AuthzServer

http://<host_IP_authzserver>:9000/authzserver/

As a requirement to create a Security Policy is necessary to previously provision:

- Security domain
- Organization
    - Users
- Source of identification

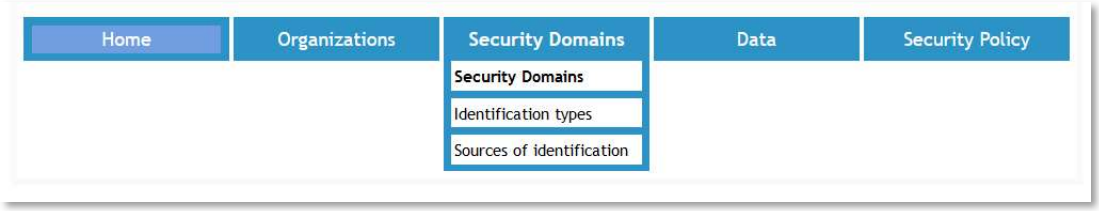
This guide will provision the different values associated to be used in the Pilot1

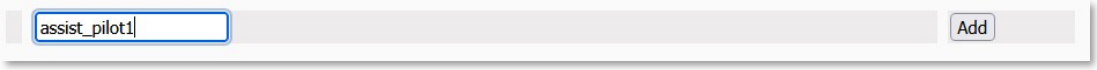
- Security domain = assist_pilot1
- Organization = transport_org
- Organization users
    - user1 / demo_truck1 (oauth2)
    - user2 / demo_truck2 (oauth2)
- Source of identification = mobileapp

