Account Porting

draft-account-porting-01

OpenID MODRNA / GSMA CPAS workshop; 27–28 Sept 2016

James Manger (Telstra)

Co-authors of spec: James, Torsten, Arne

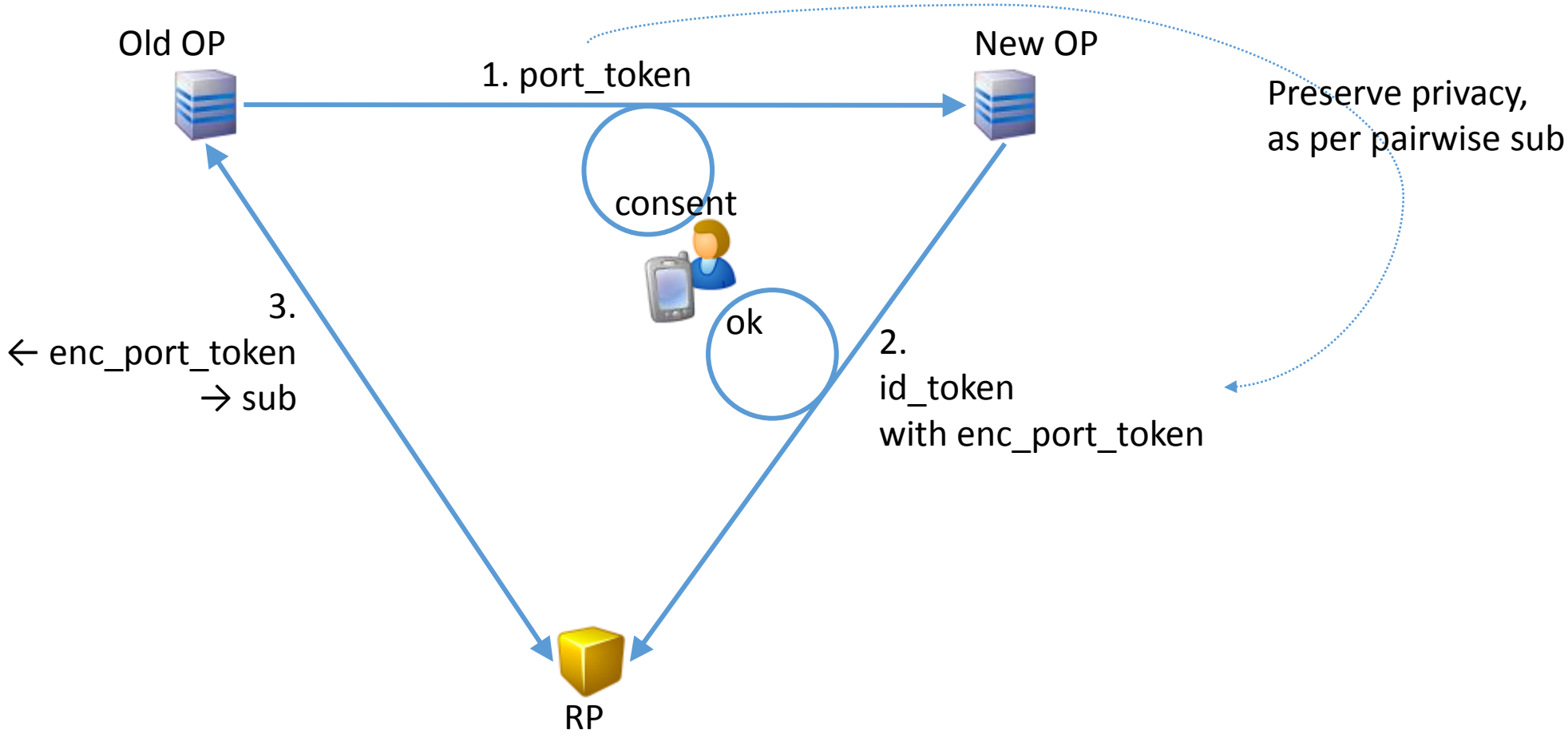
Mobile Connect should keep working when a user ports to another MNO

