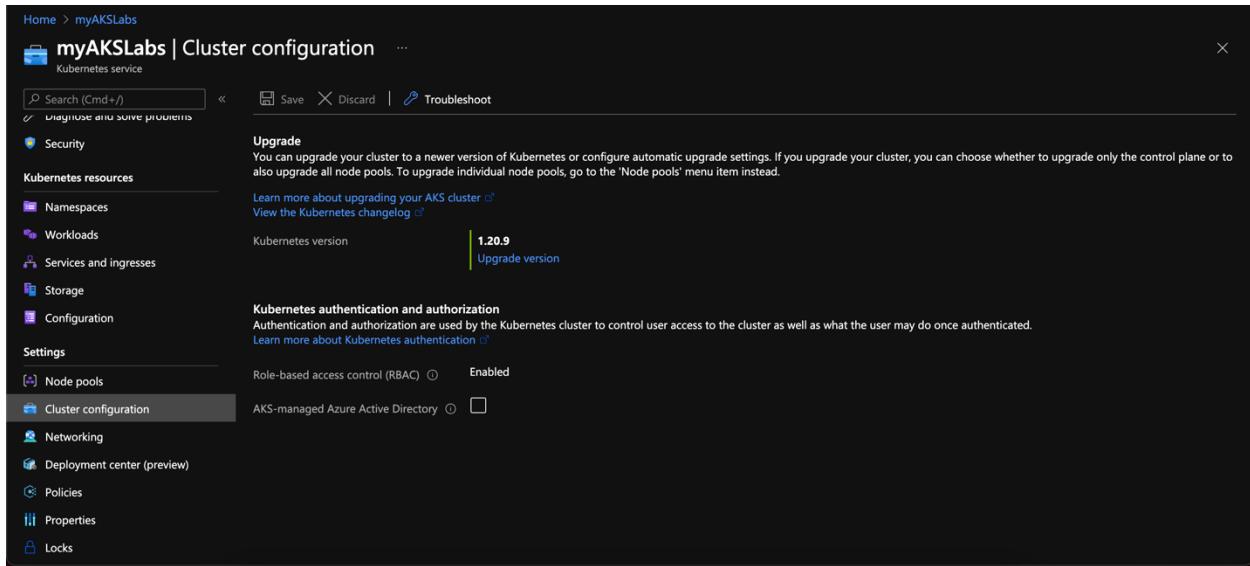


# Implementing Role Base Access Control (RBAC) in Azure Kubernetes Service (AKS) using Azure Active Directory (AAD)

## Kondisi Saat Ini:

AKS yang akan diimplementasikan RBAC dengan menggunakan AAD sudah menggunakan mode RBAC namun belum menggunakan AKS-managed Azure Active Directory sebagai autentikasinya:



Dengan begitu, saat ini setiap developer yang mengakses AKS masih memiliki role sebagai “**system:master**” sehingga bisa mengakses setiap resource yang ada di dalam Kubernetes Cluster. Untuk itu, diperlukan adanya implementasi RBAC pada Kubernetes Cluster agar akses yang dimiliki oleh Developer dapat dibatasi.

## Catatan Sebelum Implementasi:

- Ketika mode “AKS-managed Azure Active Directory” diaktifkan, maka tidak bisa di “**undo**” lg.
- Untuk para developer yang sebelumnya sudah connect, maka mereka masih akan memiliki akses penuh, sehingga para **developer harus melakukan “connect” ulang** agar konfigurasi sebelumnya diganti dengan credential yang menggunakan Azure Active Directory

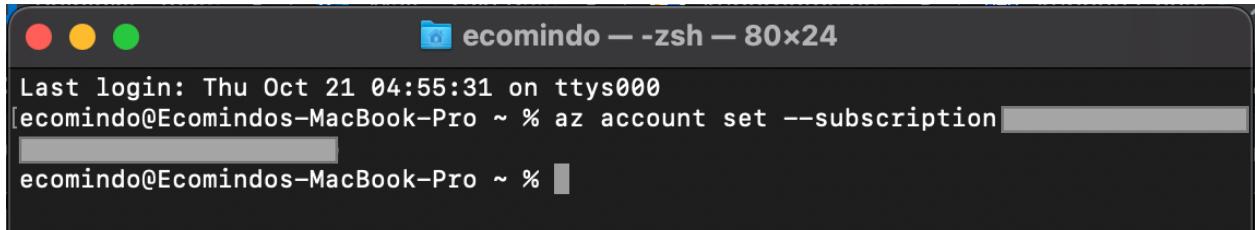
Langkah-langkah untuk mengaktifkan AKS-managed AAD:

1. Membuat Group di AAD sebagai Admin Group dalam Kubernetes Cluster:

1.1. Membuat group dengan menggunakan azure cli (command list):

- Set subscription id tempat group akan dibuat:

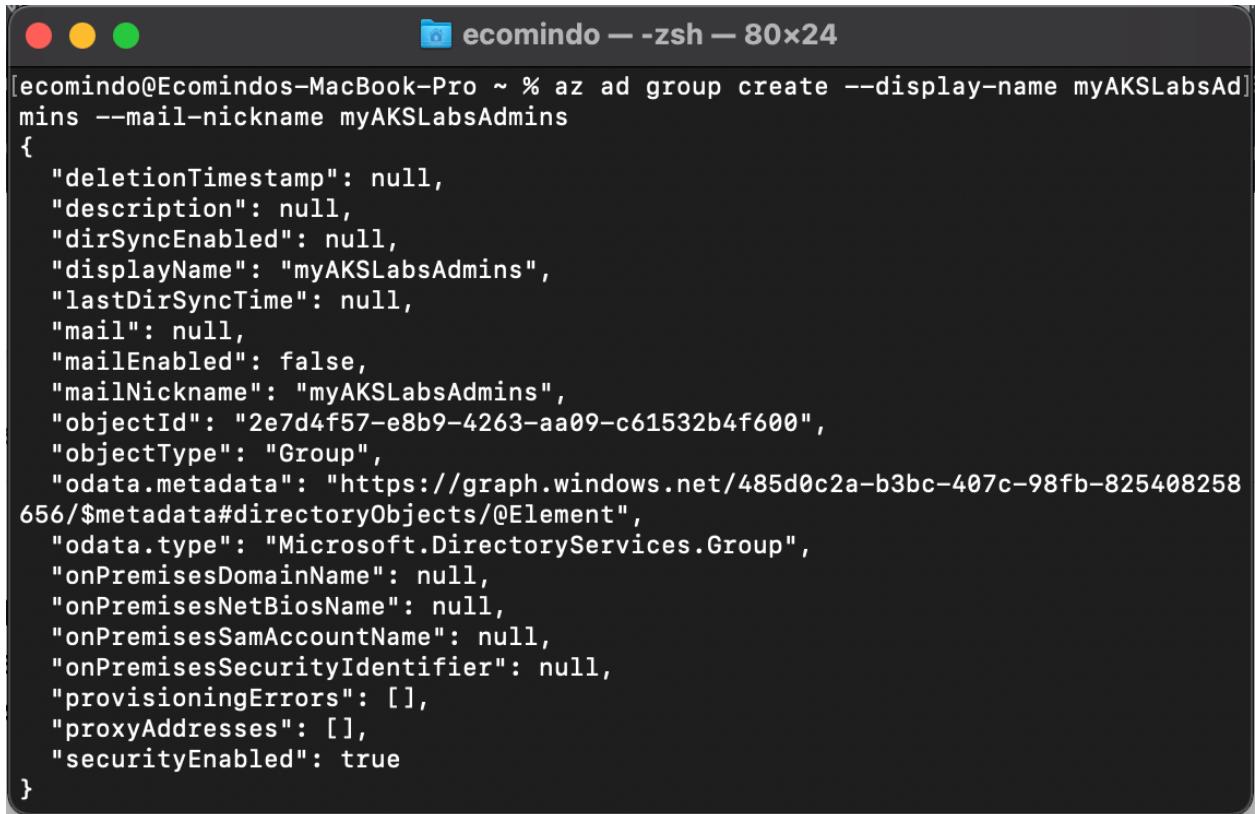
```
az account set --subscription <subscription id tempat group user group akan dibuat>
```



```
Last login: Thu Oct 21 04:55:31 on ttys000
[ecomindo@Ecomindos-MacBook-Pro ~ % az account set --subscription]
```