AuthzServer Configuration

- **Security Domain**

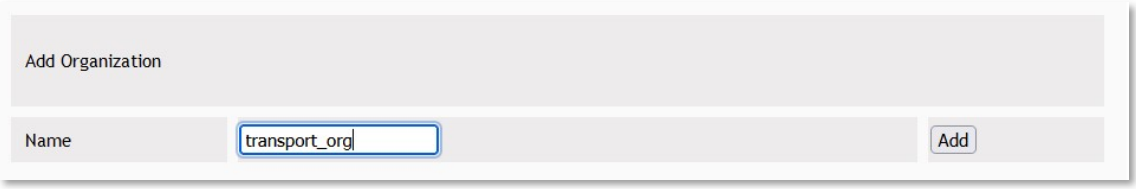
  Create a Security Domain





- **Organization**

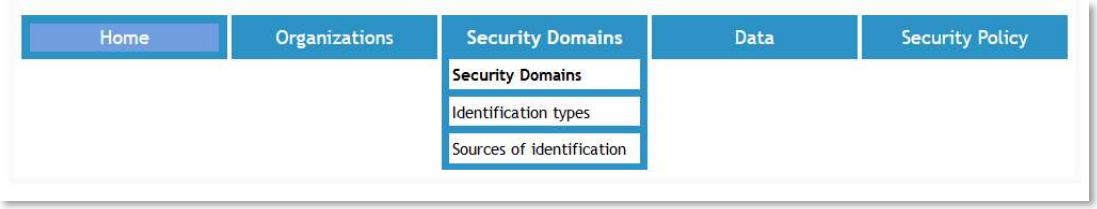
  Create organization



Later we will create/provision the users associated to this organization

- **Security Domain**

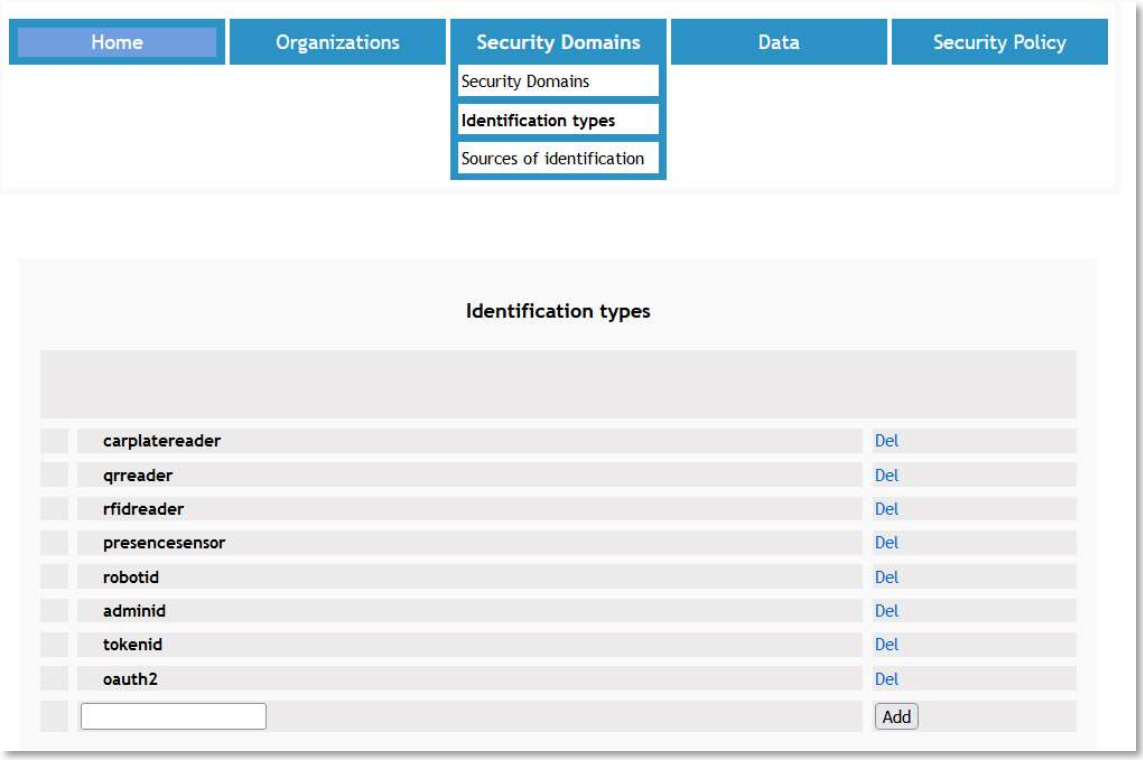
Associate the organization to the Security domain



This step will link an Organization with a Security Domain

- **Identification types**

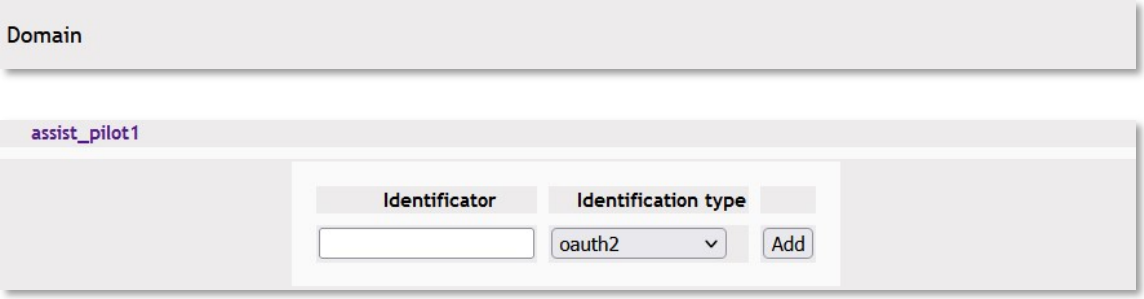
Different type of data for identification can be used. If not created, we can provide new types.



