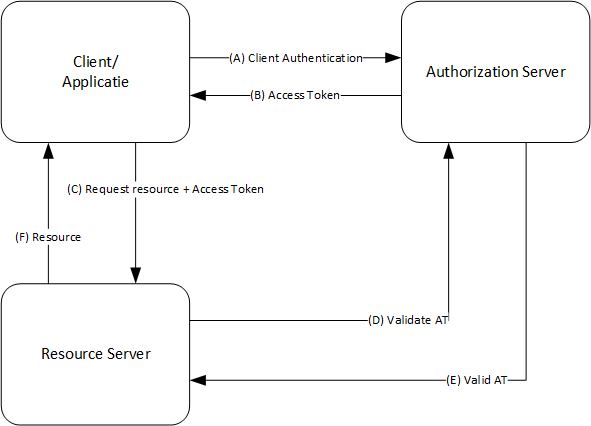
## Inleiding

De eerste en tevens eenvoudigste use case, bestaat uit koppelingen tussen applicaties binnen DSO. Een voorbeeld is het loket met de content API. De communicatie verloopt via het knooppunt. Hierbij is het alleen van belang dat de applicatie zelf informatie nodig heeft en is er geen sprake van bijvoorbeeld ingelogde eindgebruikers die een bevraging doen. De identiteit die gepropageerd wordt is in dit geval die van een applicatie. Dit is een technische identiteit. Omdat de services blootgesteld worden via het knooppunt zullen partijen kenbaar moeten maken dat ze geautoriseerd zijn op de API’s die ze bevragen. Dit word afgedwongen door de API gateway in het knooppunt.

## Beschrijving use case 1

De eerste use case, die van applicatie koppelingen, zal gebruik maken van het client credentials OAuth grant type. Dit grant type kan het beste gebruikt worden bij machine-machine koppelingen. Bij het gebruik van dit grant type zit de impliciete aanname dat de client (bijvoorbeeld het loket), een geheim kan bewaren. Samen met een username zal dit geheim gebruikt worden om te authentiseren bij de Identity Server, welke vervolgens een bearer token uitreikt. Schematisch ziet dit proces er als volgt uit:



Figuur 1 OAuth grant type: client credentials voor gebruikersgroep 1 – Interne Applicaties. Een pure OAuth flow

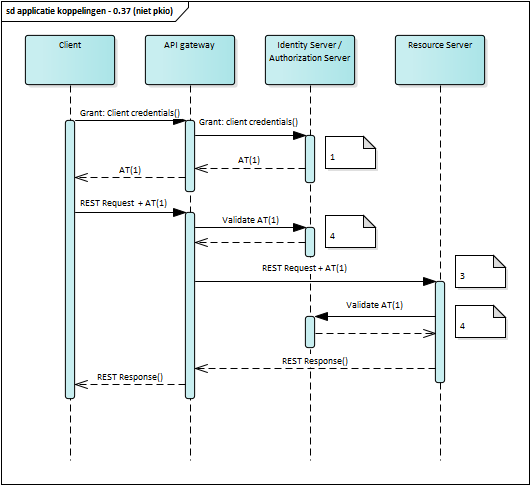
Belangrijk om hierbij te vermelden, is dat het access token een bearer token, is welke direct toegang geeft tot de backend API namens de eigenaar van dat token. In dit geval is dat dus de client applicatie welke hem heeft aangevraagd. Het is daarom van belang dat dit token geheim blijft en de client ook de enige is die dit token heeft.

Twee andere componenten die gebruik zullen maken van deze manier van het verkrijgen van een access token (naast het loket en gebruikerstoepassingen) zijn de API Key Handler en de PKIO Cert handler. Deze componenten vertalen respectievelijk een gebruikers PKIO certificaat en een API key naar een access token, waarbij ze gebruik maken van bovenstaand mechanisme [SP-API-Autorisaties].

De rollen van de standaard OAuth flows zijn bij dit grant type beperkt tot client, resource server en autorisatie server. De resource owner speelt bij dit grant type geen rol, immers het eigenaar van het access token is de client applicatie en niet een eindgebruiker. In de genoemde voorbeelden zijn de clients bijvoorbeeld het loket, de API key handler, gebruikerstoepassingen of de PKIO Cert Handler. De autorisatie server is altijd de Identity server van PR05/PR29 en de resource server kan bijvoorbeeld de content API zijn.

## Sequence Diagram Use Case 1

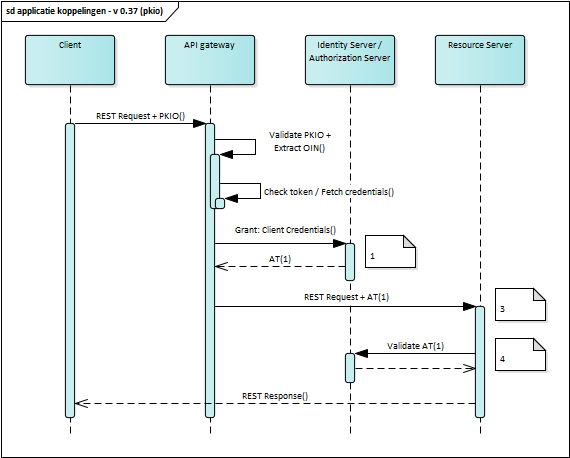
Om het interactiepatroon tussen de verschillende partijen te omschrijven is onderstaand diagram gemaakt; deze proces flow is gebaseerd op de client credentials flow (een OAuth grant type)



Figuur 2 Sequence diagram Applicaties koppelingen use case 1 - Algemene flow

In deze sequence flow kan als voorbeeld het loket (java component) de client zijn en de Content API de resource server. De identity server is de autorisation server. De eindgebruiker speelt in het OAuth gedeelte van de flow geen rol: die gaat om applicaties die met elkaar koppelen.

Er zijn twee voorbeelden van clients die deze flow gebruiken om toegang te krijgen tot resources, de PKIO Handler en API key handler. Het sequence diagram waarin een applicatie een API bevraagd met behulp van een PKIO certificaat is hieronder gegeven. Aan het principe van de client credentials flow veranderd uiteraard niets, er is een client applicatie die een resource API bevraagd met behulp van een access token. De API keyHandler werkt op soortgelijke manier. De PKIO Cert Handler en API Key Handler kunnen dus als implementatie van use case 1 gezien worden. Doordat tokens bijgehouden worden door de Handlers, is de implementatie hiervan vergemakkelijkt voor de afnemer.



Figuur 3 Sequence diagram Applicaties koppelingen use case 1 - Voorbeeld implementatie PKIO- / API Key -handler

 Indien de resource server wenst te weten welke identiteit (client) de eigenaar is van het access token, en dus toegang heeft verkregen tot de resource, zal met behulp van het access token, een zogenaamde token retrospection moeten worden gedaan. Met behulp van het token, en client credentials (client ID + secret) moet gecontroleerd worden bij de identity server, welke client applicatie bij het access token hoort.