# IMPLICIT DISCOVERY

As one of the key success of the Identity System is to be offered to the Service Providers as a single integration point, it is necessary somehow discover what is the provider to the user belongs to so he can be authenticated, this provider will be called the Home MNO.

This approach is based on hide the discovery process to the Service Providers in order to facilitate them to be connected to the Mobile Connect System. To achieve this, the Discovery system exposes the Authorization Endpoint directly and the home MNO will be discovered in the middle of the authentication process.

The whole OpenId Connect flow happens into the MNO infrastructure whereas the Discovery system is only used to discover the appropriate MNO to which the user belongs to.

The rational for this model is to avoid the Service Providers to implement any discovery system, as the same time it allows to use the Authentication Systems already deployed MNOs. This approach takes advantage of the original request to authenticate the user (sent by the Service Provider) to route it to the user’s home operator and dynamically discover the token and user info endpoints.

**Note**: it needs to extend the OpenID Connect protocol to allow sending the the UserInfo and Token endpoints in the response along with the auth\_code.



Step 1: the user makes a login request to the App Service.

Step 2: the App Service returns a 302 redirection indicating the AuthZ endpoint in the location url along with other parameters like the openid scope [more scopes], redirect\_url, client\_id, etc. The Discovery system receives that request in the AuthZ endpoint and shows a page to the user to introduce its MSISDN or choose the operator (in case of the MSISDN or MCC and MNC could be extracted from the request, this step would not be necessary)

Step 3: after the user introduce the MSISDN, the Prompt Web Server asks to the PathFinder system for the home MNO.

Step 4: the Prompt Web Server asks to the Service Directory for the Authorization endpoint corresponding to the MNO obtained in the previous step.

**Note**: *the service directory is a system that stores the AuthN endpoints of the MNOs*.

Step 5 (OPTIONAL): the Prompt Web Server asks to the MNO for a user\_Id that represents the MSISDN (that user\_Id would be the ***”subject identifier”*** specified in OIDC). The operator must keep this related information in a database. Note that this solution can be improved caching the user\_Id in the discovery system.

Step 6: the Prompt Web Server returns a 302 redirection (as a response of the request in the step 2) indicating the AuthN endpoint to the MNO which the user belongs to along with the user\_Id (OPTIONAL) that represents the user in the MNO, and the authentication request is redirected from the browser, so the user will be authenticated in its home operator.

**Note**: *In this step the Discovery system can set some cookies in the 302 redirection to improve the UX in the forthcoming authentication requests.*

(e.g: *Set –Cookie: authN-endpoint=http://www.home-operator.com/oauth/authorize*).

(e.g: *Set –Cookie: subject=xxxxxxxxxxxx*).

Step 7: the home operator receives the authentication request and do the following:

* Checks that the App Service (client) is registered
* Validates the request and makes sure the application identified by the client\_id is able to ask for the scopes requested, etc.

Step 8 (OPTIONAL): In case of ***“subject identifier”*** obtained in step 5 is not present in the request (log-hint parameter), the MNO shows a page to the user to introduce its MSISDN (in case of the MSISDN or MCC and MNC could be extracted from the request, this step would not be necessary)

Step 9: the home MNO authenticates the user.

Step 10: the MNO AuthServer asks the user for the consent. In the response the user gives the consent (Happy Path)

Step 11: the MNO AuthServer generates an auth\_code to be exchanged for the token\_id and the access\_token later.

Step 12: the MNO sends the auth\_code to the App Service using another redirection (url Location: App Service redirect\_uri). The AuthServer sends the token endpoint and the UserInfo endpoint of the home MNO in this redirect response (***extension to OAuth 2.0 for OpenID Connect***)

**Note**: *In this step the MNO can set a cookie in the 302 redirection with the user session in order to no authenticate him again if a request is received when the session is active.*

Step 13: the App Service makes a backend to backend request to change the auth\_code for the token\_id and access\_token.

Step 14: the AuthServer MNO query the AppInfo in the app database to check if the client\_id and client\_secret received in the request match with the stored ones for this app.

Step 15: if the previous step is ok, the AuthServer MNO creates the tokens (token\_id, access\_token), store the access\_token in the Tokens DB and returns the tokens to the App Service along with the UserInfo Endpoint.

Step 16: the App Service can access to the UserInfo.