- **<u>Sources of identification</u>**

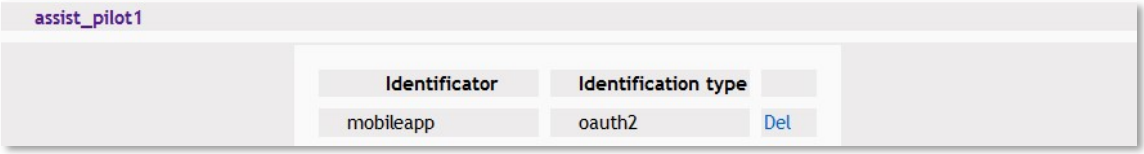
We can create sources of identification for client app to use oauth as Identification type.

It is mandatory to select the Security Domain to be used.



Identificator must be associated with the name that will be used later in the client app for the resource (idresource).

In the example we will use as Identificator= mobileapp



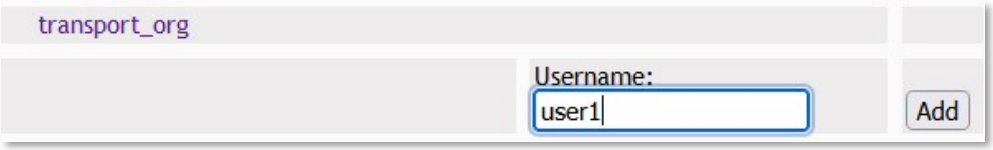
- **Organization/users**

Create users for organization transport_org

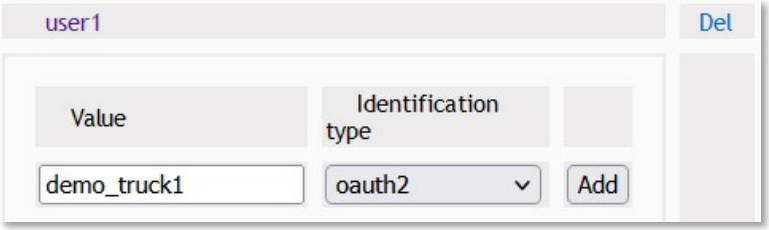
- user1

- user2

Create user1 for transport_org

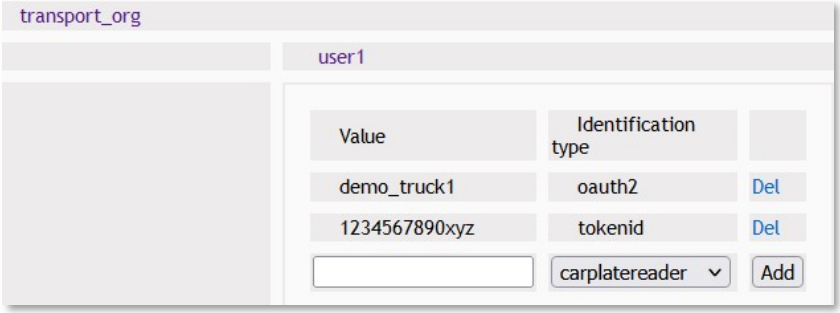


A user in an organization can have different ways to be identified

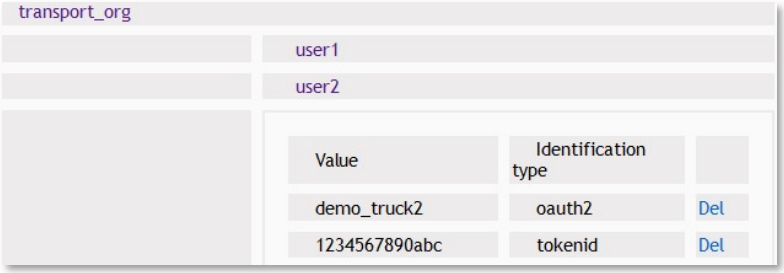


Additional values for identification can be added. For example

- tokenid



Create user2 for transport_org



Authserver will validate requests from user1 and user2 using the provisioned Identification types.

It will be necessary that the same value for user1 will be provisioned on the IdM.

The configuration guide to provision mobileapp and users is described here in IdM Keycloack clientapp configuration.

