Initial proposal

Mobile Profile

For

Connect

Version: 0.1.5

2014-07-24

Jörg Connotte (Deutsche Telekom AG)
Sebastian Ebling (Deutsche Telekom AG)
Torsten Lodderstedt (Deutsche Telekom AG)

Deutsche Telekom AG
Products & Innovation
T-Online Allee 1
64295 Darmstadt

Index

1 INTRODUCTION 3

1.1 REVISION HISTORY 3

1.2 REFERENCES 3

2 INTRODUCTION 4

3 CHALLENGES AND PROPOSAL 5

3.1 DISCOVERY 5

3.2 DYNAMIC CLIENT REGISTRATION 5

3.3 AUTHENTICATION 6

4 NON NORMATIVE EXAMPLE FLOW 7

1 Introduction

1.1 Revision History

Version	Date	Author	Revision
0.1.0		Jörg Connotte	Initial
0.1.1	26.06.2014	Jörg Connotte	Include Review from Torsten
0.1.2	11.07.2014	Jörg Connotte	Translated, merged challenge an proposal sections
0.1.3	16.07.2014	Jörg Connotte	Sequence diagram documented
0.1.4	19.07.2014	Torsten Lodderstedt	Extended documentation of sequence diagram.
0.1.5	21.07.2014	Jörg Connotte	Switch to msisdn as resource in example flow

Table 1: Revision History

1.2 References

	Title	File Name / URL	Version
[1]	Mobile Profile Working Group	http://openid.net/wg/mobile/	
[2]	OpenID Connect Core Specification	http://openid.net/specs/openid-connect-core-1_0.html	V1.0
[3]	OpenID Connect Discovery	http://openid.net/specs/openid-connect-discovery-1_0.html	V1.0
[4]	OpenID Dynamic Client Registration	http://openid.net/specs/openid-connect-registration-1_0.html	V1.0
[5]	OAuth2.0 Dynamic Client Registration	http://tools.ietf.org/html/draft-ietf-oauth-dyn-reg-18	Draft 18
[6]	Level of Assurance ISO/IEC 29115	https://www.oasis-open.org/committees/download.php/44751/285-17Attach1.pdf	12/2011

Table 2: References

2 Introduction

From the Mobile Profiles for OpenID Connect WG charter:

> *Developing a profile of OIDC intended to be appropriate for use by mobile network operators (MNOs) providing identity services to RP's and for RP's in consuming those services as well as any other party wishing to be interoperable with this profile.*
>
> *Identify and make recommendations for additional standard items.*

The following assumptions are the basis for this proposal

- The Mobile Profile is based on OpenID Connect 1.0 (including Core, Discovery, and Dynamic Client Registration)
- The Mobile Profile must be easy to implement for developers.
- Implementations of the Mobile Profile can make use of existing ecosystems, esp. those based on OpenID Connect 1.0, already in place at mobile network operators or central services.
- The Mobile Profile can be implemented by extending existing OpenID Connect service implementations.
- Implementations of the mobile profile can be leveraged in other contexts (e.g. other OpenID Connect profiles) as well.

3 Challenges and Proposal

3.1 Discovery

[3] section 2.1 specifies a way to normalize a user identifier to derive a resource and especially a host for OpenID Provider Issuer Discovery. While this works well for identifiers as email addresses and urls it does not work for MNO identifiers. In a mobile environment, MSISDN's or ip-addresses are typical resources identifying a user. The structure of those identifiers does not allow for an algorithmic normalization. Thus it will be necessary to have a specific service to perform the normalization.

The Mobile Profile will specify the interface of this service. Part of the interface specification will be the identification of possible user input identifier types relevant in the Mobile Connect scenario.

Remark:

Some of those user input identifier types may not identify a single user/device but may be sufficient to identify the right mobile operator. An example for this is the MNC/MCC tuple (mobile network code/ mobile contry code) or the IP address range.

	'classical' environment	Mobile network
User input	john@example.org	+1 555 4567890
Resource	acct:john@example.org	msisdn:15554567890?
Host	example.org	?? – *cannot be derived from user input*

Table 3: From user input to host in mobile environment

3.2 Dynamic client registration

The profile shall allow a RP to access ID services of multiple MNOs (e.g. in a market) while not requiring this RP to (manually) register with all of them. The RP shall register to a certain MNO or another entity MNOs delegated this task to once and obtain the right to access the desired MNOs' ID services. To enable this autonomy of relying parties and OpenID Providers (the MNO's) dynamic client registration must be possible. It is however necessary to verify the level of trustworthiness (and potentially entitlement) of an RP and enable the OP to approve or block RP's accordingly. Thus it will be necessary to establish some measures of control to verify the legitimacy of the RP.

For the Mobile Profile we propose to extend [3][4] with Software Statements as specified in [5].

It is beyond the scope of the Mobile Profile how the RP will obtain original legitimation. It is assumed that some trustworthy party validates the RP and issues the software statement to the RP.

