Discovery Service API Documentation

Overview

In the API Exchange Architecture, the Discovery API is used by either a Developer's Application or another Operator API inside Custom connector (XOCL) to:

1. Discover the wireless operator of an end-user
2. Get information about the endpoints of wireless operator.
3. Get a temporary client id & secret (optional) in exchange of a valid Developer application credentials. These temporary credentials would be used to make further API calls to the subscriber operator.
4. Alternate to point 3, there is a 'lite' version of Discovery that does not require developer credential and hence does not return the temporary Exchange credentials.

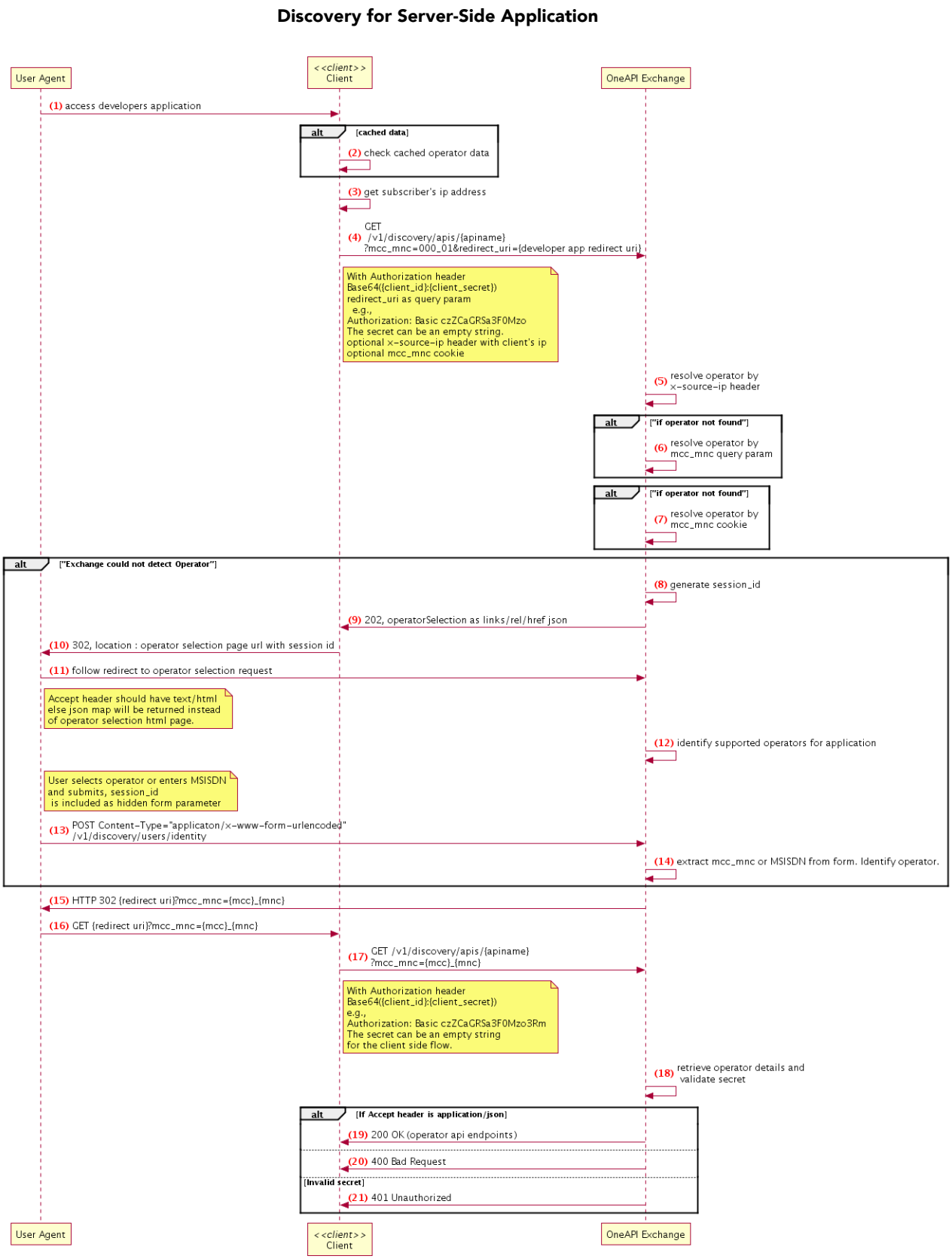
Below is the logical sequence used for identifying the Operator of an end-user (e.g., subscriber)