- Aim: For Relying Parties (RPs, aka Service Providers) to be able to ***automatically*** recognize the same user signing-in before & after they port to a new OpenID Connect Provider (OP, aka MNO)
- Specify a porting mechanism that can work for any OpenID Connect Providers, not just MNOs doing Mobile Connect
- The basic idea:
  - A user logs in to both OPs (Old & New) in the same session, linking the two
  - RPs get evidence of that linking, authenticated by both Old & New OPs

draft…01

- Step 0: setup
  - Old OP metadata lists ‘Porting data’ and ‘Porting check’ APIs; and encryption algorithms
  - New OP has client_id/secret to call ‘Port data’ API
  - RPs registered with OPs
- Step 1: port
  - New OP calls Old OP’s ‘Port data’ API → token; with OAuth 2.0 flow for user consent
- Step 2: sign-in to RP
  - OpenID Connect from New OP to RP
  - id_token has New OP’s ‘sub’ for user, plus lists Old OP with per-RP token (encrypted token)
- Step 3: confirm port
  - If RP doesn’t recognize {New OP/sub}; call Old OP’s ‘Port check API’ → Old OP’s ‘sub’
  - Repeat if still not recognized and Old OP lists an even older OP
  - Link {New OP/sub} to existing {Old OP/sub}

draft…01†



† Changes since draft…01: renamed port_token_plain/port_token to port_token/enc_port_token

Step 0: metadata

→ GET https://**OldOP**.example.net/**.well-known**/openid-configuration

←

```
{
                "issuer": "https://oldop.example.net/",
 "authorization_endpoint": "https://oldop.example.net/connect/authorize",
        "token_endpoint": "https://oldop.example.net/connect/token",
     "port_data_endpoint": "https://oldop.example.net/connect/port_data",
    "port_check_endpoint": "https://oldop.example.net/connect/port_check",
"port_enc_algs_supported": [ "A256GCM", "A256CBC-HS512" ],
  …
}
```

Step 1: Porting data API

OAuth 2.0 flow

```
→ HTTP/1.1 303 See other
  Location: https://OldOP.example.net/connect/authorize?scope=porting_data&…
  …
→ GET https://OldOP.example.net/connect/port_data/me
  Authorization: Bearer E3yyDiR5_i8CFCVDo3h8T5qgKpAdu8XkGZBv81vn428
←
  {"port_token": "3O9YHawMDXLpKb-FVjQ1_qSS9R9wbwb0TWbUxLvqAAI"}
```

Opaque to others (New OP & RPs).
Old OP can encode state of user port
(eg sub, New OP… as JWE)

Step 2: OpenID Connect login to RP

← HTTP/1.1 303 See other
Location: https://**NewOP**.example.net/connect/**authorize**?scope=**openid**&…
…
click Ok

Eg Mobile Connect
confirmation on phone

→ POST https://**NewOP**.example.net/connect/**token**?…
Authorization: Basic *B64*(rp_client_id:rp_client_secret)

← {
"access_token": "XQ2-FHA0q7gqExq7vfRbopmkg_QPSuS4FFoigG4nZvU",
"**id_token**": *JWT*({
"iss": "https://**NewOP**.example.net/",
"sub": "JRrr09BzhZ6BJ9t0yD8DzyZjg7ziB3a40jeoUvZkUgw",
"**aka**": {
"iss": "https://**OldOP**.example.net/",
“enc_port_token": "eyJ0eX…3JnIn0..48V…U3b.GYV….qfc…"
},
…})}

pairwise

Actually
JWE(“309…AAI”) per RP,
but opaque to RP

Step 3: Porting check API

No direct RP authentication

→ GET https://**OldOP**.example.net/connect/**port_check**?
iss=https://NewOP.example.net/&
sector_id=rp.example.org&
enc_port_token=eyJ0eX…3JnIn0..48v…U3b.GYV….qfc…

- Old OP:
  - Decrypts enc_port_token
  - Confirms it issued port_token
  - Matches New OP (decryption key = iss parameter = port destination)
  - Matches sector_id (query parameter = JWE parameter)

Match iss, sector_id

← {
"sub": "kY6N64H86MBTho9I-JlRnpvaLOwAb5m3yYta6pq1VFM",
"aka": {
"iss": "https://**anotherop**.example.net/",
"enc_port_token": "QSfmp2V30cNqKShhyOjQEwDaQzLNhji5w5nD-EZq3ok"
}}