Federated Provider

Create a Federated Provider

Provider name = geolocation

Value set key name = userid

Value set key name should be the key name of the field used as reference for the value set.

Configure a Federated Provider

These values will feed an endpoint PIP to be checked when evaluate the policy.

- demo_truck1 / alignment / yes
- demo_truck2 /alignment / no

Assing a value-set for the value-set name created for the Federated Provider

Assign the values that the Federated provider will provide to later match with a Security Policy

To assign a new values that will be used to evaluate in security policy, repeat the process.

This is the result of the process for provisioning values to be consumed in the Data Provider

Data Provider

After configuration of a Data Provider. This will add an endpoint to be consumed by the AuthzServer and these data will be used to be evaluated in the Security Policy

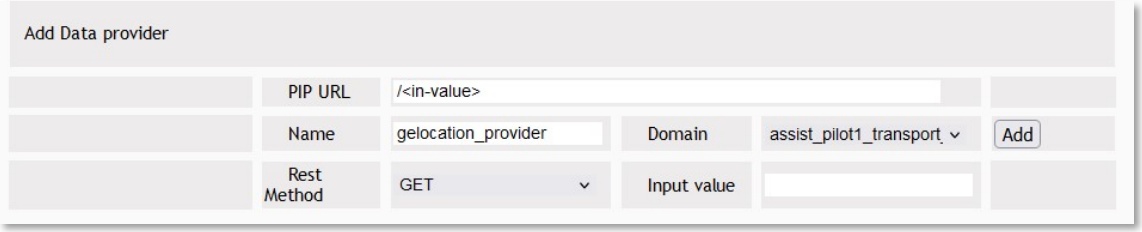
http://192.168.15.178:9000/authzserver/rest/fedprov/geolocation


```
[{"userid":"demo_truck1","alignement":"yes"},{"userid":"demo_truck2","alignement":"no"}]
```

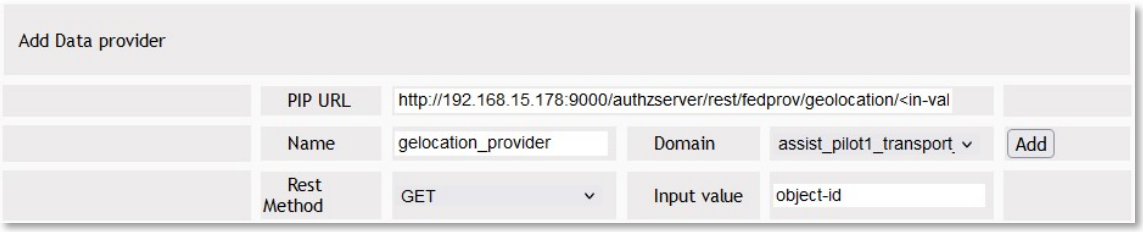

Configure a Data Provider

Data provider values should be linked with the Provider name associated in the Federated Provider i.e., geolocation to be consumed as a URL in the PIP

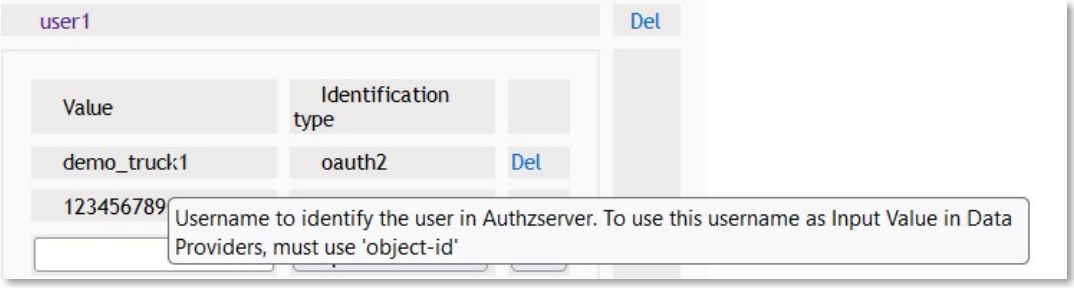
- Name = geolocation_provider
- Domain = assist_pilot1_transport
- PIP URL http://192.168.15.178:9000/authzserver/rest/fedprov/geolocation/<in_value>
- Input value = object-id (*)



