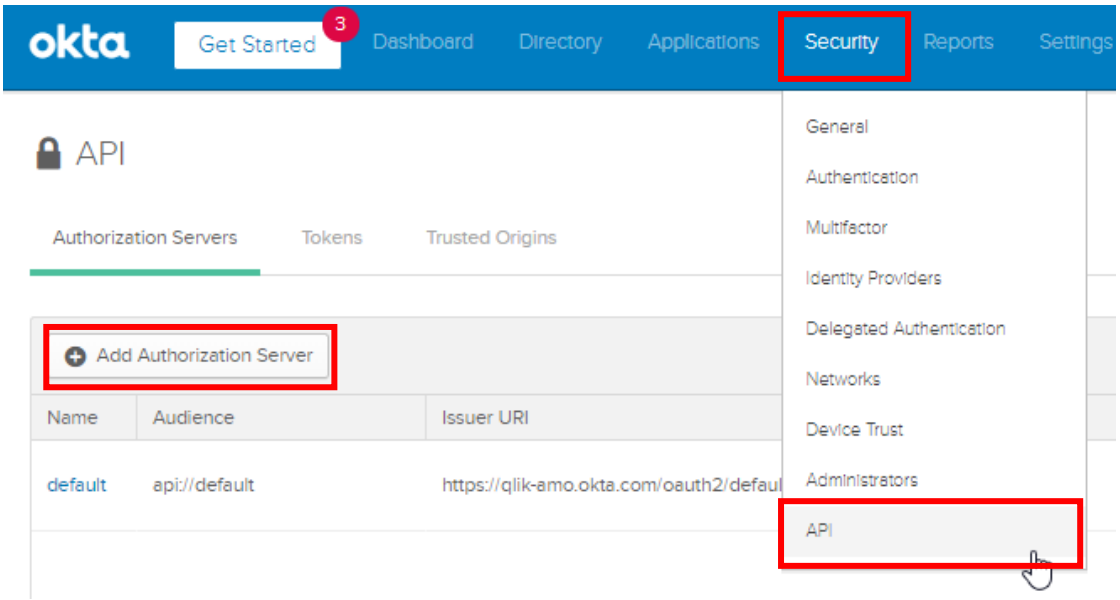
OKTA IdP を利用した Qlik Sense Enterprise on Windows から Kubernetes へのアプリの配信