5. IP address of the subscriber. Discovery uses the x-source-ip HTTP header to ascertain the operator. This is possible because exchange is aware of the external facing IP ranges of the participating operators.
6. Discovery request could encapsulate the subscriber's mcc and mnc information e.g., in form of query parameters or as cookies. This information is used to determine the operator.
7. If discovery service is unable to identify the subscriber's operator from above two scenario's then subscriber will be presented with a operator selection web page. This page will allow the user to either select the operator from a list of available operators or to submit a MSISDN if the user is unable recognize their operator from the list.

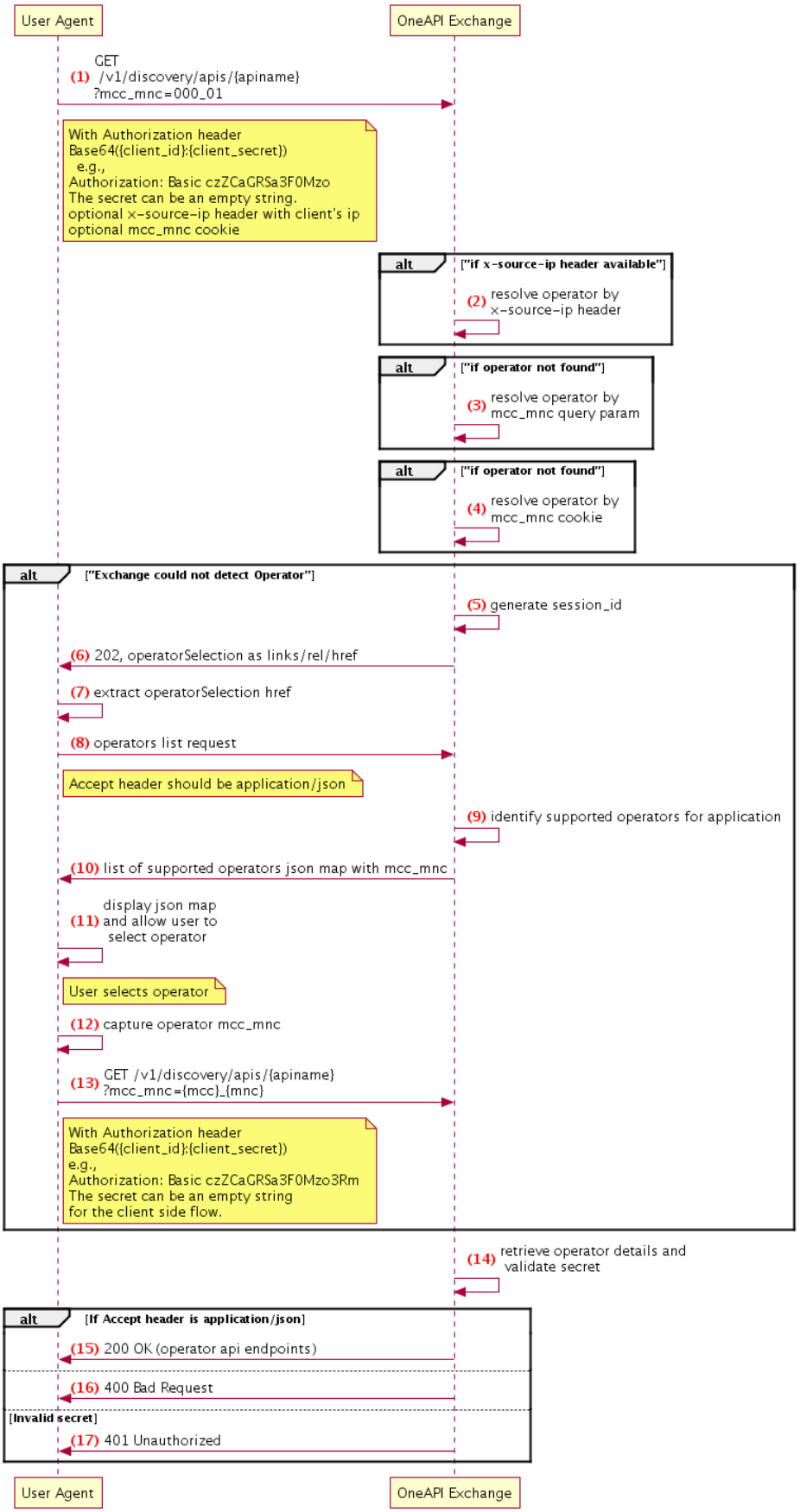
A Developer can build two kinds of applications with respect to the Discovery API.