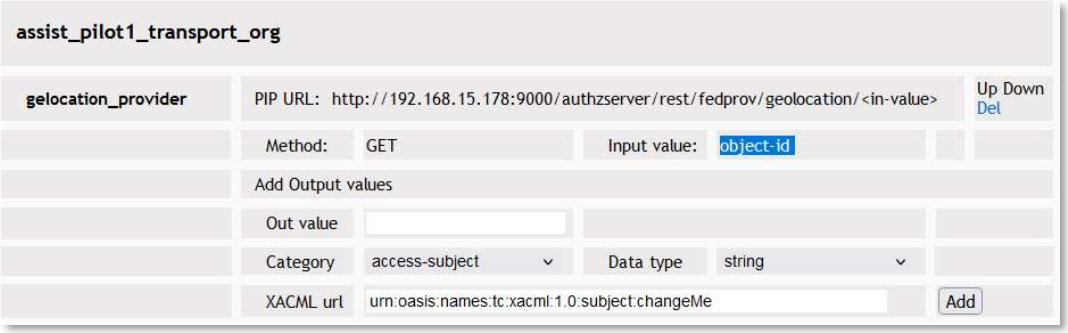
Complete the URL with the PIP using provider name and select the Domain



(*) In this case object-id due to the value will be associated to the type of identification for a user

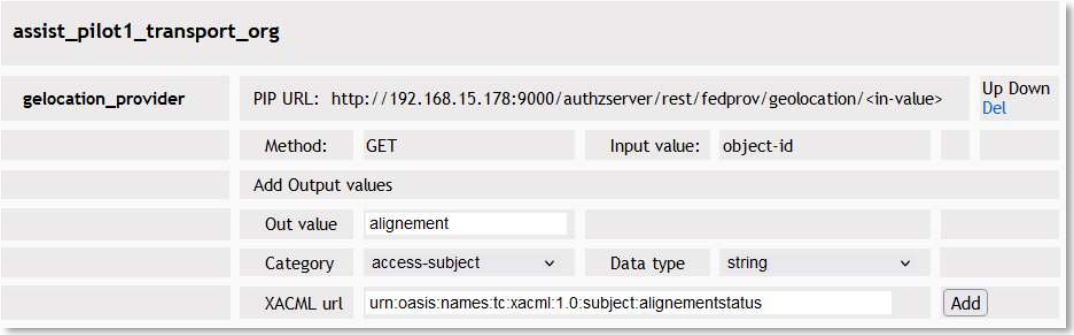


This is the result after provisioning the PIP for the geolocation provider

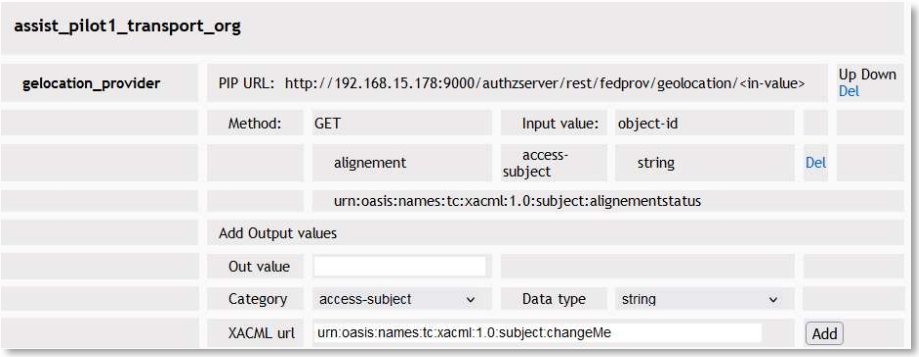


As a last step is necessary assign and change XACML URL to evaluate parameter i.e., we will give the name alignementstatus which will be the output to the variable set alignment configured for the value pair (alignment = yes, or alignment= no) assigned for the Provider name, in the Federated Provider section.