For the corner case of multiple port before an RP visit

Step 3b: Encrypting port_token

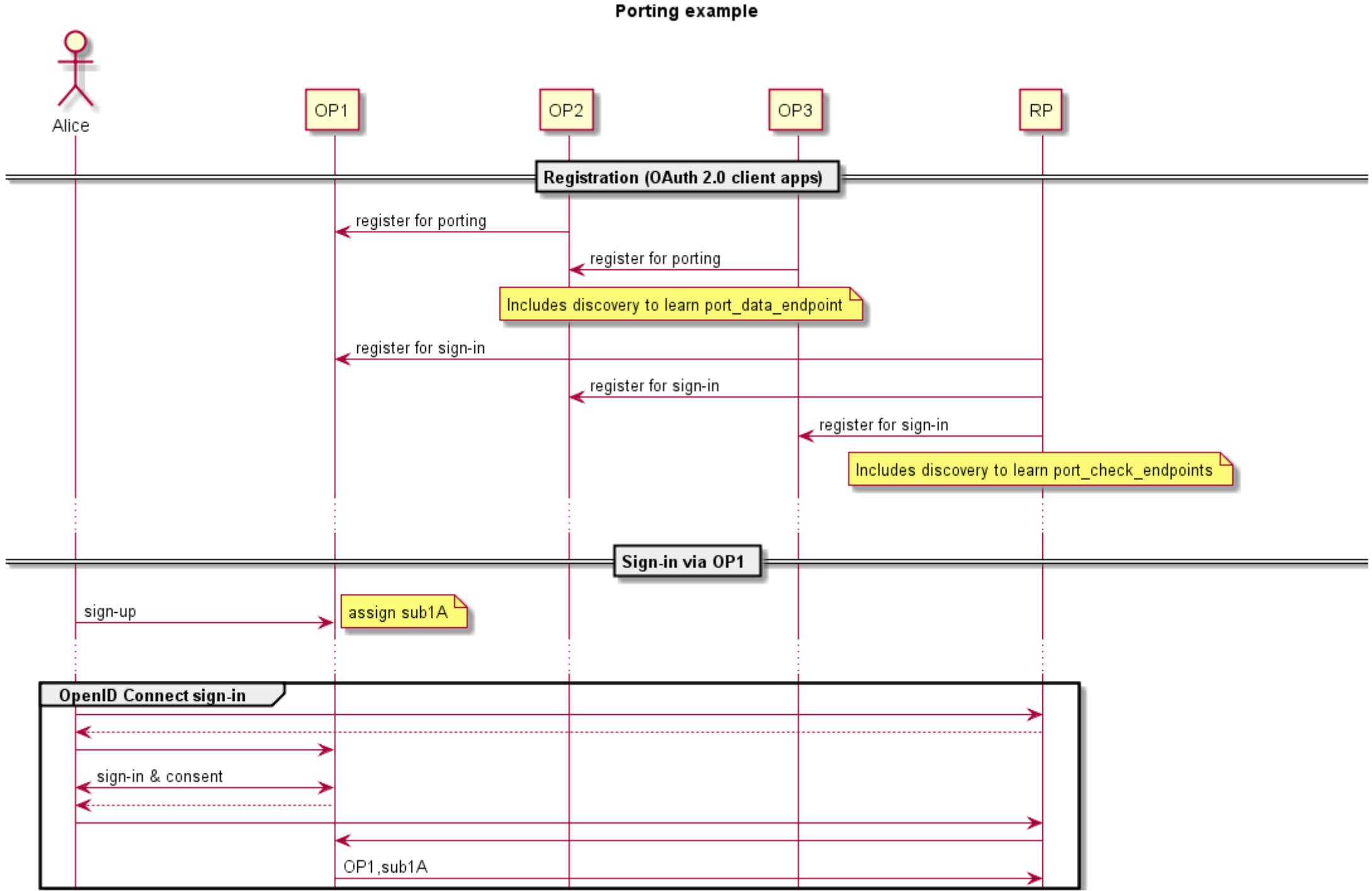
- JSON Web Encryption (JWE)
- Direct encryption with client_secret shared by Old OP and New OP
- JWE protected header:

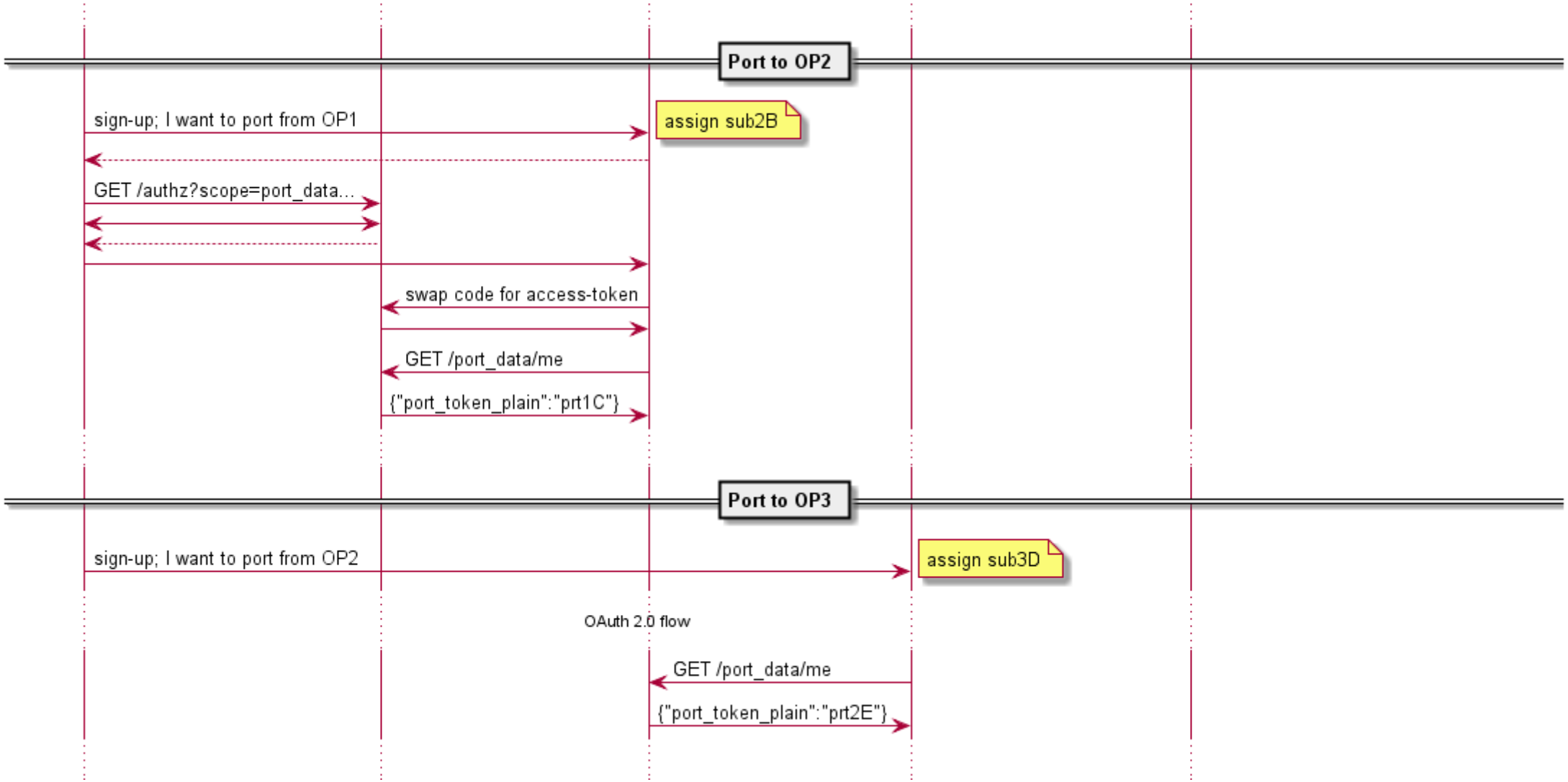
```
{
        "typ": "oidc-porting",
        "alg": "dir",
        "enc": "A256GCM",
        "kid": "s6BhdRkqt3",
  "sector_id": "rp.example.org"
}
```