- **Server-side** Application: Developers application would interact with developers own server side component. The operator endpoints in discovery response are parsed by developers server side components (referred as Client in sequence dig.,) and hence discovery mechanism involves redirect_uri and x-source-ip.
    - **redirect_uri** query parameters required to pass subscriber's user agent back to Developer Application's server side component.
    - **x-source-ip** header is the IP of subscriber, which can be passed to discovery for operator identification.
- **Client-side** Application: Where the application resides on the user’s device and interacts directly with Exchange there is no server side component involved. In such case the Discovery API can return the list of operators in a JSON format. The application can display how the operator selection page is displayed to the user. So in this case redirect_uri parameter is not used. Following sequence diagrams provide more details.

Sequence Diagram



Discovery for Client Side Applications



API Interface Details

Request Details

<table>
<tr><th colspan="2">Request Details</th></tr>
<tr><td>Verb</td><td>GET or POST</td></tr>
<tr><td>Path</td><td>/v1/discovery/apis/{apiname}
apiname - (optional) path variable, If present, system will return the end point details of the particular API only. apiname should be one of the supported apis and would result in 404 for any unknown apis.</td></tr>
<tr><td>Headers</td><td>
<table>
<tr><th>Name</th><th>Example Value</th><th>Comments</th></tr>
<tr><td>Authorization</td><td>Basic base64({client_id}:{client_secret})</td><td>Required
empty secret is allowed and in such cases, Discovery response will not have exchange secret. ex : Basic base64({client_id}:)</td></tr>
<tr><td>x-source-ip</td><td>221.12.10.1</td><td>IP address of the subscriber device.</td></tr>
<tr><td>User-Agent</td><td>Mozilla/5.0 (Linux; U; Android 1.1; en-gb; dream)</td><td>Optional : Note - The value is just an example</td></tr>
<tr><td>Accept</td><td>text/html, application/json</td><td>Optional : Note - The value is just an example</td></tr>
<tr><td>Accept-Language</td><td>en-us,de-at;q=0.8,en;q=0.5,ar-qa;q=0.3</td><td>Optional : Note - The value is just an example</td></tr>
<tr><td>Cookie</td><td>mcc_mnc=310_150</td><td>Mobile Country code Mobile Network code

Optional
If present then discovery happens
based on these values. Note - The value is just an example</td></tr>
</table>
</td></tr>
<tr><td>Query Parameters</td><td>
<table>
<tr><th>Name</th><th>Example Value</th><th>Description</th><th>Comments</th></tr>
<tr><td>redirect_uri</td><td>http://apigee.com</td><td>developers apps redirect url, the OneAPI Exchange redirects to this url after obtaining msisdn from user.</td><td>One of these two fields is mandatory

for discovery api</td></tr>
<tr><td>mcc_mnc</td><td>310_150</td><td></td><td>Mobile Country code
Mobile Network code

Optional
If present then discovery happens based on these values.
Note - The value is just an example</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><td>Body (Optional. Only applicable for POST)</td><td>POST is used in case of MSISDN based discovery.

<table><tr><th>Name</th><th>Example Value</th><th>Description</th><th>Comments</th></tr><tr><td>msisdn</td><td>msisdn=+94718004715</td><td>Subscribers MSISDN number</td><td>The developer application or the SDK may obtain the MSISDN from the native device APIs.</td></tr></table></td></tr>
</table>

Response Details

The Discovery API response are of two types.

