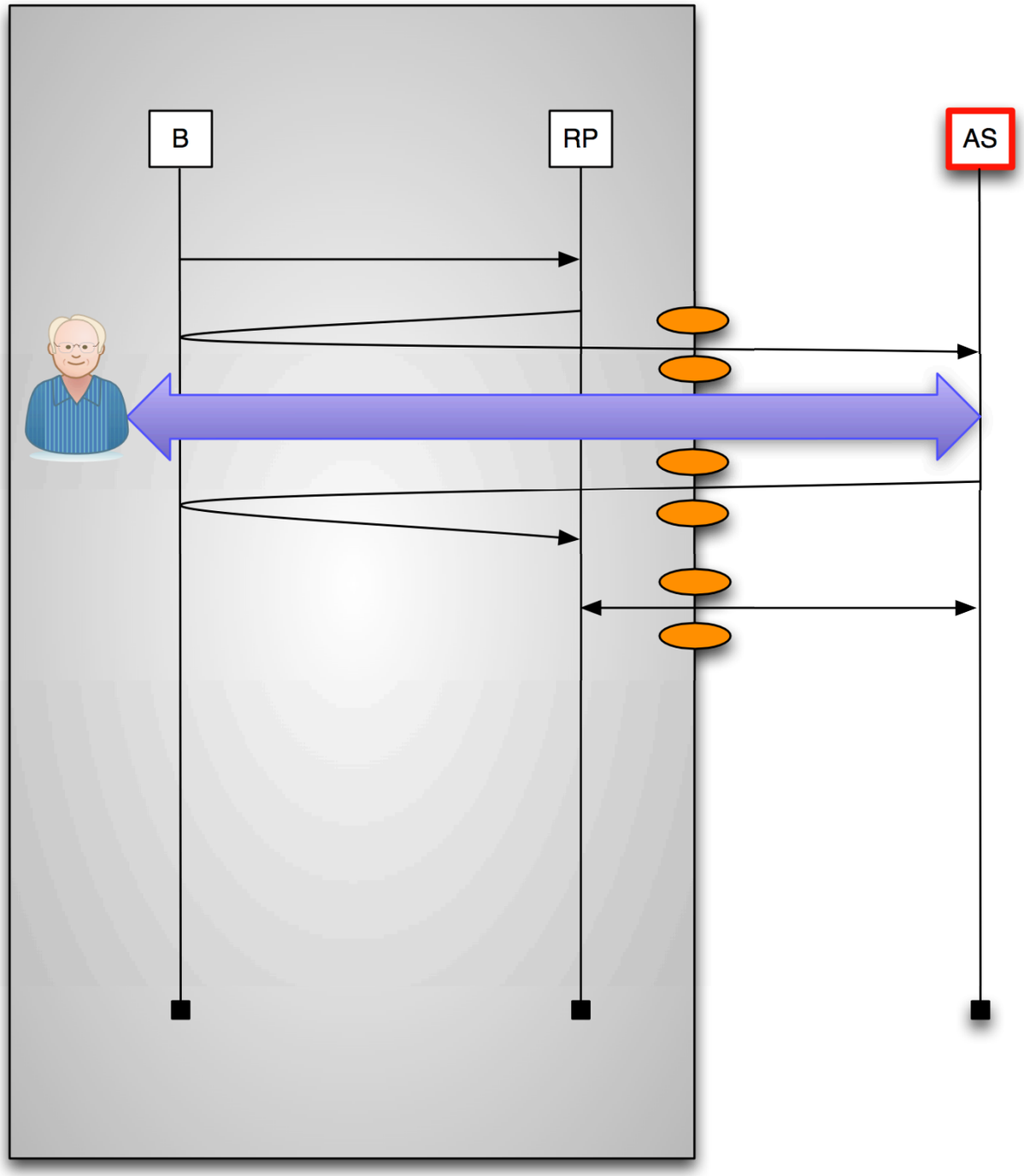
Testing the RP



test structure

- A test is a conversation with accompaning tests
- A conversation is a sequence of request-response pairs
- Each message may be preceeded/followed by a set of tests.