Qlik Sense Enterprise マルチクラウド演習

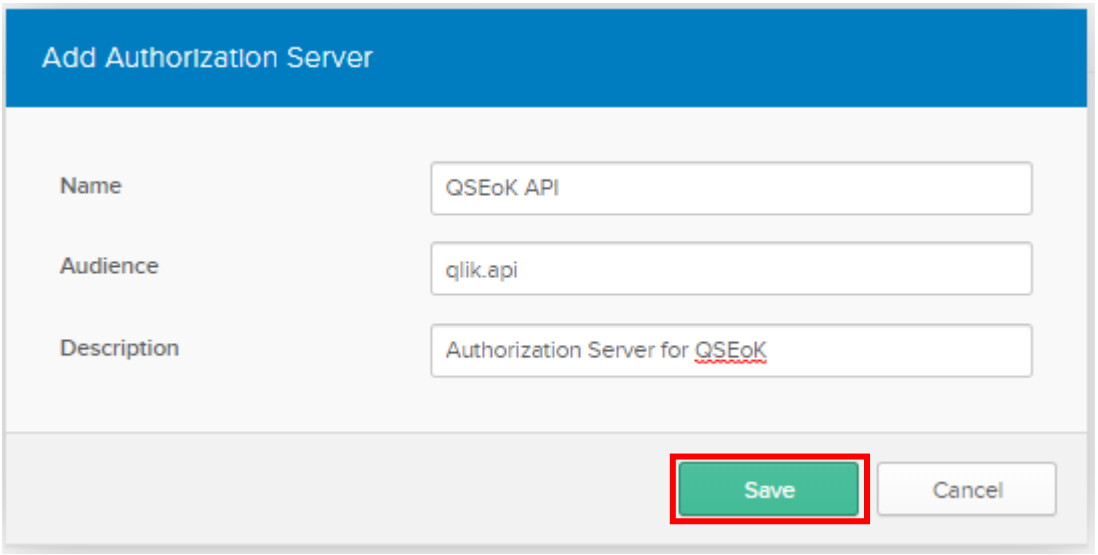
目次

1.OKTA 上での API の作成

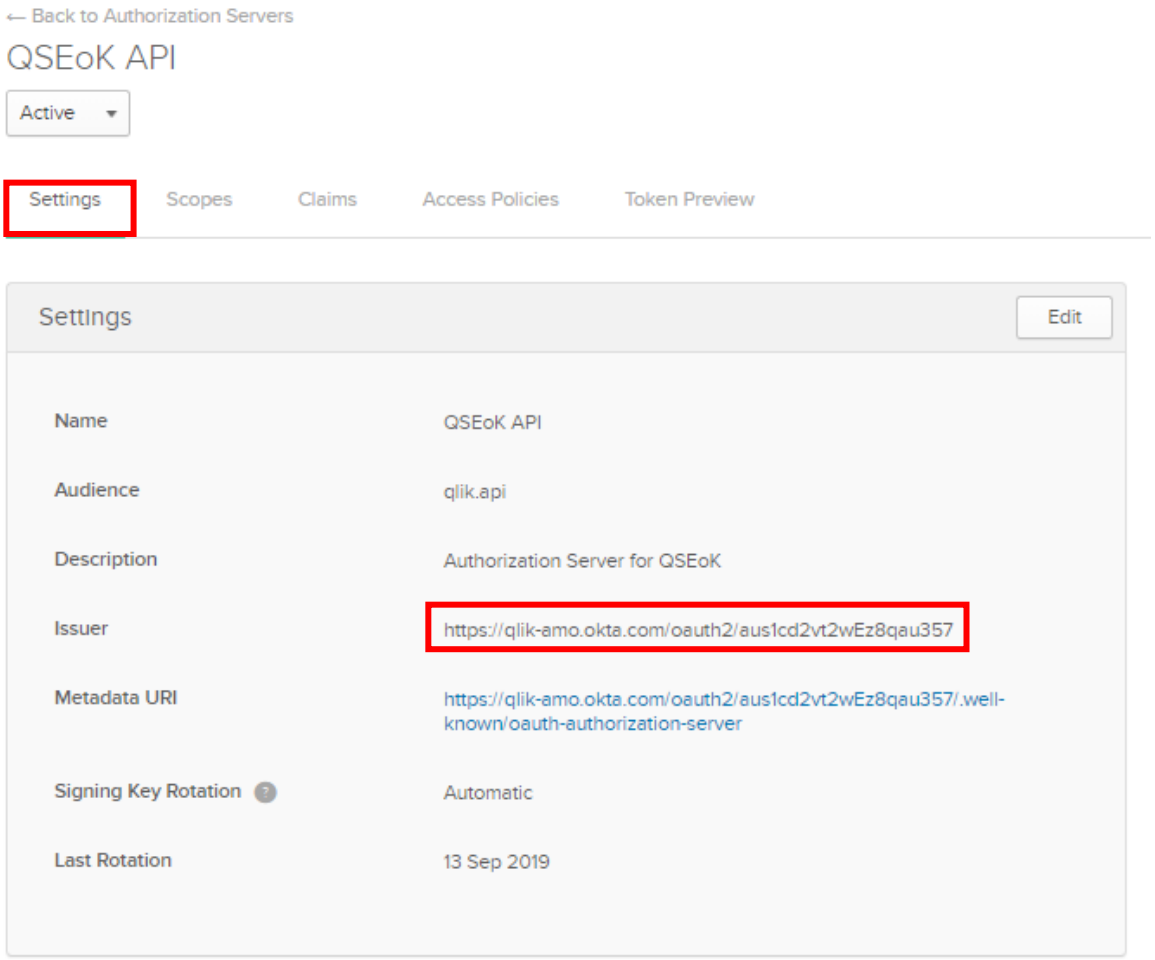
1) 「Security」>「API」を選択し、「Add Authorization Server」をクリックします。



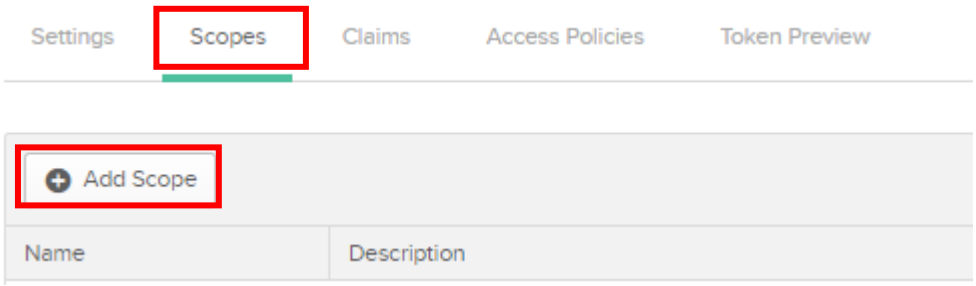
2) 以下の通り「Name」に「QSEoK API」、「Audience」に「qlik.api」と入力し、「Save」をクリックします。