1. It responds with operator API endpoint details when the operator is identified.
2. It responds with a list of operators for operator selection by subscriber when the operator is not identified. In this case finally the user selected operator's mcc and mnc is returned to the redirect URI of the originally submitted request.

Operator Identified Response

In this case the response will contain operator endpoint details in json array format. The following details will be available in the response

- **client_id:** This is the application key of the application issued temporarily by the OneAPI Exchange. Original client_id issued by Developer operator will not be shared with subscriber (other) operators. Exchange **client_id** is generated by the OneAPI Exchange is a randomly generated string.
- **client_secret:** This is optional**.** Client secret is returned only when it is supplied in the request.
- **subscriber_operator:** A String ID representing the Subscriber's operator (Serving Operator)
- **currency**: ISO 4217 currency code of the operator
- **country**: ISO 3166 country code of the operator
- **apis:** JSON object list of the supported APIs. The JSON for APIs contain the endpoint details.
- **ttl**: DNS style TTL. The developer app can keep the discovery response in cache for defined ttl.
- **subscriber_id**[1] : This is optional field. The subscriber id field is returned only if the operator has a public key set up in the exchange to encrypt MSISDNs and in the Discovery request a MSISDN was available.

[1] In order to prevent the mobile subscribers from specifying their MSISDN twice, once for the discovery service and then potentially again for identifying with the Serving Operator API service, whilst preserving confidentiality of the MSISDN through the developer application there is a provision in the OneAPI Exchange Discovery service to provide the MSISDN information in a secure way to the Serving Operator so that the user is not required to provide it again. Discovery service returns an encrypted form of the user submitted MSISDN to the developer as an optional parameter "**subscriber_id**" when MSISDN based discovery has been processed. During the API call flow the developer would then pass on this information to the Serving Operator. A Serving Operator can then decrypt the encrypted MSISDN and use this either directly as MSISDN or indirectly via association with another form of user identity in any user sign-in process. The developer should

Sample Success Response	
Status Code	200
Header	**Name** **Value** Content-Type application/json Set-Cookie mcc_mnc=310_150; Secure;Domain=.example.com

Body	{
	"ttl": "1364547792",
	"response": {
	"client_id": "VQpqwTdmOcOkmU3yfeo4QX4XGnCKS8OT",
	"client_secret": "xhsee343xhsssdf",
	"subscriber_operator": "att_us",
	"currency": "USD",
	"country": "US",
	"apis": {
	"payment": {
	"link": [
	{
	"rel": "charge",
	"href": "http://operator.com/v1/payment/{endUserId}/transactions/amount"
	},
	{
	"rel": "refund",
	"href": "http://operator.com/v1/payment/{endUserId}/transactions/amount"
	}
	]
	},
	"operatorid": {
	"link": [
	{
	"href": "https://mconnect.dialog.lk/openidconnect/authorize",
	"rel": "authorization"
	},
	{
	"href": "https://mconnect.dialog.lk/openidconnect/token",
	"rel": "token"
	},
	{
	"href": "https://mconnect.dialog.lk/openidconnect/userinfo",
	"rel": "userinfo"
	}
	]
	}
	}
	}
	"subscriber_id": "3B2E3AC7B8208CBC9D9257DC0E0E30E..."
	}

not attempt to decrypt the encrypted the subscriber_id field. This field can only be decrypted by the respective subscriber Operator. Please see notes at the end on the exact process of encryption.

It is important to note here that the MSISDN obtained via the Exchange in this process should be used by the Serving Operator only for the purpose of identification of a subscriber. It should not be trusted or taken as a measure of authentication.