Conversation including tests



Tool construction

- Front-end dealing with all user interactions
  - http://openidtest.uninett.no
- Back-end doing all the protocol work
  - implemented as a script

Request message

```
class AuthorizationRequest(GetRequest):

    request = "AuthorizationRequest"

    _request_args = {"scope": ["openid"]}

    tests = {"pre": [CheckResponseType],

             "post": [CheckHTTPResponse]}
```

Response message

```
class AuthzResponse(UrlResponse):
    response = "AuthorizationResponse"
    tests = {"post": [CheckAuthorizationResponse]}
```

Defining a conversation

Phase:

"login": (AuthorizationRequestCode, AuthzResponse)

Flow:

```
'mj-01': {
        "name": 'Request with response_type=code',
        "sequence": [”login"],
        "endpoints": ["authorization_endpoint"],
        "depends": ['mj-00'],
    }
```

Running the script

$ oicc.py -J localhost.json -H localhost -i 'mj-01'

{"***status***": 1, "tests": [
{"**status**": 0, "message": {"subject_types_supported": ["public", "pairwise"], "userinfo_signing_alg_values_supported": ["HS512", "none", "RS256", "ES256", "HS256", "RS512", "HS384", "RS384"], "x509_url": "https://localhost:8092/static/cert.pem", "issuer": "https://localhost:8092/", "token_endpoint_auth_types_supported": ["client_secret_post", "client_secret_basic", "client_secret_jwt", "private_key_jwt"], "token_endpoint": "https://localhost:8092/token", "version": "3.0", "registration_endpoint": "https://localhost:8092/registration", "jwk_url": "https://localhost:8092/static/pub.jwk", "userinfo_encryption_alg_values_supported": ["RSA1_5", "RSA-OAEP"], "scopes_supported": ["openid"], "token_endpoint_auth_methods_supported": ["client_secret_basic"], "userinfo_encryption_enc_values_supported": ["A128CBC+HS256", "A256CBC+HS512", "A256GCM"], "id_token_signing_alg_values_supported": ["HS512", "none", "RS256", "ES256", "HS256", "RS512", "HS384", "RS384"], "request_object_encryption_enc_values_supported": ["A128CBC+HS256", "A256CBC+HS512", "A256GCM"], "id_token_encryption_enc_values_supported": ["A128CBC+HS256", "A256CBC+HS512", "A256GCM"], "token_endpoint_auth_signing_alg_values_supported": ["HS512", "none", "RS256", "ES256", "HS256", "RS512", "HS384", "RS384"], "userinfo_endpoint": "https://localhost:8092/userinfo", "request_object_signing_alg_values_supported": ["HS512", "none", "RS256", "ES256", "HS256", "RS512", "HS384", "RS384"], "request_object_encryption_alg_values_supported": ["RSA1_5", "RSA-OAEP"], "response_types_supported": ["code", "token", "id_token", "code token", "code id_token", "token id_token", "code token id_token"], "id_token_encryption_alg_values_supported": ["RSA1_5", "RSA-OAEP"], "authorization_endpoint": "https://localhost:8092/authorization"}, "id": "check", "name": "Provider Configuration Response"},
{"**status**": 1, "url": "https://localhost:8092/registration", "id": "check-http-response", "name": "Checks that the HTTP response status is within the 200 or 300 range"},
{"**status**": 0, "message": "{\"redirect_uris\": \"https://localhost/authz_cb\", \"jwk_url\": \"http://localhost:8090/exports/jwk.json\", \"application_type\": \"web\", \"expires_at\": 1362967043, \"registration_access_token\": \"4zM94vWsK7IwwKWLtCUYigzN59AztSUp\", \"client_id\": \"jPCe0ep54rsk\", \"client_secret\": \"23cd1aa860c3c26df2f2e9fdb97b1868041d97e8213a33828f1d089f\", \"x509_url\": \"http://localhost:8090/exports/cert.pem\"}", "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, "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"},
{"**status**": 1, "url": "https://localhost:8092/authorization?scope=openid&state=STATE0&redirect_uri=https%3A%2F%2Flocalhost%2Fauthz_cb&response_type=code&client_id=jPCe0ep54rsk", "id": "check-http-response", "name": "Checks that the HTTP response status is within the 200 or 300 range"},
{"**status**": 1, "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."}],
"id": "mj-01"}