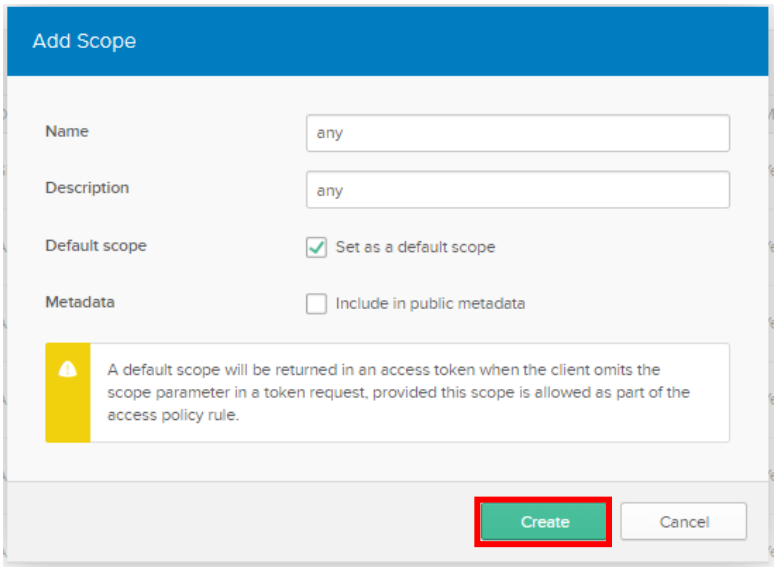
3) 「API Setting」上で「Issuer」をメモします。



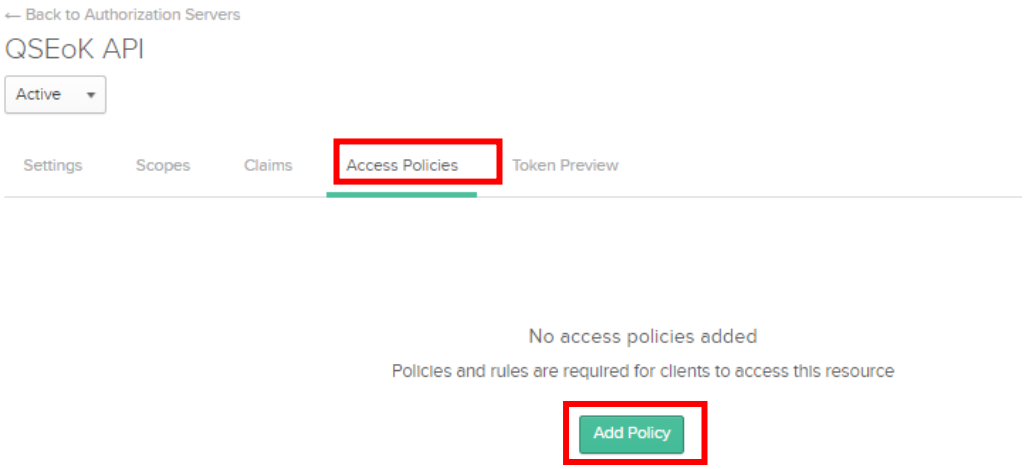
4) 「Scopes」タブをクリックして「Add Scope」をクリックします。



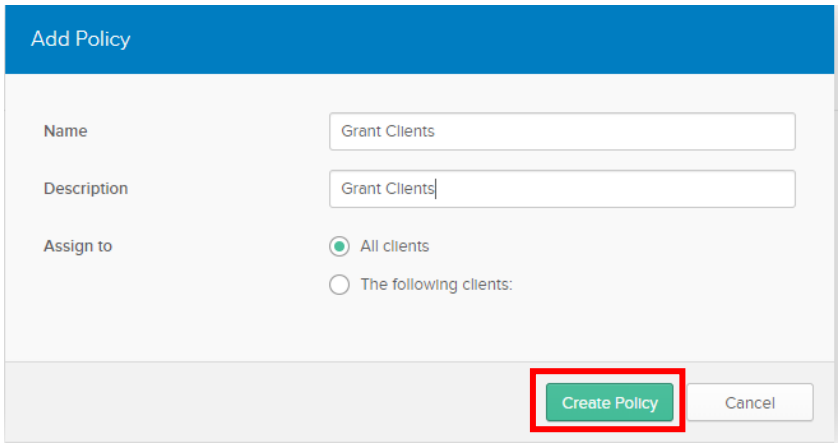
5) 以下の通り「any」と入力し、「Set as a default scope」を有効化して「Create」をクリックします。



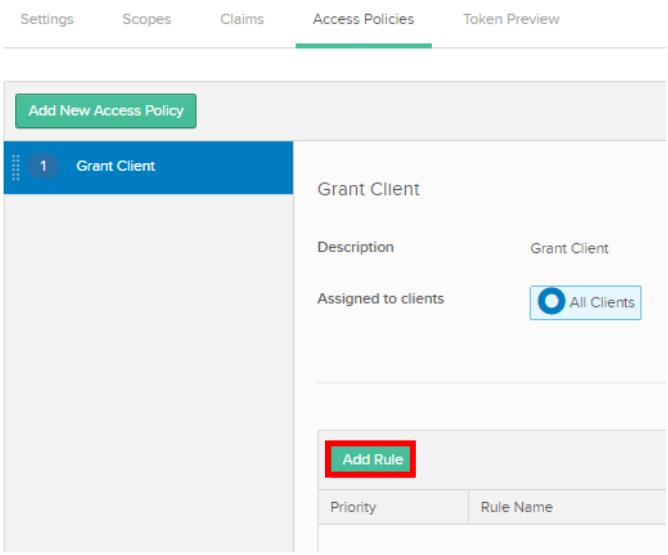
6) 「Access Policy」タブをクリックし、「Add Policy」をクリックします。