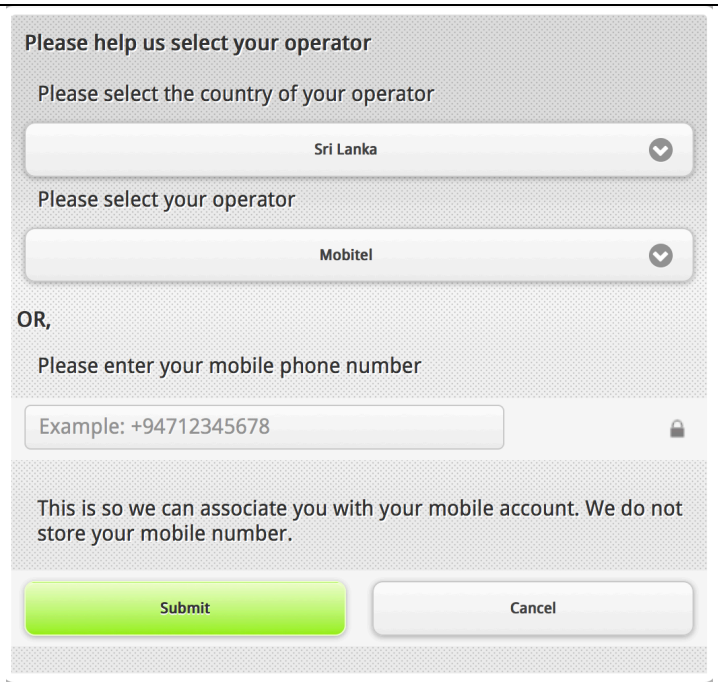
Operator-Not-Identified Response

In case OneAPI Exchange cannot identify the operator, it will prompt user with list of operators, user should select operator and submit.

Sample Response clients accepting application/json Content-Type	
Status Code	202

Body	{
	"links": [
	{
	"rel": "operatorSelection",
	"href": "http://gsmaoneapiexchange.com/v1/discovery/users/operator-selection?session_id=X4XGnCKS8OT"
	}
	]
	}

Sample Response for Clients Accepting text/html Content-Type	
Status Code	302
Response Headers	< HTTP/1.1 302 Found < Date: Mon, 30 Jun 2014 06:52:20 GMT < Location: http://gsmaoneapiexchange.com/v1/discovery/users/operator-selection?session_id=dc809e84-3e94-4744-b937-5c6bb88c5371 < Access-Control-Allow-Origin: * < Access-Control-Allow-Headers: origin, x-requested-with, x-source-ip, Accept, Authorization, User-Agent, Host, Accept-Language, Location, Referer < Access-Control-Max-Age: 600 < Access-Control-Allow-Methods: GET, POST < Access-Control-Allow-Credentials: true < Content-Length: 0 < Content-Type: text/plain; charset=UTF-8

Screenshot of the redirected page	

If developers app is client side application, i.e., indicated by Accpet:application/json header in the request (no redirect_uri required in that case), then exchange returns the list of supported operators and developers app is responsible for displaying a view and get operators selection. Below is the sample response of an application which have access to selected four subscriber operators.

Sample Response for Clients Accepting application/json Content-Type	
Status Code	200

Body	{
	"data": {
	"US": [
	{
	"mcc_mnc": "310_150",
	"opName": "AT&T"
	}
	],
	"GB": [
	{
	"mcc_mnc": "234_15",
	"opName": "Vodafone UK"
	}
	],
	"DE": [
	{
	"mcc_mnc": "262_06",
	"opName": "Deutsche Telekom"
	},
	{
	"mcc_mnc": "262_02",
	"opName": "Vodafone Germany"
	}
	]
	}
	}

At the end of this process identified or user selected mcc and mnc is returned to the redirected URI of the application (sent in the original request), like below:

Sample Response after user selected discovery	
Status Code	302
Response Headers	< HTTP/1.1 302 Found < Date: Mon, 30 Jun 2014 06:52:20 GMT < Location: http://developers.redirect.uri?mcc_mnc=000_01& subscriber_id=3B2E3AC7B8208CBC9D9257DC0E0E30E... < Access-Control-Allow-Origin: * < Access-Control-Allow-Headers: origin, x-requested-with, x-source-ip, Accept, Authorization, User-Agent, Host, Accept-Language, Location, Referer < Access-Control-Max-Age: 600 < Access-Control-Allow-Methods: GET, POST < Access-Control-Allow-Credentials: true < Content-Length: 0 < Content-Type: text/plain; charset=UTF-8

The application then can extract the mcc/mnc and make another request to obtain the operator's API end-points.