On error - tracelog

0.001245 EXPORT

0.007399 Started key provider

1.009079 <-- FUNCTION: discover

1.009117 <-- ARGS: {'location': '', 'issuer': u'https://localhost:8092/'}

1.070308 --> URL: https://localhost:8092/registration

1.070313 --> BODY: application_type=web&operation=register&redirect_uris=https%3A%2F%2Flocalhost%2Fauthz_cb&jwk_url=http%3A%2F%2Flocalhost%3A8090%2Fexports%2Fjwk.json&x509_url=http%3A%2F%2Flocalhost%3A8090%2Fexports%2Fcert.pem

1.070322 --> HEADERS: {'content-type': 'application/x-www-form-urlencoded'}

1.107948 <-- RESPONSE: <Response [200]>

1.107998 <-- CONTENT: {"redirect_uris": "https://localhost/authz_cb", "jwk_url": "http://localhost:8090/exports/jwk.json", "application_type": "web", "expires_at": 1362967547, "registration_access_token": "kFTcq4fU8AvBrQlbAwTWXLMOKoh4L3eo", "client_id": "lvbpWJzwalGC", "client_secret": "8b28e47b2ef49378630720e5f6dddfb30d3bdea675cf9e033aad1e71", "x509_url": "http://localhost:8090/exports/cert.pem"}

1.108016 <-- COOKIES: <<class 'requests.cookies.RequestsCookieJar'>[]>

1.108713 [RegistrationResponse]: {'redirect_uris': u'https://localhost/authz_cb', 'jwk_url': u'http://localhost:8090/exports/jwk.json', 'application_type': u'web', 'expires_at': 1362967547, 'registration_access_token': u'kFTcq4fU8AvBrQlbAwTWXLMOKoh4L3eo', 'client_id': u'lvbpWJzwalGC', 'client_secret': u'8b28e47b2ef49378630720e5f6dddfb30d3bdea675cf9e033aad1e71', 'x509_url': u'http://localhost:8090/exports/cert.pem'}

1.108942 --> URL: https://localhost:8092/authorization?scope=openid&state=STATE0&redirect_uri=https%3A%2F%2Flocalhost%2Fauthz_cb&response_type=code&client_id=lvbpWJzwalGC

1.108947 --> BODY: None

1.137047 <-- RESPONSE: <Response [200]>

Running all tests

$ oic_flow_tests.py -H lingon.ladok.umu.se xenosmilus2

+ (oic-verify)Special flow used to find necessary user interactions - OK
+ (oic-discovery)Provider configuration discovery - OK
+ (mj-00)Client registration Request - OK
+ (mj-01)Request with response_type=code - OK
+ (mj-02)Request with response_type=token - OK
+ (mj-51)Login no nonce - OK
+ (oic-token-userinfo)Implicit flow and Userinfo request - OK
+ (oic-token-userinfo_bb)Implicit flow, UserInfo request using POST and bearer body
  authentication - OK
+ (mj-03)Request with response_type=id_token - OK
+ (mj-04)Request with response_type=code token - OK
+ (oic-code+token-token)Flow with response_type='code token' - OK
+ (oic-code+token-userinfo)Flow with response_type='code token' and Userinfo request - OK
+ (mj-05)Request with response_type=code id_token - OK
+ (oic-code+idtoken-token)Flow with response_type='code idtoken' - OK
+ (oic-code+idtoken-token-userinfo)Flow with response_type='code idtoken' and Userinfo request - OK
+ (mj-06)Request with response_type=id_token token - OK
+ (oic-idtoken+token-userinfo)Flow with response_type='token idtoken' and Userinfo request - OK
+ (mj-07)Request with response_type=code id_token token - OK
+ (oic-code+idtoken+token-token)Flow with response_type='code token idtoken' - OK
+ (oic-code+idtoken+token-token-userinfo)Flow with response_type='code idtoken token'
  grab a second token using the code and then do a Userinfo request - OK

Questions/Comments

welcome !