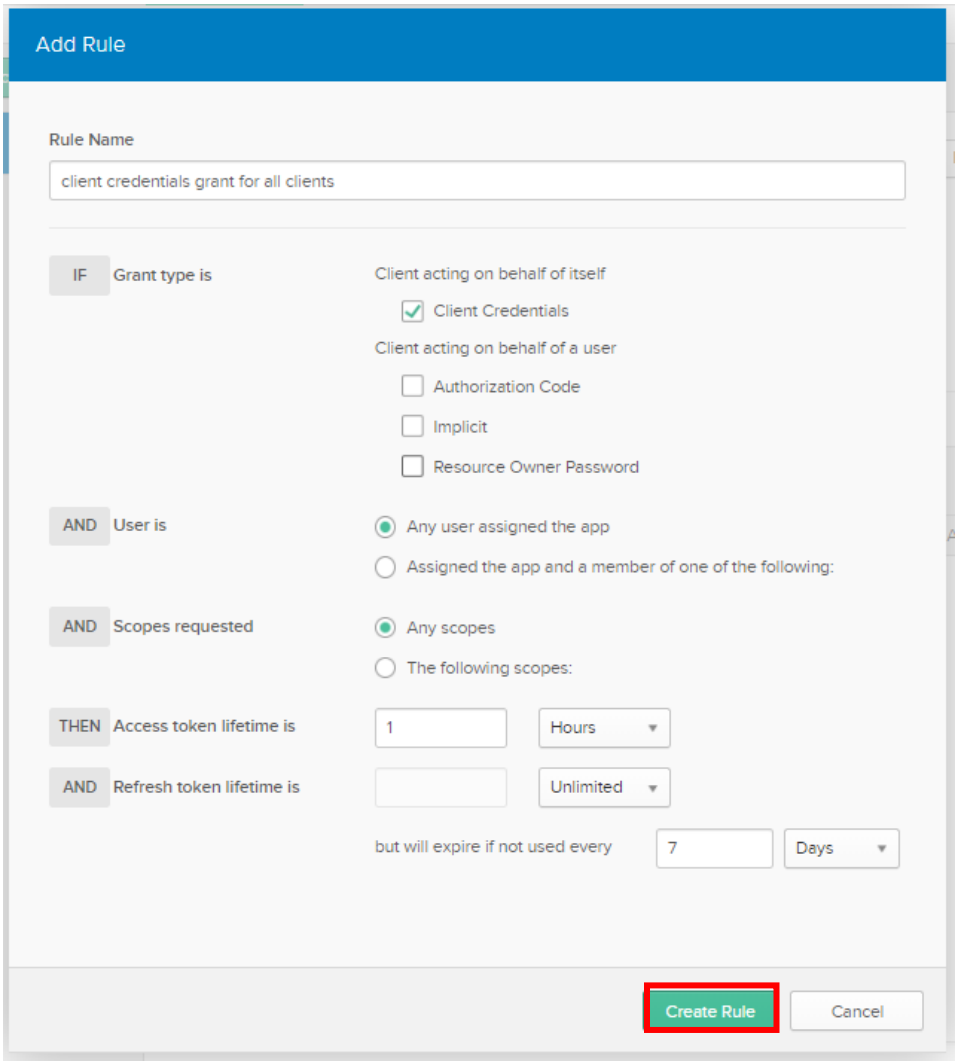
7) 以下の通り「Grant Clients」と入力して「All clients」を選択し、「Create Policy」をクリックします。



8) 「Add rule」をクリックします。

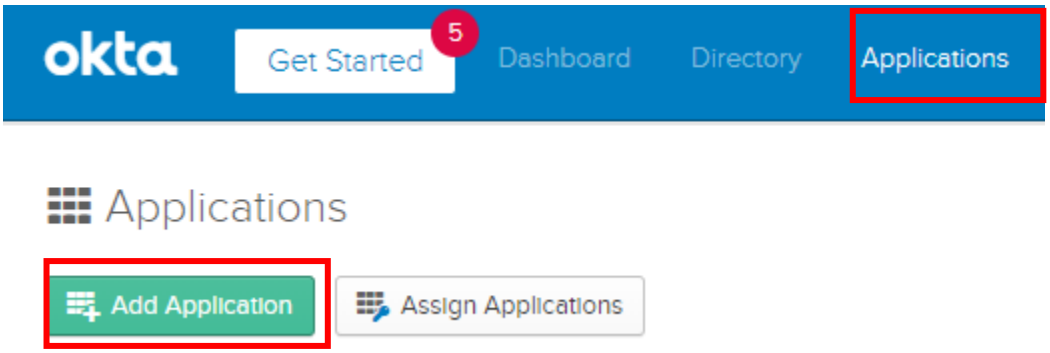


9) 「Name」に「client credentials grant for all clients」と入力して以下の通り設定し、「Create rule」をクリックします。