Note: The that the **subscriber_id** field is optional in this redirect. The subscriber id field is returned only if the operator has a public key set up in the exchange to encrypt MSISDNs and in the user selected discovery process the user had submitted a MSISDN.

Error Response

These are the status codes that the OneAPI Exchange Gateway will return. Otherwise the webserver might return generic Http status codes such as 502, 503 etc.

When an error occurs, the Response should contain HTTP status code, error code (**error**) and error message (**error_description**)explaining what went wrong.

Code	Name	Description
400	Bad Request	The server could not understand the client's request, or it was invalid. e.g. Authorization header which is a required parameter, when missing results into 400 bad request status code.
404	Not Found	The requested resource does not exist on the server. Description should contain the details. 1. Operator not found 2. Application not found
401	Unauthorized	client_secret supplied in the request is invalid.
500	Internal Server Error	An unexpected error occurred on the server. Inspect response body for details

error value can be one of followings:

- invalid_request : For HTTP code 400, Bad Request
- application_not_found : For HTTP code 404, The client_id was not found.
- invalid_application : For HTTP code 404, No active or valid client_id found in gateway.
- unexpected_error : For HTTP code 500, An unexpected error

error_description An optional human-readable text providing additional information to help in the understanding of the error occurred.

The error code and error description are included in the entity body of the HTTP response using the "application/json" content.

Sample Error Response

Sample Error Response	
Status Code	400
Header	**Name** **Value** Content-Type application/json

Body	{
	"error":"invalid_request"
	"error_description":"The server could not understand the client's request."
	}

Notes on MSISDN Encryption Process

Following are the key tenets of the MSISDN Encryption process:

1. For the encryption of the MSISDN the exchange uses RSA based asymmetric encryption / decryption while using separate key pairs for each Serving Operator[2].
2. The OneAPI Exchange stores the public key of each operator for this purpose. The key is recommended to be 2048 bits in length.
3. The OneAPI Exchange exposes an API to the Serving Operator to update the public key string by the Serving Operator themselves. The RSA public key shall be provided as a base 64 encoded textual representation.
4. In case MSISDN is used for discovery the OneAPI Exchange encrypts the user specified MSISDN and returns this to the calling application as an additional response parameter in the discovery response. The parameter is named as "subscriber_id".
5. During the API usage, the developer application would send the subscriber_id to the Serving Operator, in the case of Mobile Connect API the application will set the 'login_hint' parameter of the authorization request to the String 'ENCR_MSISDN:<subscriber_id>'
6. The Serving Operator then can recognize they have received an encrypted MSISDN and decrypt the string using its private key (which is not known to the OneAPI Exchange or any third party)
7. The OneAPI Exchange does not store the subscriber_id or the MSISDN even in encrypted form.

The OneAPI Exchange encrypts the MSISDNs using the following logic:

- A random string of non numeric characters would be appended to the MSISDN provided by the user – ensuring the source string contains at least 128 characters

2 Note: Using the Serving Operator specific public key for encryption ensures no other party (e.g. other Serving Operators) can decrypt the encrypted MSISDN

- Asymmetric RSA encryption of the resulting string is then applied using the Serving Operator public key
    - The Serving Operator public key should have 2048 bit key length - the Serving Operator can issue any suitable key for the purpose, and it is recommended that a key dedicated for this process is issued
    - Encryption will use PKCS1 padding
- The resulting encoded data is then Hex encoded for sharing with the Serving Operator
- In the OpenID context, when the application calls the operator API service e.g. OpenID Connect authorization request
    - The encoded MSISDN is passed in the API call as an API relevant parameter (e.g. login_hint in Mobile Connect where the encoded MSISDN is prefixed by the sending application or SDK using the format 'ENCR_MSISDN:<subscriber_id>') or alternatively as an HTTP Header (X-Encoded-MSISDN)
    - The Serving Operator recognises the input of the encrypted MSISDN and decodes the base64 encoded data
    - The Serving Operator then applies their private key to decode the RSA coded data
- The Serving Operator then extracts the initial (numeric) portion of the decrypted data as the MSISDN and uses this for any relevant purpose in API services/ user sign-in