- Buat Group:

```
az ad group create --display-name <groupName> --mail-nickname <groupName>
```



```
[ecomindo@Ecomindos-MacBook-Pro ~ % az ad group create --display-name myAKSLabsAd
mins --mail-nickname myAKSLabsAdmins
{
  "deletionTimestamp": null,
  "description": null,
  "dirSyncEnabled": null,
  "displayName": "myAKSLabsAdmins",
  "lastDirSyncTime": null,
  "mail": null,
  "mailEnabled": false,
  "mailNickname": "myAKSLabsAdmins",
  "objectId": "2e7d4f57-e8b9-4263-aa09-c61532b4f600",
  "objectType": "Group",
  "odata.metadata": "https://graph.windows.net/485d0c2a-b3bc-407c-98fb-825408258
656/$metadata#directoryObjects/@Element",
  "odata.type": "Microsoft.DirectoryServices.Group",
  "onPremisesDomainName": null,
  "onPremisesNetBiosName": null,
  "onPremisesSamAccountName": null,
  "onPremisesSecurityIdentifier": null,
  "provisioningErrors": [],
  "proxyAddresses": [],
  "securityEnabled": true
}
```

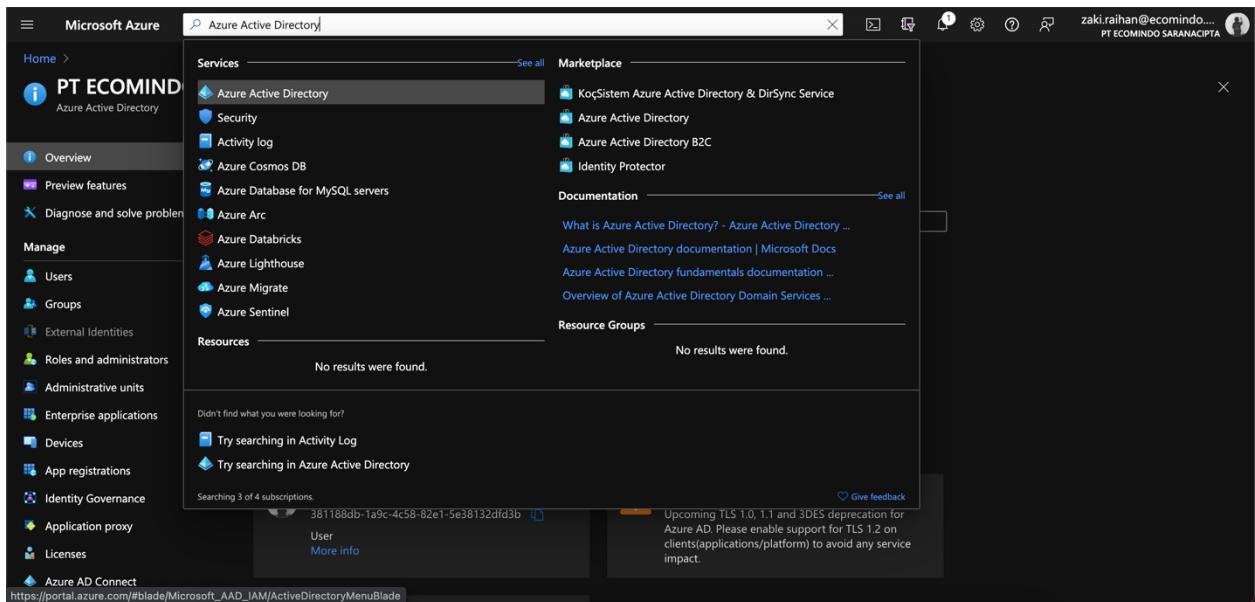
- Tambahkan member yang akan menjadi admin cluster ke dalam group yang baru dibuat:

```
az ad group member add --group <group Name> --member-id <member id>
```

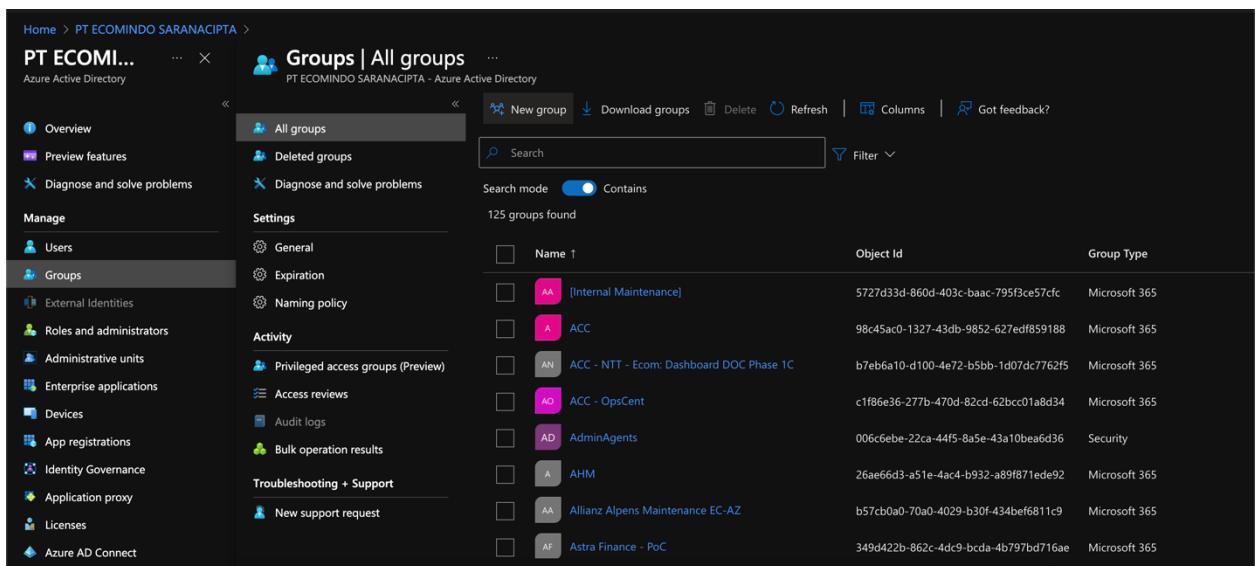
```
[ecomindo@Ecomindos-MacBook-Pro ~ % az ad group member add --group myAKSLabsAdmin  
s --member-id b0663142-3ba9-4ee3-ac90-4694ad104455  
ecomindo@Ecomindos-MacBook-Pro ~ % ]
```