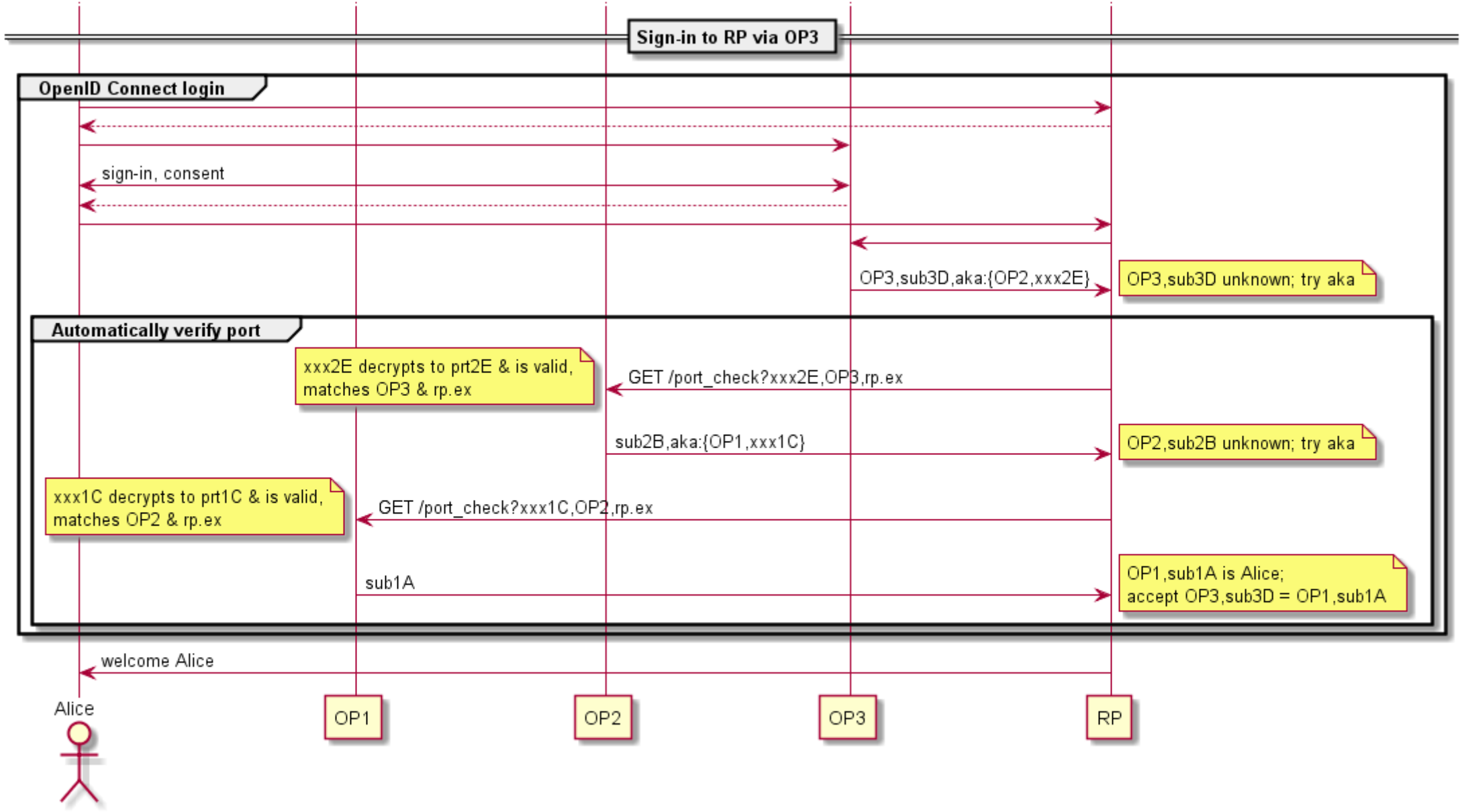
- AEAD algorithm chosen from Old OP metadata
- Secret key = truncate(SHA2(*client_secret*)), as per OpenID Connect Core [§10.2]
  - Key id = *client_id*
- Plaintext = port_token