- Out value = alignement
- urn:oasis:names:tc:xacml:1.0:subject:alignementstatus



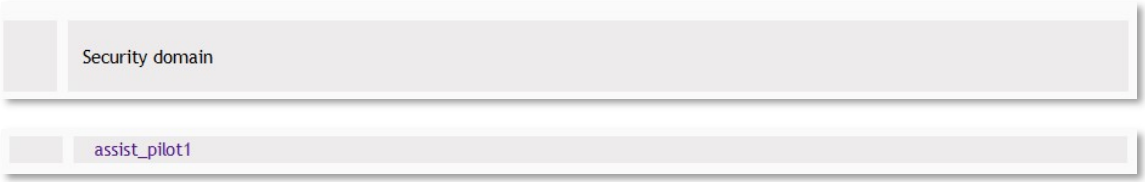
Data provider created



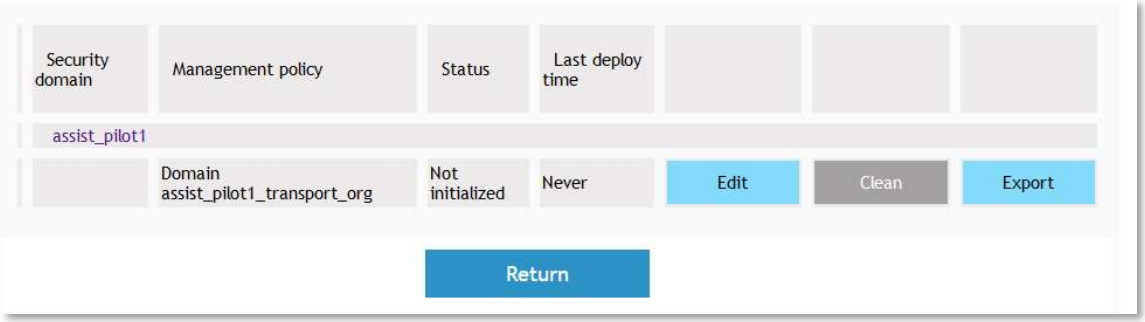
Security Policy

Create a security policy for the Security domain and Organization

Select a Security domain

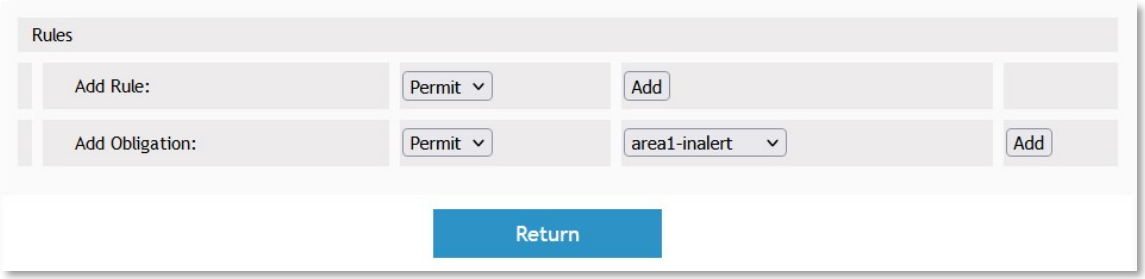


As previously linked Security domain will present the Organization associated with this security domain.

