Taito-Valtti API, Person API and ePerehdytys API OAuth 2.0 client credentials authentication

This page describes the steps needed to authenticate using OAuth 2.0 client credentials authentication flow to access the Taito-Valtti API, Person API and ePerehdytys API.

OAuth 2.0 client credentials authentication flow (https://tools.ietf.org/html/rfc6749#section-4.4) is usually used to authenticate machine to machine access where there is no end-user but some server application. In order for client to make API calls to protected resources it must first obtain an access token from the authentication server. After this the client must include the access token in the Authorization header of the requests.

Terminology

client_id	Username for the external application or "client" which wants the access protected resources.
client_secret	Password for the external application or "client".
Token endpoint	OAuth 2.0 specification defines several different URI endpoints. Token enpoint is the API endpoint used to obtain the final access token which can then be used to access the protected resources.
scope	Scope refers to a permission to access some information. Each protected resources and API endpoint might require a specific scope to access them. Please refer to the API endpoint documentation for the required scopes.

Ingredients

1. Token endpoint URI
2. OAuth 2.0 client_id
3. OAuth 2.0 client_secret
4. Protected resource or API URI

The information is environment specific. Various environments (alpha/beta/production) all have different URI endpoints and account information (client_id and client_secret).

Client credentials authentication

Client credentials authentication is done by making a POST request to the token endpoint of the authentication server. The following HTTP headers must be included in the request

HTTP Header	Required	HTTP header value
Authorization	Yes	*client_id* and *client_secret* encoded using the HTTP Basic Authentication scheme
Content-Type	Yes	*application/x-www-form-urlencoded*

The HTTP request body must contain the following parameters

Parameter	Value
grant_type	*client_credentials*
scope	a list of requested scopes separated by a white-space (eg. "scope1 scope2 scope3")

The token endpoint will respond either with HTTP response code 400 or 401 if the request fails. HTTP response code 400 means that the request was malformed (eg. some mandatory parameter was missing). HTTP response code 401 means that the *client_id* and *client_secret* were not valid and the authentication was not accepted.

For successful authentication the token endpoint returns a HTTP response code 200 with a JSON response body with the following parameters

Parameter	Value
access_token	Encoded access token.
scope	A list of scopes authorized. The list of authorized scopes might differ from the list of requested scopes in case the specified *client_id* does not have permission for some of the scopes. Each *client_id* need to be explicitly authorized to request certain scopes.
token_type	*bearer*
expires_in	Access token validity in seconds.

Example request

```
POST /auth/token HTTP/1.1
Content-Type: application/x-www-form-urlencoded
Authorization: Basic
QCFEMEIzLjQyRkNy42ODFEITAwMDEhMDEwNS4wM0Y2ITAwMDghNUIwNS5EREFCCTpzZWNyZX
Q=
Connection: close

grant_type%3Dclient_credentials%26scope%3Dscope1%20scope2%20scope3
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json
Connection: close

{"access_token":"eyJ0eXAiOiJK....<clip>.....6iFqMjg","token_type":"
bearer","expires_in":3599,"scope":"scope1 scope2"}
```

Authentication example using curl

The following is an example of the client credentials authentication using curl from command line.

--insecure option is used because the token endpoint is using self-signed certificates. *--user* parameter needs to be enclosed in single quotes and not double quotes because the *client_id* contains exclamations marks which has a special meaning in the bash shell.

```
curl --insecure --user '@!D0B3.42FF.3A77.681D!0001!0105.03F6!0008!689D.
C81F:verysecretpassword' -X POST -d
'grant_type=client_credentials&scope=taito_valtti' https://auth.beta.
vaultit.org/auth/token
{"access_token":"eyJ0eXAioI2ucxQI....<clip>....RTu48HNw","token_type":"
bearer","expires_in":3599,"scope":"taito_valtti"}
```

Accessing the Taito-Valtti API using the access token

The following is an example of using the access token acquired in the previous to access the Taito-Valtti API.

```
curl --insecure -H "Authorization: Bearer eyJ0eXAiOiJKV1MiKKzHF...
<clip>...Dsfqk2V-mA7tLU6unpbIuwZ6oL3OdF2MlteJP6LcD7iLw5PLDgF-
RNqC7qGrt5KaGaXZ-p18Ha-9Ce2RaZQLRTu48HNw" https://taito.beta.
tilaajavastuu.fi/api/taito-valtti/v2/competences/by_valtti_uid
/3495873457
{"message": "The card with given id does not exist.", "error":
"CardNotFound"}
```