It is also beyond the scope of the Mobile Profile how an OP validates the software statement.

It could be in scope to profile the use of software statements to (a) certain signature algorithm(s) and meta data.

3.3 Authentication

RPs are keen to use high quality authentication methods which can be provided by the MNOs. However a RP must be able to describe its demands for an authentication request and it must be able to do this interoperable.

The Mobile Profile will specify how RP's request a certain level of assurance for the authentication. We propose to use the definition of Level of Assurance as specified in [6] as a starting point and define respective normative ACR values.

4 Non normative example flow

The following example flow illustrates the interaction between the different parts of OpenID Connect including discovery, client registration, and authentication under the specific challenges of the mobile szenario. Note: The complete flow is typically performed only once per RP and device as the discovery result and client credentials are stored by the RP for sub-sequent authentication requests.

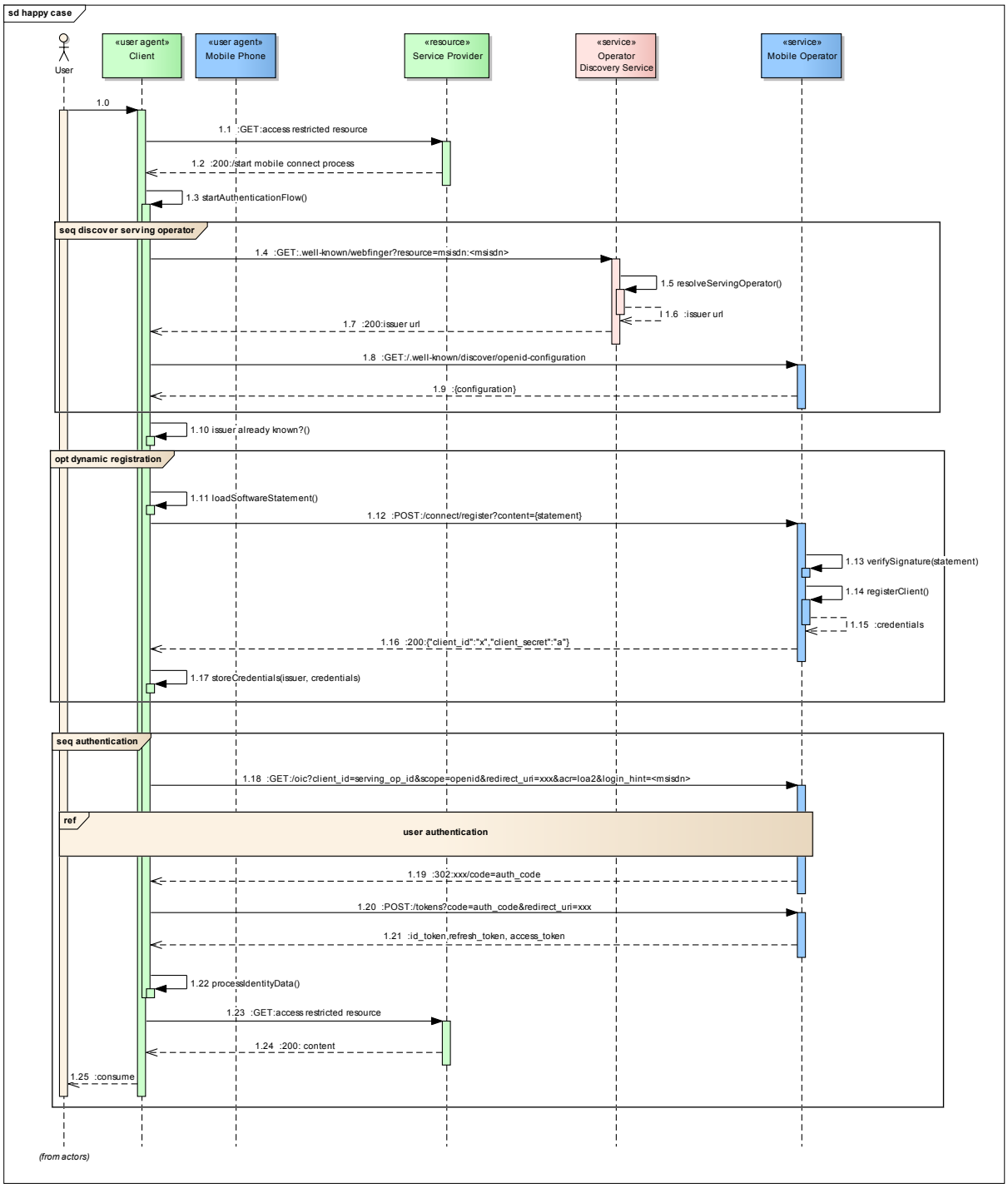

Figure 1: Example for authentication using mobile channel

The following tasks have to be performed (one of them optionally) to authenticate a user with an operator specific OpenId Provider.