## 1.2. Membuat group dengan menggunakan azure portal:

- Search Azure Active Directory pada search bar azure portal



- Pilih menu Groups pada bagian kiri Azure Active Directory, lalu pilih New Group



- Isi field sesuai kebutuhan. Untuk group type, pilih "Security". Pada bagian Owner, masukkan akun yang akan menjadi Owner dari Group. Pada bagian member, masukkan account yang akan menjadi admin di Kubernetes Cluster. Klik Create untuk membuat Group

New Group

Group type \* Security

Group name \* myAKSLabsAdmins

Group description Admins for myAKSLabs Cluster

Membership type Assigned

Owners  
1 owner selected

Members  
1 member selected

Create

- Group berhasil dibuat

Name	Object Id	Group Type	Membership Type	Email
myAKSLabsAdmins	38635e13-fca2-49b9-8f24-6b9908ed28a0	Security	Assigned	

- Jika ingin menambahkan admin di Kubernetes Cluster, buka Group yang dibuat, lalu pilih Add members, dan search akun yang ingin ditambahkan

The screenshot shows the 'Add members' dialog for the 'myAKSLabsAdmins' group in the Azure portal. The search bar at the top right contains the text 'zaki'. Below the search bar, a list of users is shown, with 'Zaki Raihan' and his email 'zaki.raihan@ecomindo.com' listed. At the bottom right of the dialog, there is a blue 'Select' button.

- Jika ingin mendapatkan Group Id bisa dilihat pada bagian Overview > Object Id

The screenshot shows the 'Overview' page for the 'myAKSLabsAdmins' group in the Azure portal. On the right side, under the 'Properties' section, there is a table with several fields. One of the fields is 'Object Id', which contains the value '38635e13-fca2-49b9-8f24-6b9908ed28a0'. This value is highlighted in the screenshot.

## 2. Mengaktifkan AKS-managed Azure Active Directory

- Masuk ke dalam AKS yang akan diaktifkan RBAC, pada menu Settings, pilih Cluster configuration

The screenshot shows the AKS Cluster configuration page. The left sidebar has 'Cluster configuration' selected. In the main area, under 'Kubernetes authentication and authorization', there is a section for 'AKS-managed Azure Active Directory' with an unchecked checkbox.

- Pilih opsi AKS-managed Azure Active Directory, lalu tambahkan Group yang sudah dibuat sebelumnya

The screenshot shows the AKS Cluster configuration page with 'Cluster configuration' selected. The 'AKS-managed Azure Active Directory' checkbox is checked. A modal window titled 'Add Azure AD groups' is open, showing a search bar with 'myAKS' and a list of groups including 'myAKSLabsAdmins'. A 'Select' button is at the bottom right of the modal.

- Setelah Group ditambahkan klik tombol Save untuk mengaplikasikan RBAC dengan menggunakan AAD

### 3. Memastikan RBAC dengan AKS-managed AAD sudah bekerja

- Untuk akun yang sudah ditambahkan ke dalam Group yang menjadi admin Kluster (dalam contoh ini group myAKSLabsAdmins) maka dapat mengakses semua Kubernetes resource pada Cluster melalui azure portal

Name	Namespace	Ready	Up-to-date	Available	Age
coredns	kube-system	✓ 2/2	2	2	3 hours
coredns-autoscaler	kube-system	✓ 1/1	1	1	3 hours
metrics-server	kube-system	✓ 1/1	1	1	3 hours
tunneelfront	kube-system	✓ 1/1	1	1	3 hours

- Untuk akun yang tidak ditambahkan ke dalam Group myAKSLabsAdmins (namun sudah diberikan akses kepada resource AKS), tidak akan bisa mengakses Kubernetes resource, sebelum dilakukan "RoleBinding" pada akunnya (**Untuk akun yang sudah melakukan "connect" pada Cluster sebelum AKS-managed AAD diaktifkan, maka "masih" bisa mengakses Cluster melalui kubectl dikarenakan credential yang tersimpan masih credential system:master**)

Forbidden (403)

deployments.apps is forbidden: User "08ad7cea-70e4-4aa6-9a96-e46e0afd0566" cannot list resource "deployments" in API group "apps" at the cluster scope. 'zaki.raihan98@gmail.com' does not have the required Kubernetes permissions to view this resource. Ensure you have the correct role/role binding for this user or group. [Learn more](#)

[Try again](#)

#### 4. Membuat Role di dalam Kubernetes Cluster untuk mengimplementasikan RBAC

Role-based access control (RBAC) atau kontrol akses berbasis *role* adalah metode pengaturan akses ke sumber daya komputer atau jaringan berdasarkan rol pengguna individu dalam organisasi.

Otorisasi RBAC pada kubernetes menggunakan grup API `rbac.authorization.k8s.io` untuk mengendalikan keputusan otorisasi. Hal ini memungkinkan kita untuk mengkonfigurasi kebijakan secara dinamis melalui API Kubernetes.

Pada Kubernetes sendiri terdapat dua jenis objek untuk melakukan pengaturan *role*, yaitu Role dan ClusterRole. Sebuah Role selalu mengatur izin dalam lingkup sebuah [Namespace](#) tertentu. ketika kita membuat Role, kita harus menentukan lingkup Namespace tempat Role tersebut akan berlaku. Sebaliknya, ClusterRole tidak terbatas pada sebuah lingkup Namespace. Perbedaan penamaan antara Role dan ClusterRole dikarenakan sebuah objek di dalam Kubernetes harus dibatasi oleh sebuah lingkup Namespace atau tidak sama sekali. Jika kita ingin mendefinisikan sebuah *role* dalam sebuah lingkup Namespace tertentu, maka kita akan menggunakan Role, namun jika kita ingin mendefinisikan *role* pada tingkat Cluster (tidak terbatas pada Namespace), maka kita menggunakan ClusterRole.

