V5.0.6 + DataPower Gateway only

OAuth metadata URL and authentication URL

You can use the Metadata URL or Authentication URL parameters to request user-defined content from a remote server and include it in the access token or in the response payload that contains the access token.

Including custom metadata

In many scenarios, custom metadata needs to be included during the access token generation process. The metadata is either stored inside the access token or it is sent along with the access token to the client application. The client application can then send that access token, or the metadata in the payload, in a subsequent request to APIConnect where the metadata is retrieved, validated, or sent to the downstream systems as required.

Examples include, but are not limited to:

- When resource owners are authenticated, metadata about the authenticated resource owner needs to be stored within the access token.
- The grant type that was used to obtain the token is another example of a metadata within the access token.
- A confirmation code that needs to be sent to the client application along with access token.

Configuring Metadata URL or Authentication URL in API Connect to obtain metadata

Metadata can be set by using either or both of the following URLs:

- Metadata URL - When you call the Metadata URL, an HTTP GET request is sent and API Connect expects an HTTP 200 OK along with an optional set of the specified response headers.
- Authentication URL - When you call the Authentication URL, the API Connect gateway sends a GET request with HTTP headers and then processes any HTTP response from the URL. For authentication, a REST authentication service is expected at the Authentication URL.
  See: Authentication URL.

The Metadata URL is configured in the API Connect UI when you configure OAuth-2 Provider API. When you save the API, the OpenAPI (Swagger 2.0) file is updated with a new section under `oauth2:`

```
metadata:
   metadata-url:
      url: 'your_Metadata_URL'
      tls-profile:
```

Both of the URLs can send the following two HTTP headers in their response:

```
API-OAUTH-METADATA-FOR-PAYLOAD
```

```
API-OAUTH-METADATA-FOR-ACCESSTOKEN
```

The response header value from `API-OAUTH-METADATA-FOR-PAYLOAD` is placed in the response payload and indicated as `metadata`.

The response header value from `API-OAUTH-METADATA-FOR-ACCESSTOKEN` is placed within the access token and

indicated as `miscinfo`.

The two metadata response headers are case insensitive and you must escape any special characters in the string value content.

An example response payload that contains metadata along with the access token:

```
{
"token_type":  "bearer",
"access_token":  "AAEkNzhjDHYyyYy...cL0Mv6ctl37w7ZU",
"metadata":  "m:metadata-for-payload_content"
"expires_in":  3600,
"scope":  "read",
"refresh_token":  "AAEnj5SynCMybF...oEZ6JjxYax_HdNg",
}
```

This example output from token introspection endpoint shows the contents of the access token with "`miscinfo`" containing the metadata information.

```
{
"active":  true,
"token_type":  "bearer",
"client_id":  "78c2f10f-799a-4e1f-8e0a-098634997a35",
"username":  "Fred Smith",
"sub":  "fred",
"exp":  1479850049,
"expstr":  "2016-11-22T21:27:29Z",
"iat":  1479846449,
"nbf":  1479846449,
"nbfstr":  2016-11-22T20:27:29Z",
"scope":  "read",
"miscinfo":  "m:metadata-for-accesstoken_content",
"client_name":  "MobileApp"
}
```

Input to Metadata URL

The following request headers are sent to the Metadata URL.

Note

Any existing metadata values that were previously sent from the Authentication URL are also sent in the two request headers `x-existing-metadata-for-payload` and `x-existing-metadata-for-access-token`. The Metadata URL can make use of this information to create a new set of metadata values.

The two request headers that are sent to the Metadata URL are displayed in bold text.

```
X-existing-metadata-for-payload     payload-from-auth-url
X-existing-metadata-for-access-token     token-from-auth-url
X-URI-in     /org/env/miscinfo/oauth2/token (the URL that was sent to APIConnect for this particu
X-METHOD-in     POST
X-POST-Body-in     client_id=client_id&grant_type=password&scope=read&username=name&password=pass
X-X-Client-IP     IP_address
X-X-Global-Transaction-ID     ID_number
...
```

Retrieving Metadata in APIConnect

As described in the previous example scenarios, the metadata can be retrieved from the access token in the Application API and sent to the downstream systems. Retrieval can be done in the API assembly, which is secured to accept tokens in the security definitions.

In the resource API that accepts an access token, the `miscinfo` field can be accessed in the Assembly with the `oauth.miscinfo` context variable, as in the example.

```
apim.setvariable('message.body',apim.getvariable('oauth.miscinfo'));
```

You can also use token introspection to look at the contents of the access token. The Creating an OAuth provider API topic has instructions to enable token introspection.

Refresh tokens and metadata

Authentication URL (if configured for authentication) is invoked first, during authentication of the resource owner. Metadata URL is invoked as the last step, just before the generation of the access token. The only exception, is when you generate access tokens from a refresh token. In cases where refresh tokens are used to generate new access tokens, the Metadata URL is not invoked, the metadata from the refresh token is retained, and then copied into the newly generated access token.

Identifying the source of the metadata

There are two places from where metadata can be set, the metadata is prefixed with keywords to differentiate between the two sources.