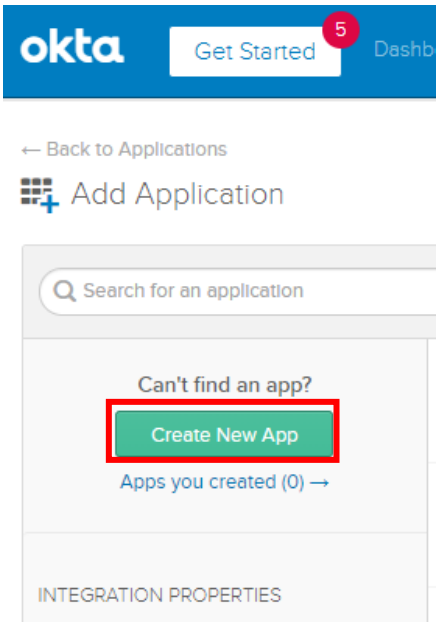


2.アプリケーションの作成

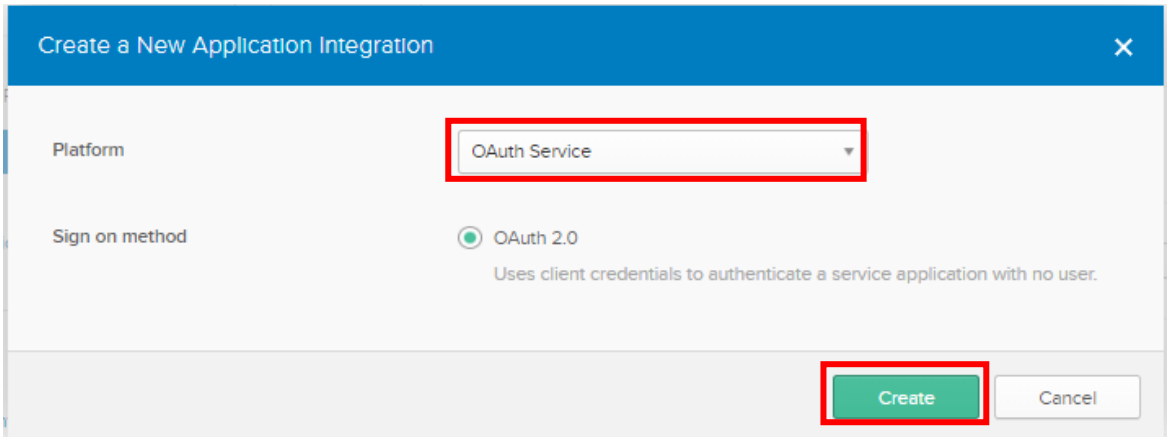
上部メニューから「Applications」を選択し、「Add Application」をクリックします。



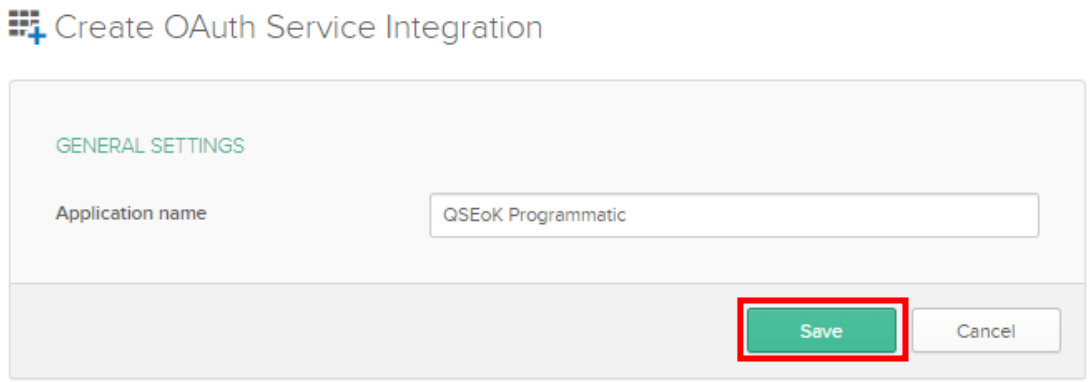
10)「Create New App」をクリックします。



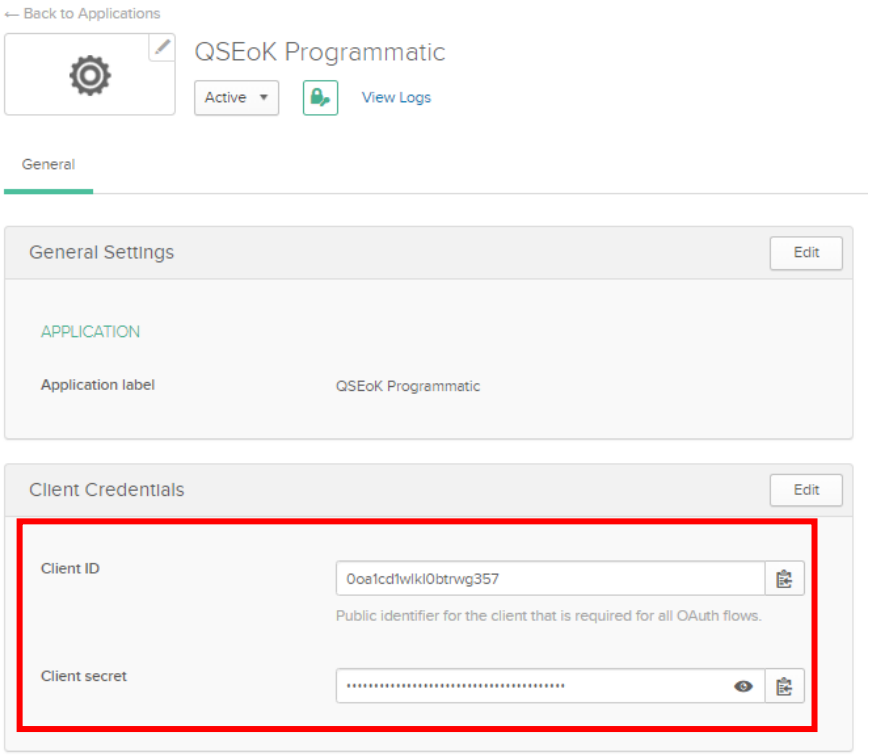
11)「Platform」は「OAuth Service」を選択し、「Create」をクリックします。