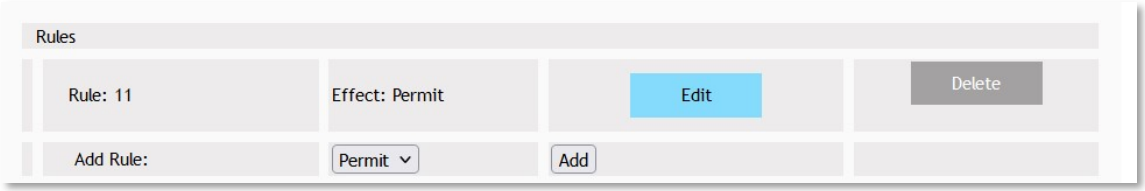


Create a Security Policy

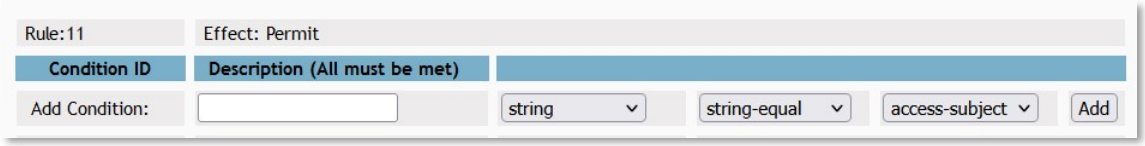
Edit



Add a Rule and Edit



Edit



An example for the Security Policy will evaluate

- Authenticated user input from an application consuming oauth2
- Alignment status between the truck and the crane simulated by the geolocation provider

```
[{"userid":"demo_truck1","alignement":"yes"}
```

```
[{"userid":"demo_truck2","alignement":"no"}
```

Security Policy is formed by rules. We can add Condition or Evaluations to a Rule.

We will create a security policy that will check

- Input data on action for a client app authenticated user
- Alignment status yes / no