MUST use fresh encryption for each sector_id

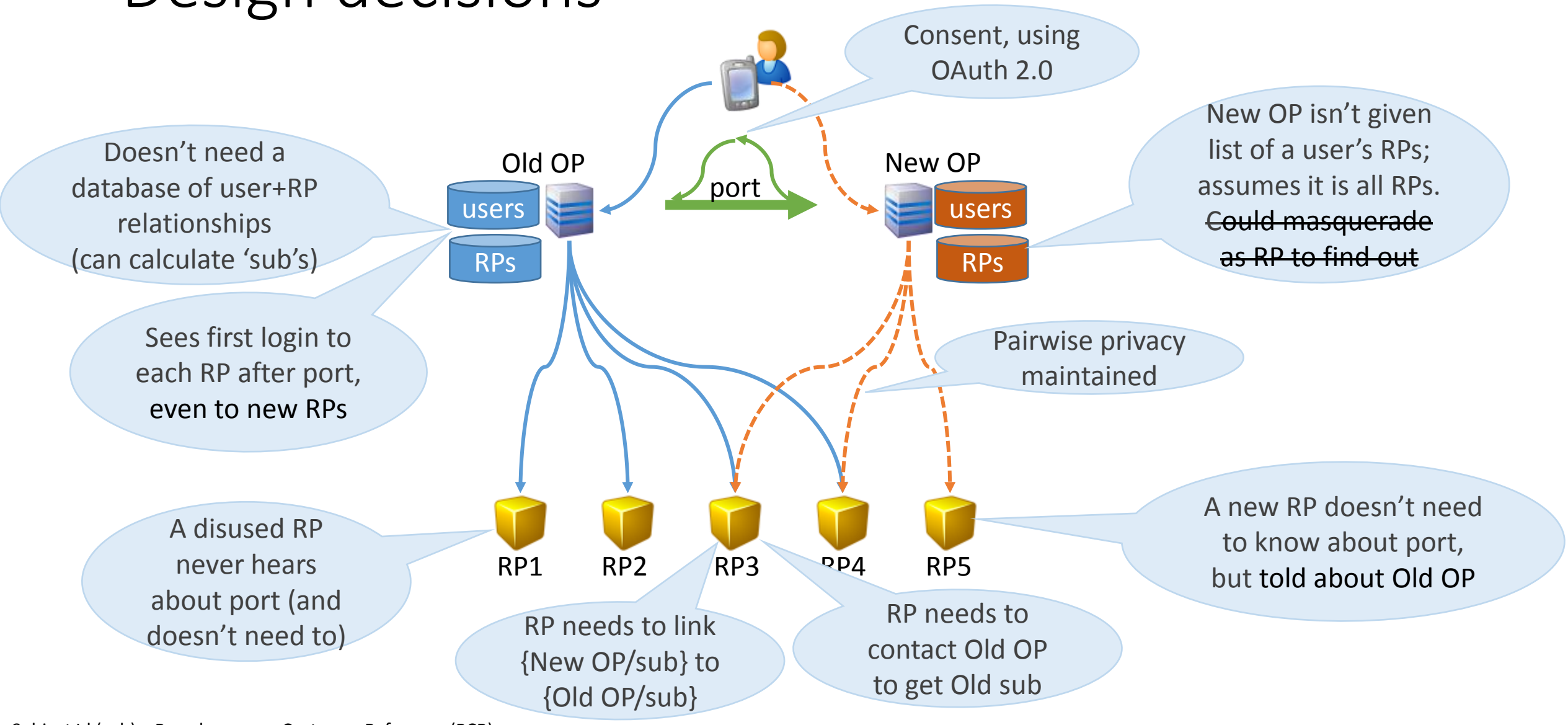
Drop if RP authenticates to Porting check API


Porting example
Alice
OP1
OP2
OP3
RP
Registration (OAuth 2.0 client apps)
register for porting
register for porting
Includes discovery to learn port_data_endpoint
register for sign-in
register for sign-in
register for sign-in
Includes discovery to learn port_check_endpoints
Sign-in via OP1
sign-up
assign sub1A
OpenID Connect sign-in
sign-in & consent
OP1,sub1A


Port to OP2
sign-up; I want to port from OP1
assign sub2B
GET /authz?scope=port_data...
swap code for access-token
GET /port_data/me
{"port_token_plain":"prt1C"}
Port to OP3
sign-up; I want to port from OP2
assign sub3D
OAuth 2.0 flow
GET /port_data/me
{"port_token_plain":"prt2E"}