12) 「Application name」に「QSEoK Programmatic」と入力して「Save」をクリックします。



13) 表示される「Client ID」、「Client Secret」をメモします。



3.API の設定

ここでは Minikube を使った場合の Qlik Sense Enterprise on Kubernetes 側の設定方法をご説明します。Azure Kubernetes Service (AKS)や Amazon Elastic Kubernetes Service (AKS)などの設定を行う場合にはそれぞれのドキュメントをご参照ください。

1) この文書では以降の導入作業を root ユーザーで実施します。Windows ホスト上で PowerShell を開き、コンソールから以下で SSH アクセスを行います。パスワードには「osboxes.org」を入力します。

```
ssh root@elastic.example
```

2) エディタを利用して以下の内容の values-okta2.yaml ファイルを作成します。尚、以下は Minikube を利用した場合の例になりますので、Azure Kubernetes Service(AKS)や Amazon Elastic Kubernetes Service(EKS)などを利用するなどの場合には別紙をご参考の上 YAML ファイルの内容を変更してください。

前段の演習「01_QSE on Kubernetes の導入」で NFS Provisioner の設定を既に行っているか、いないかで利用すべきファイルの内容が異なります。(どちらを利用すべきか不明な場合は「① NFS Provisioner の設定を行っていない場合」をご利用ください。) また、黄色でハイライトされている「discoveryUrl」、「clientId」、「clientSecret」の値はこのドキュメントの前段の手順でメモした OKTA 上のアプリケーションの「Issuer」、及び「13_OKTA IdP を利用した Qlik Sense Enterprise on Kubernetes の設定」の中の手順で取得した「OKTA Homepage」、「Client ID」、「Client Secret」の値に基づいて書き換える必要があります。(ここで入力する「Client ID」、「Client Secret」は、ひとつ前の手順で取得した「QSEoK Programmatic」アプリケーションのものではありませんのでご注意ください。「QSEoK」アプリケーションのものを利用します。)

① NFS Provisioner の設定を行っていない場合

```
#This setting enables dev mode to include a local MongoDB install
devMode:
  enabled: true

#This setting accepts the EULA for the product
engine:
  acceptEULA: "yes"

# MINIKUBE SPECIFIC SETTINGS (dont not use with other K8
providers)_____
elastic-infra:
 nginx-ingress:
   controller:
     service:
       type: NodePort
       nodePorts:
         https: 32443
```

```
    extraArgs.report-node-internal-ip-address: ""

hub:
 ingress:
   annotations:
     nginx.ingress.kubernetes.io/auth-signin:
https://$host:32443/login?returnto=$request_uri

management-console:
 ingress:
   annotations:
     nginx.ingress.kubernetes.io/auth-signin:
https://$host:32443/login?returnto=$request_uri

edge-auth:
  oidc:
    redirectUri: https://elastic.example:32443/login/callback

#This setting is for IdP
identity-providers:
  secrets:
    idpConfigs:
      - discoveryUrl: "https://<OKTA Homepage>/.well-known/openid-configuration"
        clientId: "0oa1cciqlaHg3TXuE357"
        clientSecret : "KhsGaGm8U3lx8KFv-1xg0kzCKJxlj6bcSeKfmx1k"
        realm: "Okta"
        hostname: "elastic.example"
        scope: "openid profile groups"
       primary: true
      - discoveryUrl: "https://<Issuer>/.well-known/openid-configuration"
        realm: "Okta"
        hostname: "elastic.example"
        primary: false
        claimsMapping:
          client_id: [ "client_id", "cid" ]
```

② NFS Provisioner の設定を行っている場合

```
#This setting enables dev mode to include a local MongoDB install
devMode:
```

```
  enabled: true

#This setting accepts the EULA for the product
engine:
  acceptEULA: "yes"

#These setting specifies the storage for the services, in this case, NFS
global:
  persistence:
    storageClass: nfs-client

#MongoDB is set to persist with NFS storage so that data will not be lost if the
container is destroyed
mongodb:
  persistence:
    enabled: true
    accessMode: ReadWriteMany
    storageClass: nfs-client

# MINIKUBE SPECIFIC SETTINGS (dont not use with other K8
providers)_____
elastic-infra:
 nginx-ingress:
   controller:
     service:
       type: NodePort
       nodePorts:
         https: 32443
     extraArgs.report-node-internal-ip-address: ""

hub:
 ingress:
   annotations:
     nginx.ingress.kubernetes.io/auth-signin:
https://$host:32443/login?returnto=$request_uri

management-console:
 ingress:
   annotations:
```

```
    nginx.ingress.kubernetes.io/auth-signin:
https://$host:32443/login?returnto=$request_uri

edge-auth:
  oidc:
    redirectUri: https://elastic.example:32443/login/callback

#This setting is for IdP
identity-providers:
  secrets:
    idpConfigs:
      - discoveryUrl: "https://<OKTA Homepage>/.well-known/openid-configuration"
        clientId: "0oa1cciqlaHg3TXuE357"
        clientSecret : "KhsGaGm8U3lx8KFv-1xg0kzCKJxlj6bcSeKfmx1k"
        realm: "Okta"
        hostname: "elastic.example"
        scope: "openid profile groups"
        primary: true
      - discoveryUrl: "https://<Issuer>/.well-known/openid-configuration"
        realm: "Okta"
        hostname: "elastic.example"
        primary: false
        claimsMapping:
          client_id: [ "client_id", "cid" ]
```