- `seq discover serving operator`: Resolve the right OpenID Provider and its metadata using [3].
- `opt dynamic registration`: If necessary register the client with the OpenID Provider using software statements as proposed in [5].
- `seq authentication`: perform authentication using [2].

The steps in detail:

Step 1.1: The client (e.g. a browser) wants to access a protected resource at the service provider

Step 1.2ff: The service provider initiates the login process

Step 1.4: To discover the appropriate operator to log in the user, a webfinger request is sent to a dedicated mobile aware web finger service in order to derive the issuer URL from available information, such as MSISDN, MNC/MCC, or MSISDN.

Note: this document does not specify how the RP obtains the URL of the dedicated service. One option would be to obtain it during development from a MNO and configure it at the RP.

```
GET /.well-known/webfinger
  &resource=msisdn:15554567890
  &rel=http%3A%2F%2Fopenid.net%2Fspecs%2Fconnect%2F1.0%2Fissuer
  HTTP/1.1

Host:discovery.example.net
```

Listing 1: Mobile Profile Issuer Discovery Request

In this example, the RP is a web application, which asked the user to enter his msisdn. It sends this msisdn as resource to the web finger service. The structure of this request is similar to [3] section 2. We assume the mobile profile will define certain new resource types, such as MSISDN and ip-address.

Step 1.7: After resolving the operator, the web finger service will respond with a web finger response carrying a OpenID Connect Issuer URL. So the rest of the discovery process is performed as described in [3].

```
HTTP/1.1 200 OK
  Content-Type: application/jrd+json
  {
   "subject": "msisdn  :15554567890",
   "links":
    [
     {
      "rel": "http://openid.net/specs/connect/1.0/issuer",
      "href": "https://serving.operator.com"
     }
    ]
  }
```

Listing 2: Mobile Profile Issuer Discovery Response

Step 1.8 – Step 1.9: The client requests the OpenID Connect configuration of the appropriate operator using the OpenID Provider Configuration Request as specified in [3] section 4.

```
HTTP/1.1 200 OK
  Content-Type: application/jrd+json
```

```
  {
   "issuer": "https://serving.operator.com",
   "response_types_supported":["code"],
   "authorization_endpoint": "https://serving.operator.com/oauth2/auth",
   "token_endpoint": "https://serving.operator.com/oauth2/token",
   "userinfo_endpoint": "https://serving.operator.com/oidc/userinfo",
   ...
  }
```

Listing 3: Mobile Profile Issuer Discovery Response

Step 1.10: The client checks if the OpenID Provider information is already known to it (based on the issuer URL) and if it already possesses credentials to access the OpenID Connect endpoints.

Step 1.11ff: If the OpenID Provider is not already known to the client, dynamic client registration is utililized to register with the operator. For this the client was issued a software statement (at development time) which proves the validity of the client and its entitlement to access the operators ID service.

Step 1.12: The client posts a register request as specified in [4] section 3.1 using the software statement as specified in [5] section 3.1.

```
POST /register HTTP/1.1
  Content-Type: application/json
  Accept: application/json
  Host: serving.operator.com
  {
    ...
    "redirect_uris":[
      "https://client.example.org/callback",
      "https://client.example.org/callback2"
    ],
    "software_statement":"eyJhbGciOiJFUzI1NiJ9.
          eyJpc3Mi[...omitted for brevity...].
          J9l-ZhwP[...omitted for brevity...]",
    "scope":"openid …",
    "example_extension_parameter":"example_value"
  }
```

Listing 4: Mobile Profile register request

Step 1.13 – Step 1.15: The operator's OP checks the validity of the registration request using the software statement and issues new credentials for this client.

Step 1.16: The operator responds as specified in [4] section 3.2.

```
HTTP/1.1 201 Created
  Content-Type: application/json
  Cache-Control: no-store
  Pragma: no-cache
  {
   "client_id": "s6BhdRkqt3",
   "client_secret":
     "ZJYCqe3GGRvdrudKyZS0XhGv_Z45DuKhCUk0gBR1vZk",
     …
  }
```

Listing 5: Mobile Profile register response

Step 1.17: The client stores the newly acquired credential information in relation to the issuer URL provided in Step 1.7.

Step 1.18 – Step 1.21: The client follows through with the standard OpenID Connect login flow a specified in [2] using the `authorization_endpoint` and `token_endpoint` provided in Step 1.9. As part of the authentication request, the client may specify a certain expected level of assurance. In the authentication response, the OP provides the RP with a user id ("sub" claim in ID token) as well as information about the level of assurance ("acr" claim).

Step 1.22: The client directly 'logs in' the user. Alternatively, if the RP detects that the user is visiting it first time, she is registered.

Step 1.23 – Step 1.24: The client now knows who the user is and can access the restricted resource at the service provider accordingly.

Note: Although this flow demonstrates an authentication process utilizing the authorization code flow, the mobile profile is intended to be compatible will any other OAuth grant type, e.g. refresh token.