Contoh yaml file untuk membuat sebuah Role:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  namespace: default
  name: pod-reader
rules:
- apiGroups: [""]
  resources: ["pods"]
  verbs: ["get", "watch", "list"]
```

Keterangan:

I. namespace

Merupakan tempat pada Namespace *role* yang dibuat berlaku

II. name

Merupakan nama dari *role* yang dibuat. Kita bisa membuat *role* dengan nama yang sama namun di Namespace yang berbeda

III. rules

Merupakan aturan yang akan kita terapkan terhadap *role* yang akan dibuat. Satu *role* dapat memiliki beberapa rule

#### IV. apiGroups

Mengindikasikan API Group di dalam Kubernetes yang berkaitan dengan *resource* yang diatur dalam *role*. Jenis dari API Group yang mungkin ditambahkan adalah sebagai berikut:

- "" (string kosong)
- apps
- authentication.k8s.io
- authorization.k8s.io
- extensions
- rbac.authorization.k8s.io
- custom api groups (jika menggunakan custom resource)
- dan lainnya (lebih lengkap dapat diakses [disini](#))

#### V. resource

Merupakan *resource* yang diatur aksesnya dalam sebuah *rule* di dalam Role. Jenis resource yang dapat ditambahkan adalah sebagai berikut:

- pods
- replicsets
- deployments
- statefulsets
- daemonsets
- jobs
- cronjobs
- persistentvolumeclaims
- persistentvolumes
- storageclass
- secrets
- dan lainnya (lebih lengkap dapat diakses [disini](#))

#### VI. verbs

Merupakan *command* yang diatur aksesnya dalam sebuah *rule* di dalam Role. Jenis *verb* bergantung dari *resource* yang akan diatur, karena tidak semua *resource* bisa diplikasikan semua yang ada di dalam *verb*. Jenis *verb* yang dapat ditambahkan adalah sebagai berikut (selengkapnya dapat diakses [disini](#)):

- create
- delete
- deletecollection
- get
- list
- patch
- update
- watch

## VII. resourceName

Merupakan pengaturan yang spesifik mengatur akses spesifik terhadap sebuah *resource*. Contoh, jika kita ingin membatasi seseorang hanya boleh melakukan operasi *get* pada sebuah configmap yang bernama "my-config" maka kita bisa menambahkan *rule* seperti berikut:

```
rules:  
- apiGroups: [""]  
  resources: ["configmaps"]  
  resourceNames: ["my-config"]  
  verbs: ["get"]
```

## 5. Membuat RoleBinding di dalam Kubernetes Cluster untuk mengimplementasikan RBAC

RoleBinding memberikan izin yang ditentukan dalam sebuah Role kepada pengguna atau sekelompok pengguna. RoleBinding menyimpan daftar subject (User, Group, ServiceAccount) dan referensi kepada Role yang diberikan. RoleBinding memberikan izin dalam Namespace tertentu sedangkan ClusterRoleBinding memberikan akses tersebut pada lingkup klaster.

RoleBinding dapat merujuk Role apa pun di Namespace yang sama. Atau, RoleBinding dapat mereferensikan ClusterRole dan memasangkan ClusterRole tersebut ke Namespace dari RoleBinding. Jika kita ingin memasangkan ClusterRole ke semua Namespace di dalam klastermu, kita dapat menggunakan ClusterRoleBinding.

Contoh yaml file untuk membuat sebuah RoleBinding:

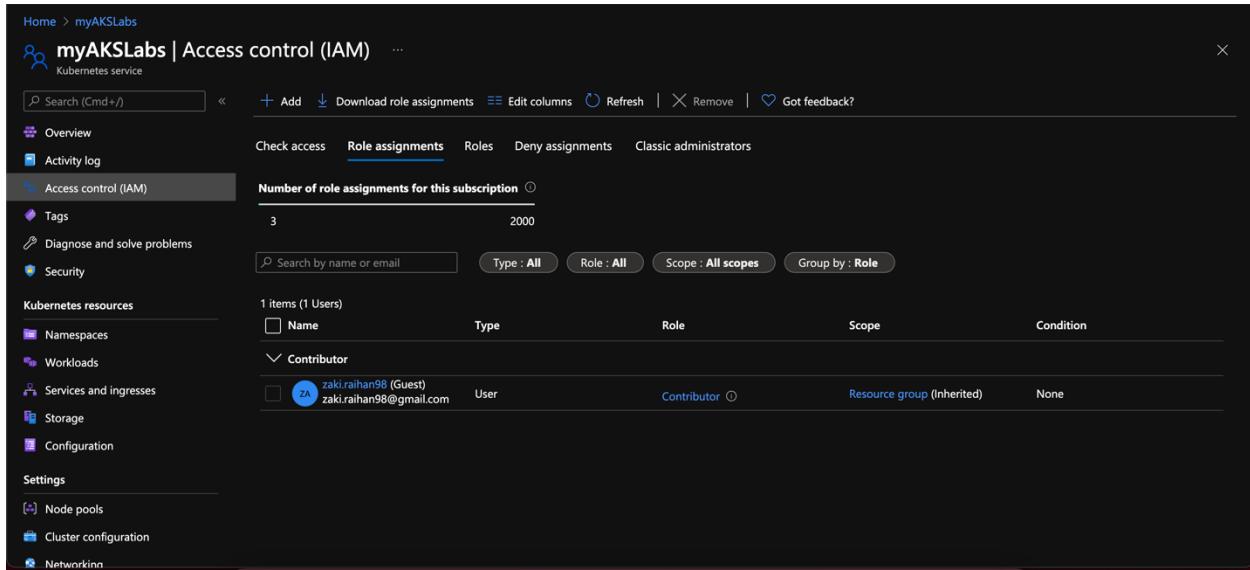
```
apiVersion: rbac.authorization.k8s.io/v1
# RoleBinding memungkinkan "jane" untuk membaca Pod di Namespace "default"
# Kita harus sudah membuat Role bernama "pod-reader" di Namespace tersebut.
kind: RoleBinding
metadata:
  name: read-pods
  namespace: default
subjects:
# Kita bisa mencantumkan lebih dari satu "subjek"
- kind: User
  name: jane # "name" peka huruf besar-kecil
  apiGroup: rbac.authorization.k8s.io
roleRef:
  # "roleRef" menentukan pengikatan (binding) ke Role / ClusterRole
  kind: Role # ini harus Role atau ClusterRole
  name: pod-reader # ini harus sesuai dengan nama Role atau ClusterRole
yang ingin kita gunakan
  apiGroup: rbac.authorization.k8s.io
```

Pada AKS nantinya, untuk "name" pada subjects dengan kind "User" dapat isi dengan email yang sudah diberikan akses ke dalam resource AKS, atau member-id (jika email yang didaftarkan merupakan *guest* di Azure Active Directory).

Untuk subjects dengan kind "Group", kita bisa menggunakannya dengan cara membuat group baru di Azure Active Directort (mirip seperti membuat group admin), namun kali ini group diisi oleh akun yang akan diberikan RoleBinding dan pada bagian "name" bisa diisi oleh Object Id dari group tersebut (bisa dilihat di Azure Active Directory).

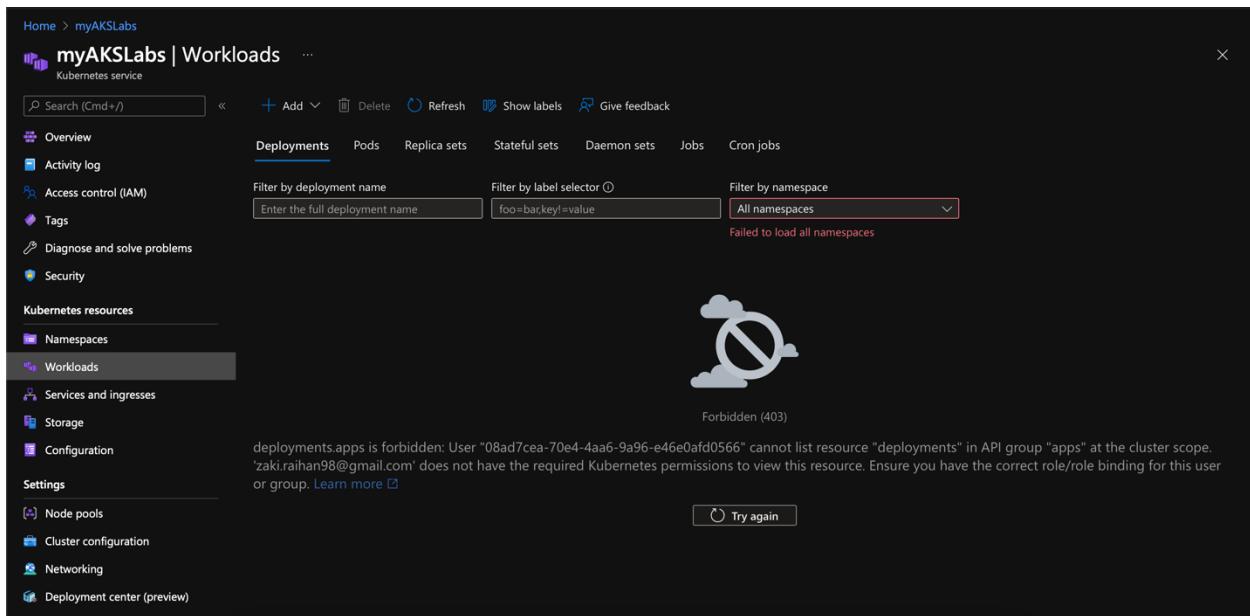
## 6. Contoh Studi Kasus

Pada studi kasus ini, kita akan menggunakan Cluster yang sudah kita setting di atas (Cluster myAKSLabs). Disini kita menambahkan sebuah akun baru milik Zaki dengan email "zaki.raihan98@gmail.com" sebagai seorang developer. Dikarenakan akun tersebut merupakan akun di luar organisasi, maka akun tersebut bersifat Guest.



The screenshot shows the 'Role assignments' tab in the IAM section of the myAKSLabs Kubernetes service. It displays a table with one item, 'zaki.raihan98 (Guest)', which is a User type with a Contributor role, inheriting the Resource group scope. The sidebar on the left includes options like Overview, Activity log, and Workloads.

Ketika Zaki mencoba mengakses resource pada AKS, Zaki mendapatkan pesan error seperti berikut:



The screenshot shows the 'Deployments' tab in the Workloads section of the myAKSLabs Kubernetes service. A 'Forbidden (403)' error message is displayed, stating that the user does not have the required permissions to list deployments. The sidebar on the left includes options like Overview, Activity log, and Workloads.

Terlihat bahwa dikarenakan Zaki menggunakan email diluar organisasi, maka User yang terdaftar menggunakan GUID (yang merupakan Object Id dari user, bisa dilihat lebih detail melalui Azure Active Directory) seperti yang digaris bawah pada contoh.