Sign-in to RP via OP3
OpenID Connect login
sign-in, consent
OP3,sub3D,aka:{OP2,xxx2E}
OP3,sub3D unknown; try aka
Automatically verify port
xxx2E decrypts to prt2E & is valid,
matches OP3 & rp.ex
GET /port_check?xxx2E,OP3,rp.ex
sub2B,aka:{OP1,xxx1C}
OP2,sub2B unknown; try aka
xxx1C decrypts to prt1C & is valid,
matches OP2 & rp.ex
GET /port_check?xxx1C,OP2,rp.ex
sub1A
OP1,sub1A is Alice;
accept OP3,sub3D = OP1,sub1A
welcome Alice
Alice
OP1
OP2
OP3
RP
<-- dotted lines are redirects

Design decisions



Subject Id (sub) = Pseudonymous Customer Reference (PCR)

Alternative #1: Authenticate RP ✓

- Instead of Old OP recognizing the same sector_id for RP as New OP, the RP authenticates to the Old OP's Porting check API
- Step 3: Porting check API

→ POST https://**OldOP**.example.net/connect/**token**?grant_type=client_credentials&…
Authorization: Basic *B64*(rp_client_id:rp_client_secret)

→ GET https://**OldOP**.example.net/connect/**port_check**?
iss=https://**NewOP**.example.net/&
enc_port_token=eyJ0eX…3JnIn0..48V…U3b.GYV….qfc…
Authorization: Bearer v8BmvE6ZxKukm_3_rGRbvTOoynXP1SBlWkGAxfA9Fwo

Omit?

- Old OP:
  - Decrypts enc_port_token; confirms port_token valid for a port to New OP; match iss to kid
  - Return "sub" appropriate to the authenticated RP

← { "sub": "kY6N64H86MBTho9I-JlRnpvaLOwAb5m3yYta6pq1VFM" }

- ✓ Old OP and New OP don't have to agree on sector_id
- ✗ Extra effort for RP to authenticate; extra OAuth 2.0 flow

Alternative #2: Signed porting statements

- draft-account-**migration**-02
- Old OP maintains a DB of user+RP relationships
- Creates a signed message for each RP/sector_id vouching to the port
- New OP gets signed messages from Old OP (with OAuth 2.0 consent)
- New OP includes an RP's signed message in id_token
- ✓ RP does not need to call Old OP (other than to get public key)
- ✗ Old OP needs long-term signing keys
- ✗ OPs need user+RP DB; cannot just calculate 'sub' on demand

Alternative #3: Same “sub”

- Early 2016 version of Mobile Connect Lifecycle Processes
- New OP gets Subject Ids (subs) from Old OP; reuses same subs
- RP should confirm port with Old OP
- ✓ id_token unchanged
- ✗ Risks from confusing subs from OPs in scheme vs other OPs
- ✗ OPs need user+RP DB; cannot calculate ‘sub’

Changes for draft…-02 — to be agreed

- ❑Rename port_token_plain/port_token to port_token/enc_port_token, porting_data scope to port_data
- ❑Drop no-encryption-needed option from Porting data API
- ❑Authenticate RP to Old OP’s Porting check API
  - ❑Standard OAuth 2.0 grant_type=client_credentials flow first
- ❑Add privacy considerations
- ❑…
- ❑…
- ❑…

Resources

- Mobile Operator Discovery, Registration & Authentication (MODRNA) OpenID working group
  - Web page: http://openid.net/wg/mobile/
  - Bitbucket repo: https://bitbucket.org/openid/mobile/
  - Email list: openid-specs-mobile-profile@lists.openid.net
  - Email archive: http://lists.openid.net/pipermail/openid-specs-mobile-profile/
- draft-account-porting
  - Web: https://id.cto.telstra.com/2016/openid/draft-account-porting.html
  - Bitbucket repo: draft-account-porting.{html|.eg.png}
  - Earlier drafts named draft-account-migration
- GSMA CPAS (Mobile Connect technical group)
  - Basecamp: https://basecamp.com/2588644/projects/9050876