# Azure AD App Registration Prozess

Die Authentifizierung am Tekton Control Panel und am Controller API erfolgt über das Microsoft Login (SSO). Grundlage dafür ist eine [App Registration](https://confluence.sbb.ch/x/TgjHY), welche über das IAM API angelegt und gepflegt wird. Jede ESTA Tekton Instanz benötigt ihre eigene App Registration. Die automatisierte Erstellung einer App Registration ist bereits Teil des ESTA Tekton Setups und wird über das [esta-tekton-pipeline-config](https://code.sbb.ch/projects/KD_ESTA/repos/esta-tekton-pipeline-config/browse/charts/esta-tekton-pipeline-config/templates) Helm Chart konfiguriert und mit einen Taskrun ausgeführt.

Für die Kommunikation mit dem Azure AD API wurden bisher statisch definierte Credentials verwendet, welche für alle Tekton Namespaces identisch sind und auch in jedem Namespace als Secret hinterlegt sind. Der App Registration Prozess soll nun auch die automatisierte Erstellung einer individuellen APIM Application für Azure AD pro Tekton Namespace beinhalten. Der Setup Prozess sieht dann wie folgt aus:

1. APIM Application für Azure AD erstellen  
   (via ArgoCD mit APIM Credentials im ArgoCD Namespace hinterlegt)
2. Client ID und Secret für Azure AD API als Secret azure-ad-client-credentials im Tekton Namespace anlegen  
   (diese werden vom Controller auch für die Manipulation von group-role Assignments verwendet)
3. Azure AD App Registration via API registrieren  
   (mit den Credentials aus Secret azure-ad-client-credentials)
4. Client ID in Secret esta-tekton-oauth-client-ids speichern
5. ESTA Tekton Controller startet mit Values aus azure-ad-client-credentials und esta-tekton-oauth-client-ids

## Scripts für das Erstellen einer Azure AD Application via APIM API (1. und 2.)

(Client ID und Secret für das APIM API sind in ESTA Passwords hinterlegt)

**create-azure-app.sh** Quelle erweitern

#!/bin/bash

set -e

TEKTON\_NAMESPACE="$1" # provide the Tekton namespace as argument

APIM\_ACCOUNT\_ID="603"

APIM\_API\_URL='https://api-backend.api.sbb.ch:443'

OAUTH\_URL='https://sso.sbb.ch/auth/realms/SBB\_Public/protocol/openid-connect/token'

CLIENT\_ID='<apim-app-id>'

CLIENT\_SECRET='<apim-app-secret>'

if [ -z "${TEKTON\_NAMESPACE}" ]; then

echo "Missing TEKTON\_NAMESPACE argument. Aborting."

exit 1

fi

##

## Authenticate to APIM API

##

AUTH\_TOKEN=$(curl -s -X POST ${OAUTH\_URL} \

-H 'Content-Type: application/x-www-form-urlencoded' \

-d grant\_type=client\_credentials \

-d client\_id=${CLIENT\_ID} \

-d client\_secret=${CLIENT\_SECRET} \

| grep -o -E '"access\_token":"[^"]+"' | cut -f 2 -d ":" | tr -d '"\n')

echo "Successfully obtained created an APIM access token"

##

## Create an APIM Azure AD Application

##

cat > azure-app-request.json <<EOL

{

"serviceId": 163,

"planId": 1114,

"name": "ESTA Tekton Azure AD ($TEKTON\_NAMESPACE)",

"description": "Managing Azure AD App Registrations for ESTA Tekton",

"oauthFlow": "clientCredentials",

"authProvider": "AZURE\_AD",

"megaId": "FEE502DB61786599"

}

EOL

HTTP\_STATUS=$(curl -sk -X POST "${APIM\_API\_URL}/accounts/${APIM\_ACCOUNT\_ID}/applications" \

-H "Accept: application/json" \

-H "Authorization: Bearer ${AUTH\_TOKEN}" \

-H "Content-Type: application/json" \

-d '@azure-app-request.json' \

-o azure-app-response.json \

-w "%{http\_code}")

if [[ ${HTTP\_STATUS} -gt 299 ]]; then

echo "App creation request failed with error $HTTP\_STATUS"

exit 2

fi

# extract application client ID and secret

APP\_CLIENT\_ID=$(jq -r .applicationId azure-app-response.json)

APP\_CLIENT\_SECRET=$(jq -r .secret azure-app-response.json)

echo "New Application with ID $APP\_CLIENT\_ID created"

##

## Create azure-ad-client-credentials secret

##

cat > azure-ad-client-credentials.yaml <<EOL

kind: Secret

apiVersion: v1

metadata:

name: azure-ad-client-credentials

namespace: $TEKTON\_NAMESPACE

type: Opaque

stringData:

clientId: '$APP\_CLIENT\_ID'

clientSecret: '$APP\_CLIENT\_SECRET'

scope: 'api://azure\_ad\_api.prod/.default'

tokenEndpoint: 'https://login.microsoftonline.com/2cda5d11-f0ac-46b3-967d-af1b2e1bd01a/oauth2/v2.0/token'