```
deployments.apps is forbidden: User "08ad7cea-70e4-4aa6-9a96-e46e0afd0566" cannot
list resource "deployments" in API group "apps" at the cluster scope.
'zaki.raihan98@gmail.com' does not have the required Kubernetes permissions to view
this resource. Ensure you have the correct role/role binding for this user or group.
```

Jika yang kita tambahkan menggunakan email yang terdaftar di organisasi (bukan Guest), maka User akan menggunakan email sesuai dengan yang digunakan (Dalam contoh User menjadi "fauzan.dika@ecomindo.com").

The screenshot shows the Kubernetes UI interface for managing Deployments. At the top, there are tabs for Deployments, Pods, Replica sets, Stateful sets, Daemon sets, Jobs, and Cron jobs. The Deployments tab is selected. Below the tabs are three filter inputs: 'Filter by deployment name' (with placeholder 'Enter the full deployment name'), 'Filter by label selector' (with placeholder 'foo=bar/key1=value'), and 'Filter by namespace' (with dropdown set to 'All namespaces'). A red box highlights the 'All namespaces' dropdown. Below the filters, a message says 'Failed to load all namespaces'. In the center, there is a large 'Forbidden (403)' error icon, which is a cloud with a slash through it. Below the icon, the text 'Forbidden (403)' is displayed. At the bottom of the error area, a message states: 'deployments.apps is forbidden: User "fauzan.dika@ecomindo.com" cannot list resource "deployments" in API group "apps" at the cluster scope. "fauzan.dika@ecomindo.com" does not have the required Kubernetes permissions to view this resource. Ensure you have the correct role/role binding for this user or group.' A blue 'Learn more' link is present. At the very bottom, there is a 'Try again' button with a circular arrow icon.

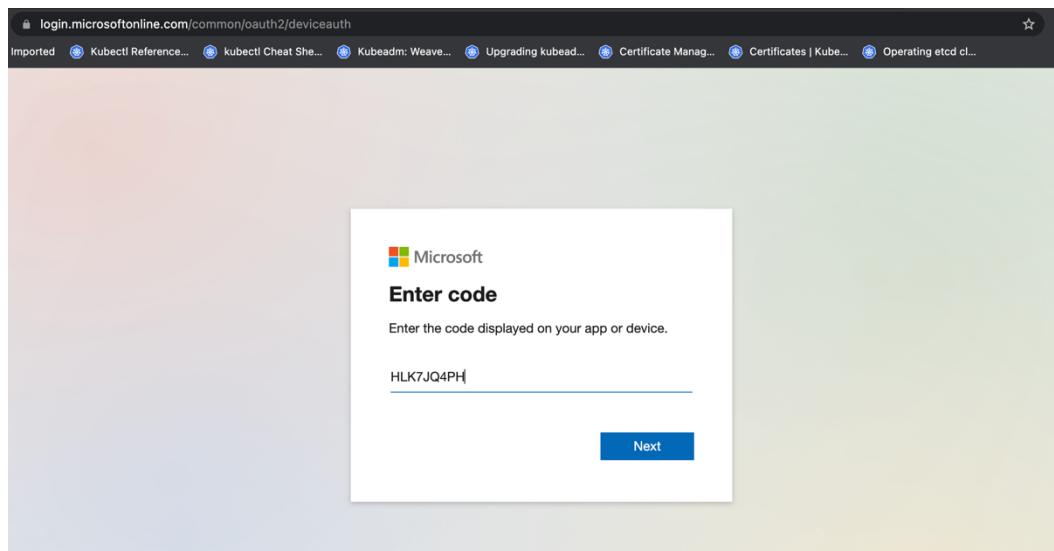
Dikarenakan Zaki merupakan seorang Developer, maka Kubernetes Admin (akun yang terdaftar pada Group myAKSLabsAdmins) membuatkan sebuah *role* untuk Zaki, yaitu "developer". Untuk saat ini *role* "developer" diberikan akses untuk melihat, membuat, dan menghapus beberapa resource yaitu Pod, ReplicaSet, dan Deployment yang ada pada Namespace "dev".

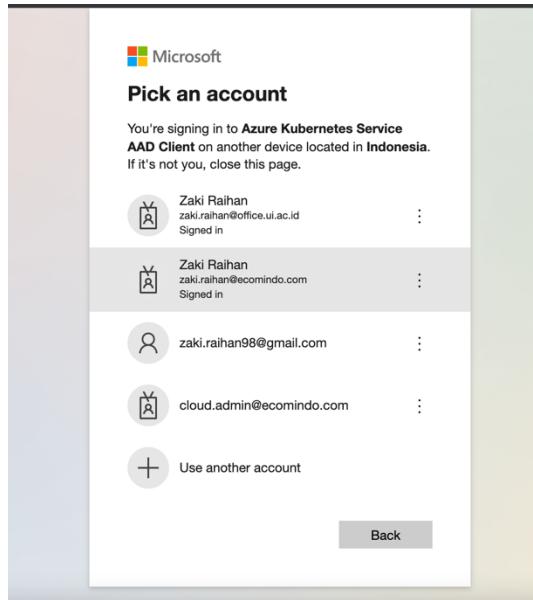
Agar Kubernetes Admin bisa menambahkan Role ke dalam Kubernetes Cluster, maka Kubernetes Admin dapat membuat koneksi dengan AKS dengan menggunakan "kubectl". Pertama, Kubernetes Admin melakukan "connect" seperti yang dicontohkan oleh Azure.

The screenshot shows the Azure portal interface. On the left, there's a sidebar with 'myAKSLabs' selected as a Kubernetes service. The main area shows the 'Overview' tab for the 'myAKSLabs' cluster. It displays basic information like Resource group (myAKSLabs), Status (Succeeded (Running)), Location (Southeast Asia), Subscription (Azure for Students), and Tags. Below this, under 'Kubernetes resources', there are sections for Namespaces, Workloads, Services and ingresses, Storage, Configuration, Node pools, Cluster configuration, and Networking. The 'Properties' tab is currently selected. On the right, a modal window titled 'Connect to myAKSLabs' provides instructions on how to connect using command line tools like kubectl or the Azure CLI. It includes sample commands for listing deployments, pods, and details about specific deployments. The modal also contains a 'Give feedback' button.

Lalu, Kubernetes Admin dapat menggunakan command contoh seperti "kubectl get all" dan nantinya akan diminta untuk login. Masuk ke url yang diberikan, masukkan code, dan login menggunakan akun azure (yang sudah didaftarkan pada Group myAKSLabsAdmins) :

```
ecomindo@Ecomindo-MacBook-Pro: ~ % kubectl get all
To sign in, use a web browser to open the page https://microsoft.com/devicelogin and enter the code HLK7JQ4PH to authenticate.
```





Setelah itu, Kubernetes Admin dapat menggunakan command "kubectl" untuk mengelola Cluster. Selanjutnya dibuatlah sebuah file yaml dengan nama developer-role.yaml seperti berikut:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  namespace: dev
  name: developer
rules:
- apiGroups:
  - ""
  - apps
  resources:
  - pods
  - deployments
  - replicasesets
  verbs:
  - get
  - list
  - create
  - delete
```

Setelah itu, Kubernetes Admin menambahkan *role* ke dalam kubernetes cluster dengan menggunakan command:

```
kubectl apply -f developer-role.yaml
```

*role* "developer" untuk Namespace "dev" berhasil ditambahkan

```
ecomindo@Ecomindos-MacBook-Pro yaml % kubectl apply -f developer-role.yaml
role.rbac.authorization.k8s.io/developer created
ecomindo@Ecomindos-MacBook-Pro yaml % kubectl get role -A
NAMESPACE      NAME                                     CREATED AT
dev            developer                               2021-10-22T07:49:53Z
kube-public     system:controller:bootstrap-signer   2021-10-22T03:48:19Z
kube-system    extension-apiserver-authentication-reader 2021-10-22T03:48:19Z
kube-system    system::leader-locking-kube-controller-manager 2021-10-22T03:48:19Z
kube-system    system::leader-locking-kube-scheduler   2021-10-22T03:48:19Z
kube-system    system:controller:bootstrap-signer   2021-10-22T03:48:19Z
kube-system    system:controller:cloud-provider       2021-10-22T03:48:19Z
kube-system    system:controller:token-cleaner      2021-10-22T03:48:19Z
kube-system    tunnelfront-secret                   2021-10-22T03:48:56Z
ecomindo@Ecomindos-MacBook-Pro yaml %
```

Selain itu Kubernetes Admin juga ingin memberikan akses kepada developer, agar bisa melihat semua Namespace yang ada di Kubernetes Cluster (get dan list) untuk itu, Kubernetes Admin membuat sebuah Cluster Role "developer-clusterrole.yaml" seperti berikut:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: developer
rules:
- apiGroups:
  - ""
  resources:
  - namespaces
  verbs:
  - list
```

## 6.1. Melakukan RoleBinding dan ClusterRoleBinding dengan menenggunakan User

Untuk melakukan RoleBinding pada sebuah User kita dapat membuat yaml file seperti berikut:

```
# developer-rolebinding.yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  creationTimestamp: null
  name: developer-binding
  namespace: dev
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: developer
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: User
  name: 08ad7cea-70e4-4aa6-9a96-e46e0afd0566 #objectId user, dikarenakan akun Guest
```

Untuk mengaplikasikan RoleBinding, bisa menggunakan command:

```
kubectl apply -f developer-rolebinding.yaml
```

Setelah itu maka akun milik Zaki (zaki.raihan98@gmail.com) bisa mengakses Kubernetes Cluster sesuai dengan yang telah didefinisikan di dalam Role developer (mengakses pods, replicaset, deployment di namespace dev, namun tidak di namespace lainnya).