3) Qlik Sense Enterprise on Kubernetes を展開します。

```
helm upgrade --install qliksense qlik/qliksense -f values-okta.yaml
```

4) 以下を実行し、展開した Pod の状況を確認します。STATUS が ContainerCreating から全て Running となるまで待ちます。

```
kubectl get pods
```

以下のコマンドを実行すると定期的な繰り返し処理で上記のコマンドを実行することも可能です。

```
while [ 1 ]; do kubectl get pods; sleep 5; clear; done
```

5) 以下のスクリプトの<Issuer >、<client id>、<client secret>の部分をメモした OKTA のアプリケーションの情報に置き換え、コンソールから実行します。(ここで入力する「Client ID」、「Client Secret」は、ひとつ前の手順で取得した「QSEoK Programmatic」アプリケーションのものとなります。)

```
curl --request POST ¥
--url https://<Issuer>/v1/token  ¥
-u '<client id>:<client secret>' ¥
--header 'content-type: application/x-www-form-urlencoded' ¥
--header 'accept: application/json' ¥
--data 'grant_type=client_credentials'
```

6) 以下のようなアクセストークンが返されますのでメモします。

```
{"token_type":"Bearer","expires_in":3600,"access_token":"eyJraWQiOiJPRDI4anRFOFRiejRDVHJBUGxlN24xeEFYSEtvbFNWWkZFMUZVdHhsaXg0IiwiYWxnIjoiUlMyNTYifQ.eyJ2ZXIiOjEsImp0aSI6IkFULmdmWC1ldk5YRXlGcDZUbHE3c3ZUWG1CQk9pcHltN0FSLTFGblJQci1uNU0iLCJpc3MiOiJodHRwczovL3FsaWstYW1vLm9rdGEuY29tL29hdXRoMi9hdXMxY2QydnQyd0V6OHFhdTM1NyIsImF1ZCI6InFsaWsuYXBpIiwiaWF0IjoxNTY4Mzc0MzIxLCJleHAiOjE1NjgzNzc5MjEsImNpZCI6IjBvYTFjZDF3bGtJMGJ0cndnMzU3Iiwic2NwIjpbImFueSJdLCJzdWIiOiIwb2ExY2Qxd2xrSTBidHJ3ZzM1NyJ9.O6VOIvjL025puqTp_oHMS5fxCUzFodyvsyCIDsGwddanMg2kaFcKIn1pyaI1ULQSTN5gVtuD2l4b7UXuk6Kov2wa1MEz-Mi8y5gW56ktO6EK2N4FOYpznOSvpjmBf_4dJ6fzZBmgP2M0Ub_s0Y39LZPiZeV2TRX-9tdk0J_xdc7GYULiJ1l9_2nCwcrUNhu03apAu04Yeg2j8Rae21LgTRfDqfUeE7zsISh3djRDPFec-O98lWO8N8-dAh1VPHodgy3YBAae6I-slgq-PRalPd3ncVg3TxZkKm3FA2gEUrY37V8mqddOhYxvw7F2H-lRLWbc8usHj3aQf5WFKOy9vQ","scope":"any"}[
```

7) 以下のスクリプトのアクセストークンを取得したアクセストークンで置き換え、コマンドを実行します。

```
curl -k --request GET ¥
--url https://elastic.example:32443/api/v1/collections  ¥
--header 'Authorization: Bearer
eyJraWQiOiJPRDI4anRFOFRiejRDVHJBUGxlN24xeEFYSEtvbFNWWkZFMUZVdHhsaXg0IiwiYWxnIjoiUlMyNTYifQ.eyJ2ZXIiOjEsImp0aSI6IkFULmdmWC1ldk5YRXlGcDZUbHE3c3ZUWG1CQk9pcHltN0FSLTFGblJQci1uNU0iLCJpc3MiOiJodHRwczovL3FsaWstYW1vLm9rdGEuY29tL29hdXRoMi9hdXMxY2QydnQyd0V6OHFhdTM1NyIsImF1ZCI6InFsaWsuYXBpIiwiaWF0IjoxNTY4Mzc0MzIxLCJleHAiOjE1NjgzNzc5MjEsImNpZCI6IjBvYTFjZDF3bGtJMGJ0cndnMzU3Iiwic2NwIjpbImFueSJdLCJzdWIiOiIwb2ExY2Qxd2xrSTBidHJ3ZzM1NyJ9.O6VOIvjL025puqTp_oHMS5fxCUzFodyvsyCIDsGwddanMg2kaFcKIn1pyaI1ULQSTN5gVtuD2l4b7UXuk6Kov2wa1MEz-
```