EOL

echo "Successfully created secret azure-ad-client-credentials.yaml"

# wait a bit for the new application to become active

sleep 90

# wait a bit for the new application to become active

sleep 20

##

## Test the app credentials

##

RETRIES=0

AUTH\_TOKEN=""

while [ -z "${AUTH\_TOKEN}" ] && [ $RETRIES -lt 20 ]; do

AUTH\_TOKEN=$(curl -fs -X POST https://login.microsoftonline.com/2cda5d11-f0ac-46b3-967d-af1b2e1bd01a/oauth2/v2.0/token \

-H 'Content-Type: application/x-www-form-urlencoded' \

-d grant\_type=client\_credentials \

-d client\_id="${APP\_CLIENT\_ID}" \

-d client\_secret="${APP\_CLIENT\_SECRET}" \

-d scope='api://azure\_ad\_api.prod/.default' \

| grep -o -E '"access\_token":"[^"]+"' | cut -f 2 -d ":" | tr -d '"\n')

RETRIES=$[$RETRIES+1]

if [ -z "${AUTH\_TOKEN}" ]; then

echo "No access access token obtained, retry #${RETRIES} in 10 seconds..."

sleep 10

fi

done

if [ -z "${AUTH\_TOKEN}" ]; then

echo "Failed to obtain access token :-("

exit 1

fi

echo "Querying registered applications:"

curl -fsv -X 'GET' \

'https://azure-ad.api.sbb.ch/v1/applications?displayNameStartsWith=SBB%20ESTA%20Tekton&listOnlyOwnApps=true' \

-H 'accept: application/json' \

-H "Authorization: Bearer $AUTH\_TOKEN"

echo ""

echo "All done."

## Migration

Für bestehende Tekton Namespaces sollten optimalerweise ebenfalls individuelle APIM Applications und neue App Registrations erstellt werden. Im Taskrun, welcher die App Registration ausführt, werden in einem ersten Schritt die bestehenden Group-Role-Assignments gelesen und in die statische App Registration YAML gepatched. Für die Migration muss also die neue API Application ID als owner in die bestehenden App Registrations eingetragen werden. Diese Modifikation muss mit den alten globalen API Credentials erfolgen. Die vollständige Migration erfolgt also in zwei Schritten:

1. Re-Assign App Registration mit neuer APIM Application als Owner
   1. Copy azure-ad-client-credentials Secret
   2. Erstellen der APIM Application
   3. Download der App Registration (JSON)
   4. Hinzufügen der neuen APIM Application als Owner
   5. Upload der App Registration (JSON)
2. Entfernen der alten APIM Application als Owner
   1. Download der App Registration (JSON)
   2. Entfernen der alten APIM Client ID als Owner
   3. Upload der App Registration (JSON)
   4. Entfernen des alten statischen azure-ad-client-credentials Secret (Copy)

**reassign-app-registration.sh** Quelle erweitern

#!/bin/bash

# authenticate to Azure API (with old APIM client)

CLIENT\_ID='<azure-ad-client-id>'

CLIENT\_SECRET='<azure-ad-client-secret>'

AUTH\_TOKEN=$(curl -sf -X POST https://login.microsoftonline.com/2cda5d11-f0ac-46b3-967d-af1b2e1bd01a/oauth2/v2.0/token \

-H 'Content-Type: application/x-www-form-urlencoded' \

-d grant\_type=client\_credentials \

-d client\_id="${CLIENT\_ID}" \

-d client\_secret="${CLIENT\_SECRET}" \

-d scope='api://azure\_ad\_api.prod/.default' \

| grep -o -E '"access\_token":"[^"]+"' | cut -f 2 -d ":" | tr -d '"\n')

if [ -z "$AUTH\_TOKEN" ]; then

echo "Failed to authenticate!"

exit 22

else

echo "Got access token for azure\_ad\_api.prod"

fi

# export app registration

APP\_REG\_CLIENT\_ID="$1" # pass client ID to patch as argument

NEW\_CLIENT\_ID='<new-tekton-azure-ad-client-id>'

curl -sf -X 'GET' \

"https://azure-ad.api.sbb.ch/v1/app-configs/json?clientId=${APP\_REG\_CLIENT\_ID}" \

-H 'Accept: application/json' \

-H "Authorization: Bearer $AUTH\_TOKEN" \

-o existing-app-registration.json

echo "Downloaded app registration to existing-app-registration.json"

# add new client as owner # remove old client ID and add new client as owner

jq ".appRegistrations[0].owners = (.appRegistrations[0].owners | map(select(.clientId // "" | contains(\"${APP\_REG\_CLIENT\_ID}\") | not)) + [{\"clientId\":\"${NEW\_CLIENT\_ID}\"}]" existing-app-registration.json > updated-app-registration.json

# upload patched app registration

curl -sf -X 'POST' \

"https://azure-ad.api.sbb.ch/v1/app-configs/json" \

-H 'accept: application/json' \

-H 'Content-Type: application/json' \

-H "Authorization: Bearer $AUTH\_TOKEN" \

-d @updated-app-registration.json

echo ""

echo "Successfully updated app registration for client $APP\_REG\_CLIENT\_ID"