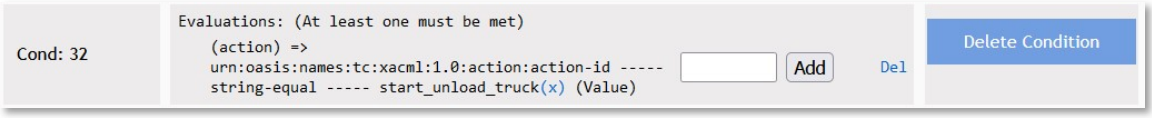
Add new condition

- urn:oasis:names:tc:xacml:1.0:action:action-id

In the condition setting is mandatory to select **`action`**

The condition to be checked is action = start_unload_truck

In the condition setting is necessary to select **`Value`**



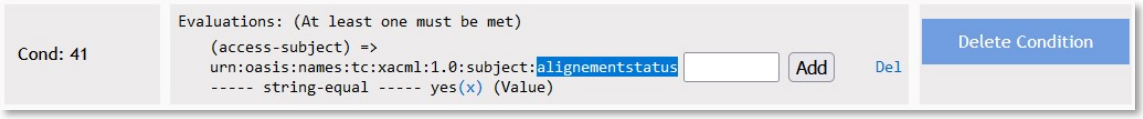
Add a new condition in the rule

- urn: oasis:names:tc:xacml:1.0:subject:alignementstatus

In the condition setting is mandatory to select **`access-subject`**

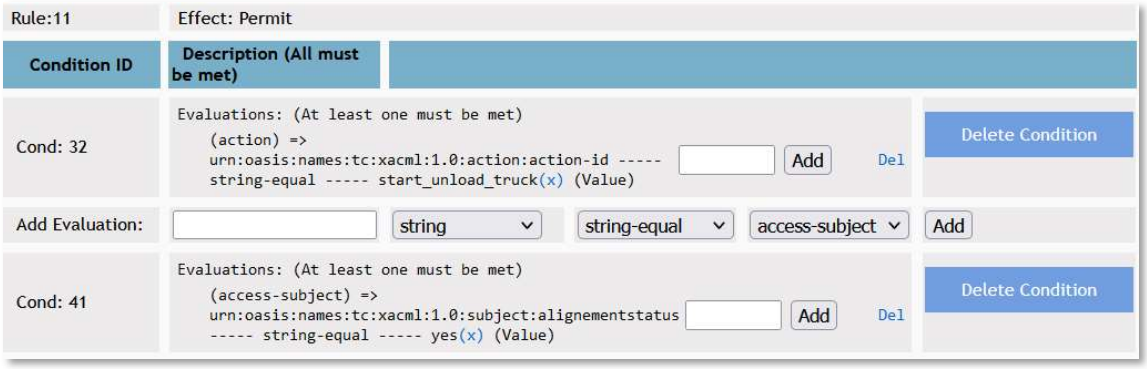
The condition to be checked is alignementstatus = yes or alignementstatus = no

In the condition setting is necessary to select `Value` (that will be replaced by yes or no)

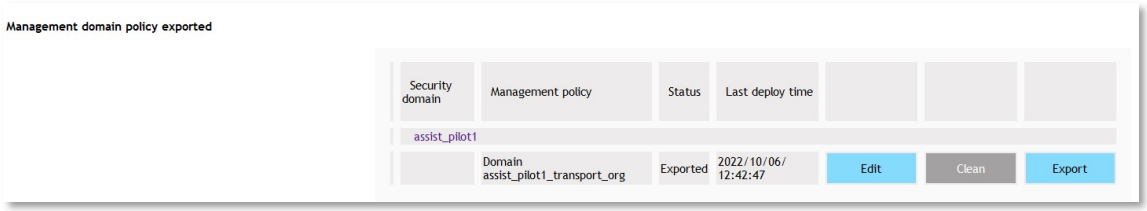


Rule 11 is formed with

- Cond: 32
- Cond: 41



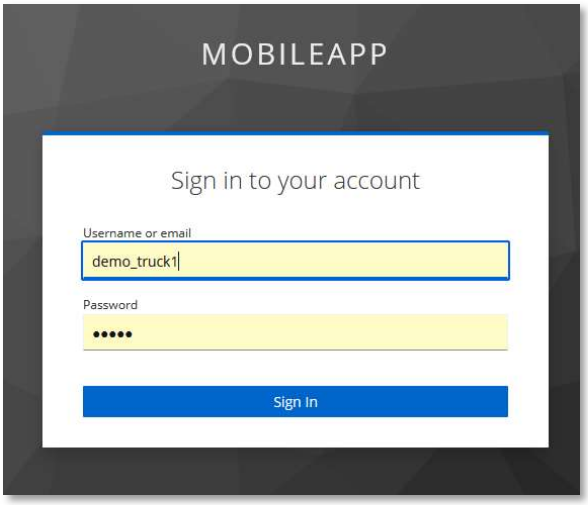
Once that the Security Policy is created it should be exported



Note: Authenticated user on an external application

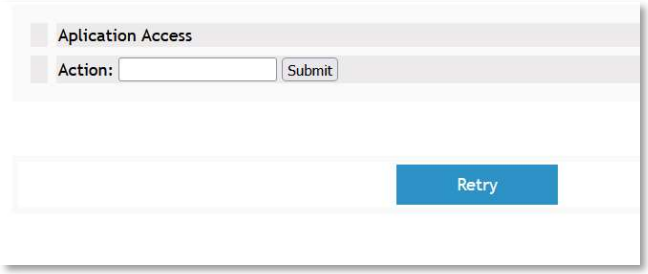
http://192.168.15.178:8000/clientApp/mobileapp.jsp

Redirection to IdM to authenticate the client application authentication against the IdM



After user authentication the client application will accept input values to be evaluated in the Security Policy.

http://192.168.15.178:8000/clientApp/mobileapp.jsp



Security policy rule will be created to evaluate application input on Action

start_unload_truck


Aplication Access
Action:
start_unload_truck
Submit
Retry