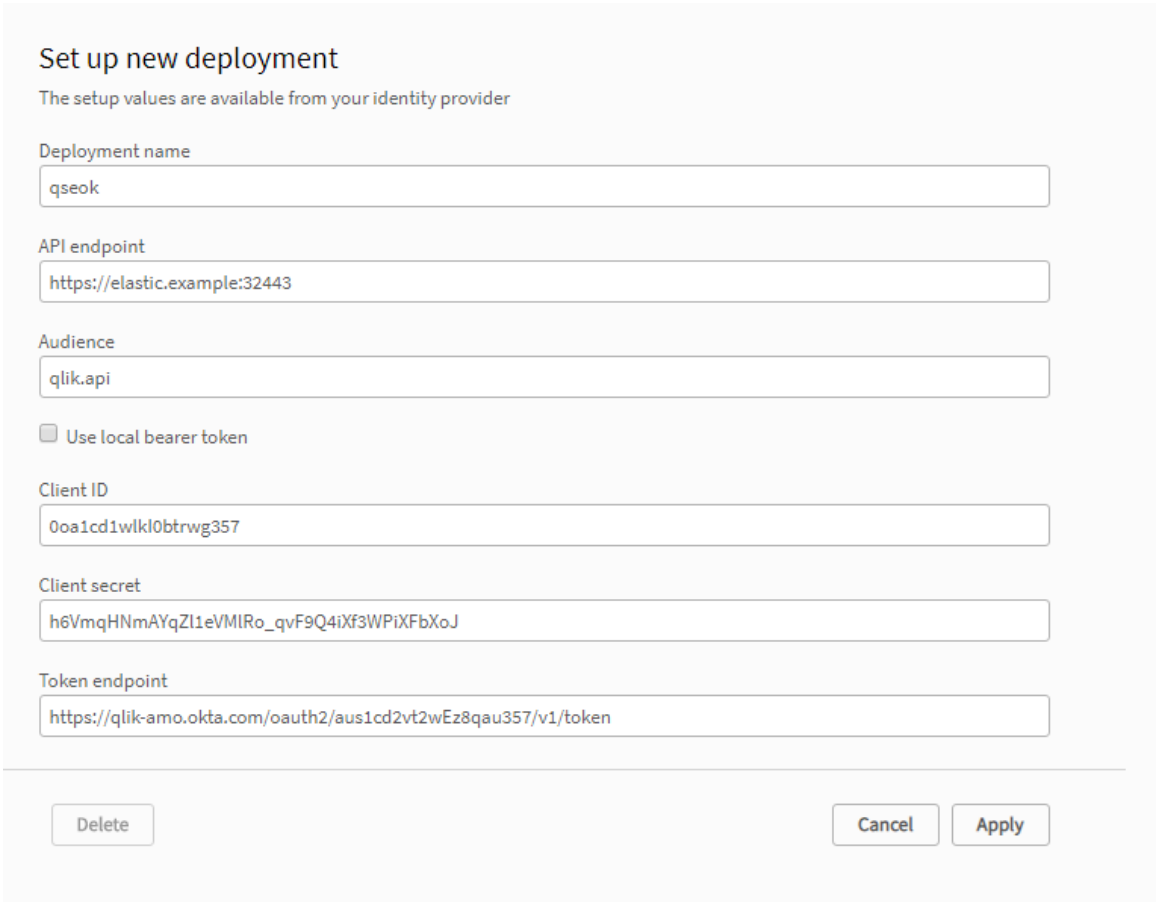
```
Mi8y5gW56ktO6EK2N4FOYpznOSvpjmBf_4dJ6fzZBmgP2M0Ub_s0Y39LZPiZeV2TRX-
9tdk0J_xdc7GYULiJ1l9_2nCwcrUNhu03apAu04Yeg2j8Rae21LgTRfDqfUeE7zsISh3djR
DPFec-O98lWO8N8-dAh1VPHodgy3YBAae6I-slgq-
PRalPd3ncVg3TxZkKm3FA2gEUrY37V8mqddOhYxvw7F2H-
lRLWbc8usHj3aQf5WFKOy9vQ'
```

認証が成功すると以下のようなデータが返されます。(失敗すると 401 Authorization Required のエラーとなります。)

```
{"data":[],"links":{"self":{"href":"https://elastic.example:32443/api/v1/collections"}}}
```

4.Qlik Sense Enterprise on Windows の Multi-Cloud Setup Console(MSC)でのアプリケーション配信設定

別紙「04_Qlik Sense Enterprise on Windows から Kubernetes へのアプリの配信」の 2 章～6 章の手順に従ってください。尚、Multi-Cloud Setup Console(MSC)上での OKTA の設定は以下の様になります。前段の手順でメモした OKTA 上のアプリケーションの「Issuer」、「Client ID」、「Client Secret」の値に基づいて、「Client ID」、「Client secret」、「Token endpoint」を置き換える必要があります。(ここで入力する「Client ID」、「Client Secret」は、ひとつ前の手順で取得した「QSEoK Programmatic」アプリケーションのものとなります。また「Token endpoint」は「Issuer」に「/v1/token」を付加したものとなります。)


Set up new deployment
The setup values are available from your identity provider
Deployment name
qseok
API endpoint
https://elastic.example:32443
Audience
qlik.api
Use local bearer token
Client ID
0oa1cd1wlkl0btrwg357
Client secret
h6VmqHNmAYqZl1eVMlRo_qvF9Q4iXf3WPiXFbXoJ
Token endpoint
https://qlik-amo.okta.com/oauth2/aus1cd2vt2wEz8qau357/v1/token
Delete
Cancel
Apply

About Qlik

Qlik is on a mission to create a data-literate world, where everyone can use data to solve their most challenging problems. Only Qlik's end-to-end data management and analytics platform brings together all of an organization's data from any source, enabling people at any skill level to use their curiosity to uncover new insights. Companies use Qlik products to see more deeply into customer behavior, reinvent business processes, discover new revenue streams, and balance risk and reward. Qlik does business in more than 100 countries and serves over 48,000 customers around the world.

qlik.com