```
zaki_raihan98@Azure:~$ kubectl get pods,replicasets,deployment -n dev
NAME           READY   STATUS    RESTARTS   AGE
pod/nginx-6799fc88d8-lfjzh   0/1     ContainerCreating   0          6s

NAME             DESIRED   CURRENT   READY   AGE
replicaset.apps/nginx-6799fc88d8  1         1         0        6s

NAME           READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/nginx   0/1       1           0          7s
zaki_raihan98@Azure:~$ kubectl get pods,replicasets,deployment -n default
Error from server (Forbidden): pods is forbidden: User "08ad7cea-70e4-4aa6-9a96-e46e0afd0566" cannot list resource "pods" in API group "" in the namespace "default"
Error from server (Forbidden): replicaset.apps is forbidden: User "08ad7cea-70e4-4aa6-9a96-e46e0afd0566" cannot list resource "replicaset" in API group "apps" in the namespace "default"
Error from server (Forbidden): deployment.apps is forbidden: User "08ad7cea-70e4-4aa6-9a96-e46e0afd0566" cannot list resource "deployment" in API group "apps" in the namespace "default"
```

Untuk melakukan ClusterRoleBinding cara yang dilakukan sama, Kubernetes Admin dapat membuat file yaml seperti berikut:

```
# developer-clusterrolebinding.yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  creationTimestamp: null
  name: developer-binding
  namespace: dev
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: developer
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: User
  name: 08ad7cea-70e4-4aa6-9a96-e46e0afd0566 #objectId user, dikarenakan akun Guest
```

Untuk mengaplikasikan ClusterRoleBinding, bisa menggunakan command:

```
kubectl apply -f developer-clusterrolebinding.yaml
```

Setelah itu maka akun milik Zaki (zaki.raihan98@gmail.com) bisa melihat semua Namespace yang ada di dalam Kubernetes Cluster, namun tidak bisa membuat baru.

```
zaki_raihan98@Azure:~$ kubectl get namespace
NAME      STATUS   AGE
default   Active  6h46m
dev       Active  171m
kube-node-lease  Active  6h46m
kube-public  Active  6h46m
kube-system  Active  6h46m
zaki_raihan98@Azure:~$ kubectl create namespace zakinew
Error from server (Forbidden): namespaces is forbidden: User "08ad7cea-70e4-4aa6-9a96-e46e0afd0566" cannot create resource "namespaces" in API group
"" at the cluster scope
zaki_raihan98@Azure:~$
```

Untuk melakukan maintenance terkait RoleBinding dan ClusterRoleBinding, seperti memberi akses dan menghapus akses, Kubernetes Admin dapat menambahkan atau menghapus subjects yang ada di dalam RoleBinding yang telah dibuat dan bisa kembali menjalankan command "kubectl apply -f developer-rolebinding.yaml". Hal yang sama juga berlaku untuk Role and ClusterRole, Kubernetes Admin dapat mengubah file yaml dan langsung menggunakan command "apply" untuk mengubah Role and ClusterRole di dalam cluster

## 6.2. Melakukan RoleBinding dan ClusterRoleBinding dengan menenggunakan Group

Melakukan RoleBinding dan ClusterRoleBinding dengan menggunakan Group sama seperti dengan menggunakan User. Hanya saja, pada bagian "subjects" kind yang digunakan ada Group dan "name" diisi dengan group id. Pada bagian ini, proses pembuatan Group akan menggunakan azure cli, namun kita bisa juga menggunakan Azure Portal dengan langkah-langkah yang sudah dijelaskan pada [bagian 1](#).

Pertama kita perlu membuat sebuah Group baru. Sebelumnya kita telah membuat myAKSLabsAdmins sebagai group yang berisikan akun-akun yang dijadikan Kubernetes Admin. Sekarang kita akan membuat group baru yaitu "myAKSLabsDeveloper" yang akan berisikan akun yang memiliki role "developer" di dalam Kubernetes Cluster.

```
ecomindo@Ecomindos-MacBook-Pro yaml % az ad group create --display-name myAKSLabsDeveloper --mail-nickname myAKSLabsDeveloper
{
  "deletionTimestamp": null,
  "description": null,
  "dirSyncEnabled": null,
  "displayName": "myAKSLabsDeveloper",
  "lastDirSyncTime": null,
  "mail": null,
  "mailEnabled": false,
  "mailNickname": "myAKSLabsDeveloper",
  "objectId": "3bb59bf9-d05d-45fb-92e7-d6f0f3a04372",
  "objectType": "Group",
  "odata.metadata": "https://graph.windows.net/485d0c2a-b3bc-407c-98fb-825408258656/$metadata#directoryObjects/@Element",
  "odata.type": "Microsoft.DirectoryServices.Group",
  "onPremisesDomainName": null,
  "onPremisesNetBiosName": null,
  "onPremisesSamAccountName": null,
  "onPremisesSecurityIdentifier": null,
  "provisioningErrors": [],
  "proxyAddresses": [],
  "securityEnabled": true
}
ecomindo@Ecomindos-MacBook-Pro yaml %
```

Simpan Object Id dari Group. Object Id dari Group (atau Group Id) akan digunakan sebagai parameter name pada RoleBinding nantinya.

Setelah itu tambahkan akun developer ke dalam myAKSLabsDeveloper

```
az ad group member add --group myAKSLabsDeveloper --member-id 08ad7cea-70e4-4aa6-9a96-e46e0afd0566
```

Pada file yaml developer-rolebinding.yaml dan developer-clusterrolebinding.yaml ubah subjects menggunakan "kind: Group" dan "name: <objectId dari Group>" (Untuk Role dan ClusterRole tidak perlu diubah). Sehingga menjadi seperti berikut

```
# developer-rolebinding.yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  creationTimestamp: null
  name: developer-binding
  namespace: dev
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: developer
subjects:
# contoh role binding pada User
# - apiGroup: rbac.authorization.k8s.io
#   kind: User
#   name: 08ad7cea-70e4-4aa6-9a96-e46e0afd0566 #objectId user, dikarenakan akun Guest

# contoh role binding pada Group
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: 3bb59bf9-d05d-45fb-92e7-d6f0f3a04372 #objectId user, dikarenakan akun Guest
---