- Metadata from the Metadata URL is prefixed with `m:`
- Metadata from the Authentication URL is prefixed with `a:`

> **Note**
>
> When revocation is enabled, some internal details are also stored in the `miscinfo` field, in square brackets within the access token as shown in the example.
>
> ```
> "miscinfo": "[tlsprofile@https://api-revoke-url:443/server/revocation-url]m:metadata-for-acc
> ```

Maximum size of the metadata

Metadata for the access token cannot exceed 512 bytes.

Metadata for payload does not have a specific size restriction, except for when you use the Authorization code grant type, as described in the following sections.

Characters not allowed in metadata in certain scenarios

When you use the Authorization code grant type, or when a consent form is used for implicit grant type, there is a temporary state or code where the metadata from the authentication URL is stored. API Connect internally uses two prefixes - `!ma` and `!mp` to differentiate between payload and token metadata received from the Authentication URL and store them internally in the temporary state/code. Hence these specific character sequences - `!ma` and `!mp` should not be used as the metadata itself.

Grant types and metadata

The OAuth grant types described below include `authorization code`, `implicit`, and `client credentials`.

- **Authorization code grant type**:
  See the following section for details of the payload and token size limits for this grant type.

  Example of the `authorization code` grant type:

```
 $ curl -k -d
"grant_type=authorization_code&code=$mycode&client_id=$myid&scope=scope_introspect&redirect_u
{ "token_type":"bearer",
"access_token":"AAEkOThlZDhhNjYtYTQ1ZS00YTMzLWE0N2QtZmE2OGZkMzQ0NzQ20ZN5Tl_TqYFeIfB7BFf6HFgib


$ curl -k -H "X-IBM-Client-Id: $myid" -H "X-IBM-Client-Secret: $mysecret" -d "token_type_hint=
{ "active":true, "token_type":"bearer", "client_id":"98ed8a66-a45e-4a33-a47d-fa68fd344746", "
```

- **Implicit grant type**:
  When implicit grant type is used, the access token and the metadata are returned in the `location` header as a fragment, as you see in the example.

```
< Location:
 https://localhost#access_token=AAEkOThlZDhhNjYtYTQ1ZS00YTMzLWE0N2QtZmE2OGZkMzQ0NzQ2buS2KfWdq

$ curl -k -H "X-IBM-Client-Id: $myid" -H "X-IBM-Client-Secret: $mysecret" -d "token_type_hint=
              { "active":true, "token_type":"bearer", "client_id":"98ed8a66-a45e-4a33-a47d-fa
```

- **Client credentials grant type**:
  Authentication URL will not be invoked when using client credentials grant type, as there is no resource owner. The metadata from Authentication URL is not available for this grant type. However, content returned from Metadata URL will be included as metadata.

Authorization Code grant type size limitations

When metadata is included from an Authentication URL for an Authorization code grant type, as it is a three legged flow, both the content and the payload are stored within the `dp-state` and carried on to the authorization code and to the access token. Note that around 10 characters are used internally to differentiate between the metadata for payload and metadata for the access token when stored in the `dp-state`. In addition, if revocation is enabled, that will also be part of the token metadata. Hence the combined size of the token metadata, the payload metadata (including the 10 characters of internal data), and internal revocation details, cannot exceed 512 bytes in total.

If the overall size of the metadata exceeds 512 bytes, then the access token generation succeeds, but the metadata fields contain an error message of "`metadata too large`" as shown in the example.

```
"metadata":"m:error: metadata too large for AZ code grant type[Authorization Code-metadata-url-pa
"miscinfo":"[r:gateway]m:error: metadata too large for AZ code grant type[Authorization Code-meta
```

This size restriction can be overcome when the metadata is sent from the Metadata URL and not from the Authentication URL, because the metadata is not stored in `dp-state` or in the authorization code.

Behavior when retrieving metadata with both Metadata URL and Authentication URL

If a Metadata URL is configured and a connection to the external server is successful, the response headers overwrite any existing metadata obtained from the Authentication URL to become the final value. Therefore, you must carefully examine the incoming request headers and create appropriate response headers from the Metadata URL.

If a Metadata URL is configured, but the connection to Metadata URL fails, then a failure message of "`error on metadata url`" is written for metadata in both the payload and the access token.

If a Metadata URL is configured and the connection is successful, but the remote server does not send any of the specified HTTP response headers, a blank value is written for metadata in both the payload and the access token.

> **Attention**
>
> Metadata URL overwrites existing values from Authentication URL. This includes blank values.

If no Metadata URL is configured, the metadata that is obtained from the Authentication URL is retained as the final value.

Sample gatewayscript for simulating a Metadata URL endpoint

This sample gatewayscript can be used in the `gws policy` in the assembly of an API in API Connect for simulating a Metadata URL endpoint.

```
// Get the input headers that contain the exiting headers from the Authentication URL if any.
var existingToken = apim.getvariable('request.headers.x-existing-metadata-for-access-token');
var existingPayload = apim.getvariable('request.headers.x-existing-metadata-for-payload');
// Append metadata to the ones obtained from auth URL
apim.setvariable('message.headers.API-OAUTH-METADATA-FOR-ACCESSTOKEN', existingToken + ' token-fi
apim.setvariable('message.headers.API-OAUTH-METADATA-FOR-PAYLOAD', existingPayload + ' payload-fi
```

Parent topic:

→ Protecting an API with OAuth

Related concepts:

V5.0.6 +

→ Authentication URL

Last updated: Tuesday, 4 September 2018