# developer-clusterrolebinding.yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  creationTimestamp: null
  name: developer-binding
  namespace: dev
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: developer
subjects:
# contoh role binding pada User
# - apiGroup: rbac.authorization.k8s.io
#   kind: User
#   name: 08ad7cea-70e4-4aa6-9a96-e46e0afd0566 #objectId user, dikarenakan akun Guest

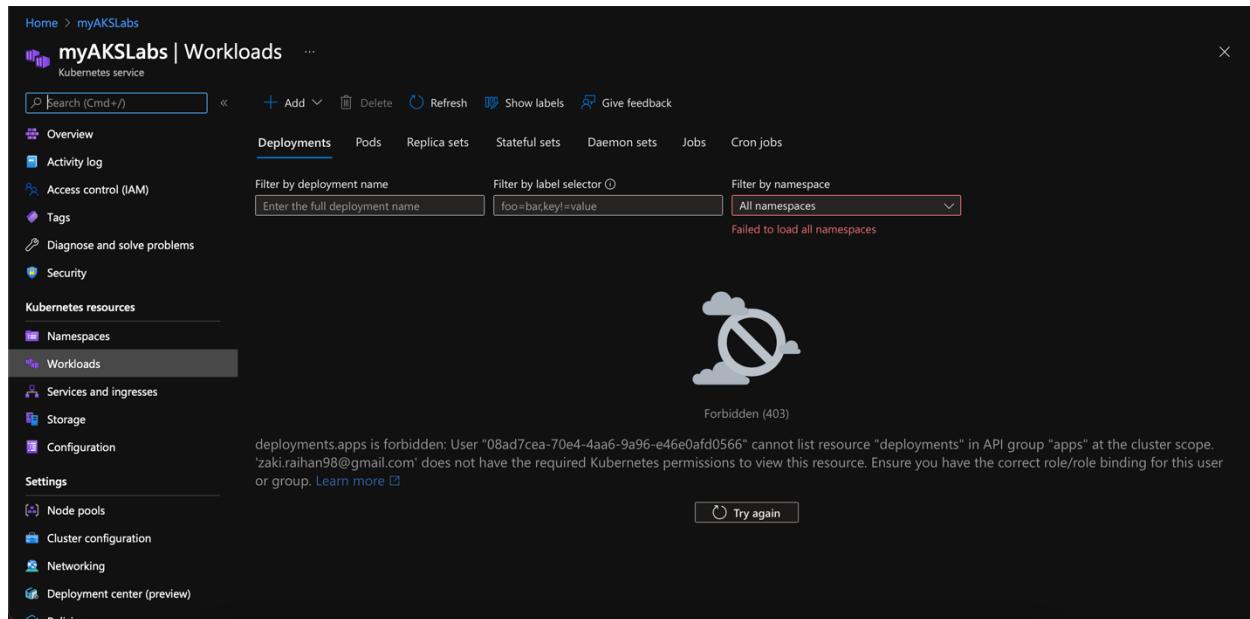
# contoh role binding pada Group
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: 3bb59bf9-d05d-45fb-92e7-d6f0f3a04372 #objectId user, dikarenakan akun Guest
```

Untuk melakukan maintenance, Kubernetes Admin hanya perlu mengubah file yaml dari RoleBinding dan ClusterRoleBinding jika ingin menambah group baru ke dalam Binding atau menambahkan langsung User ke dalam Binding. Sedangkan jika ingin menambahkan akun ke dalam Group, Kubernetes Admin dapat menggunakan Azure Portal (Azure Active Directory) dan melakukan penambahan atau pengurangan akun di dalam Group tanpa perlu mengubah file yaml dari RoleBinding dan ClusterRoleBinding.

## 7. Notes

### 7.1. Akses Developer terhadap Kubernetes Cluster melalui Azure Portal

Agar Developer dapat mengakses *resource* melalui Azure Portal dengan baik, maka setidaknya User atau Group dari Akun Developer harus memiliki sebuah ClusterRole yang mengizinkan mereka untuk melakukan operasi "list" terhadap resource "namespace". Hal ini dikarenakan di dalam Azure Portal sendiri, pada saat User mengakses sebuah resource (misalkan Pod) maka Azure Portal juga perlu melakukan query untuk mendapatkan namespace yang ingin dituju, sehingga apa bila seseorang memiliki akses "list" terhadap resource "pod" pada namespace "dev", namun dia tidak memiliki akses terhadap list namespace, maka dia tidak akan bisa mengaksesnya lewat Azure Portal, seperti pada gambar di bawah ini.



Terlihat bahwa saat Azure Portal gagal melakukan query terhadap Namespace yang ada di dalam Cluster ("Failed to load all namespaces"), dikarenakan User tidak memiliki akses "list" terhadap resource "namespace".

Setelah User ditambahkan ClusterRole yang memiliki akses "list" terhadap resource "namespace", maka user bisa mengakses "list" pada resource "pod" lewat Azure Portal, seperti pada gambar di bawah ini.

Home > myAKSLabs

myAKSLabs | Workloads

Kubernetes service

Search (Cmd +/)

Add Delete Refresh Show labels Give feedback

Overview Activity log Access control (IAM) Tags Diagnose and solve problems Security

Namespaces Workloads Services and ingresses Storage Configuration

Deployments Pods Replica sets Stateful sets Daemon sets Jobs Cron jobs

Filter by deployment name Enter the full deployment name foo=barkey1=value

Filter by label selector Filter by namespace dev

Name	Namespace	Ready	Up-to-date	Available	Age ↓
nginx	dev	✓ 1/1	1	1	4 hours

Namun yang bisa di akses hanya pada namespace "dev". Pada namespace selain "dev", user tidak dapat mengaksesnya.

Home > myAKSLabs

myAKSLabs | Workloads

Kubernetes service

Search (Cmd +/)

Add Delete Refresh Show labels Give feedback

Overview Activity log Access control (IAM) Tags Diagnose and solve problems Security

Namespaces Workloads Services and ingresses Storage Configuration

Deployments Pods Replica sets Stateful sets Daemon sets Jobs Cron jobs

Filter by deployment name Enter the full deployment name foo=barkey1=value

Filter by label selector Filter by namespace kube-system

Forbidden (403)

deployments.apps is forbidden: User "08ad7cea-70e4-4aa6-9a96-e46e0af0566" cannot list resource "deployments" in API group "apps" in the namespace "kube-system". "zakiraihan98@gmail.com" does not have the required Kubernetes permissions to view this resource. Ensure you have the correct role/role binding for this user or group. [Learn more](#)

Try again

## 7.2. Akses file yaml yang ada pada Contoh

Untuk file yaml yang dijadikan contoh dapat diakses pada link berikut:

<https://github.com/zakiraihan/AKS-RBAC-example>