**Cluster service account, role, cluster role, role binding relationship.**

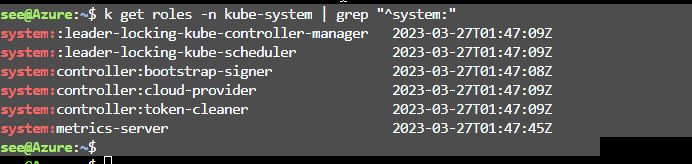
First, you should create an identity (Service account) for your workload.

Then, you have to define the permissions and include them into a Role.

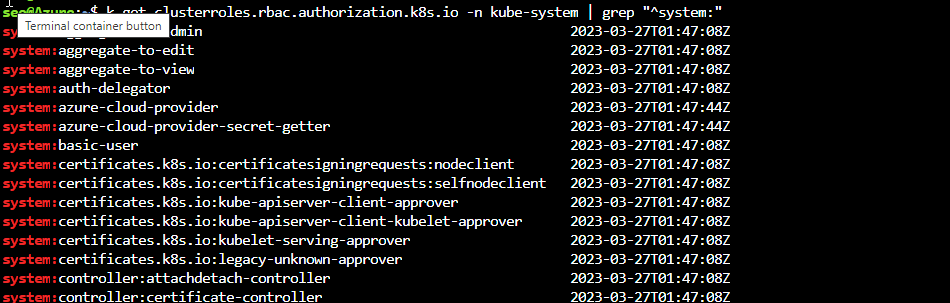
And finally, you want to link the identity (Service Account) to the permissions (Role) with a RoleBinding.

Next time the app issues a request to the Kubernetes API, and it will be granted access to the resources.

**Kubernetes ships with few roles and clusterRoles.**



**clusterRoles.**



Roles and RoleBindings are placed inside and grant access to a specific namespace, while ClusterRoles and ClusterRoleBindings do not belong to a namespace and grant access across the entire cluster.

it is possible to mix these two types of resources.

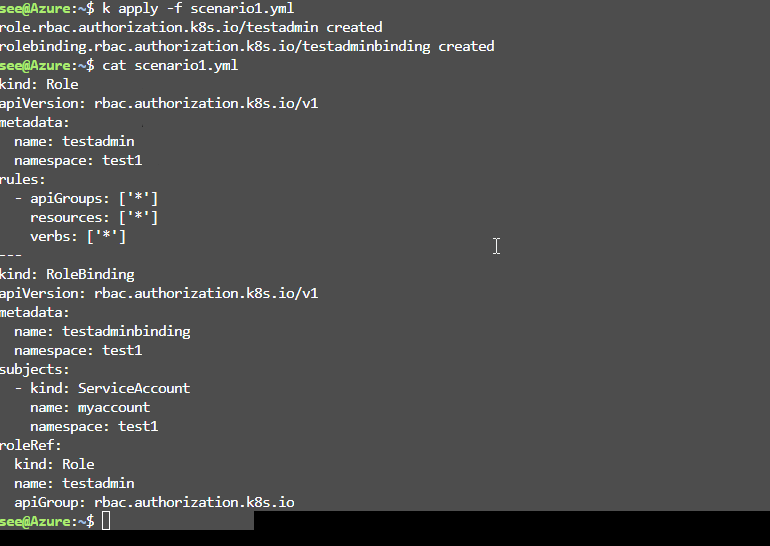
For example, what happens when a RoleBinding links an account to a ClusterRole?

Let starts some test by creating some namespaces:

**First scenario:**

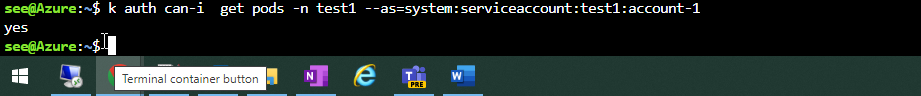
Role and RoleBinding in the same namespace.

Let's start with creating a Role and a RoleBinding to grant the Service Account access to the test1 namespace:



Service account, role, rolebinding are in the namespace test1

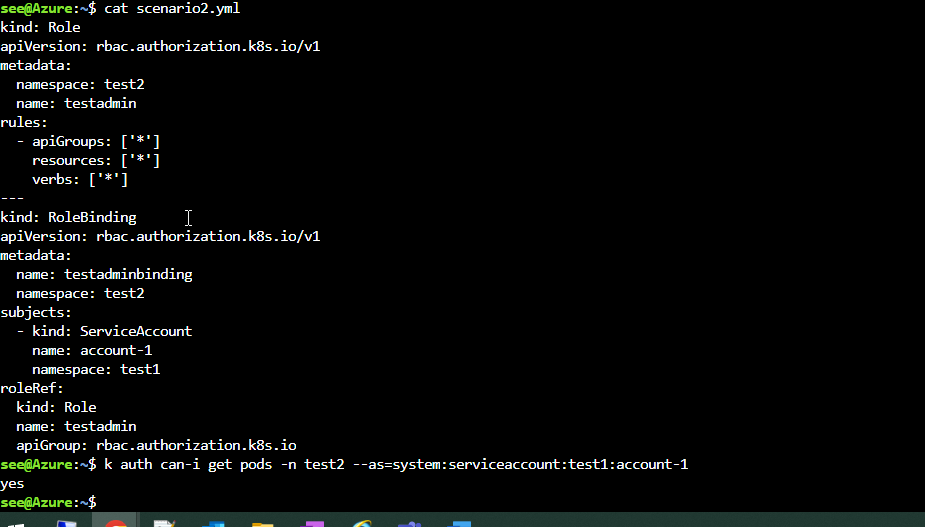
Test the access, authorization to the resources in same namespace.



Second scenario:

Role and rolebinding in different namespace.

Service account (account-1) in namespace test1 but will bind to role and rolebinding in namespace test2. Let's try.



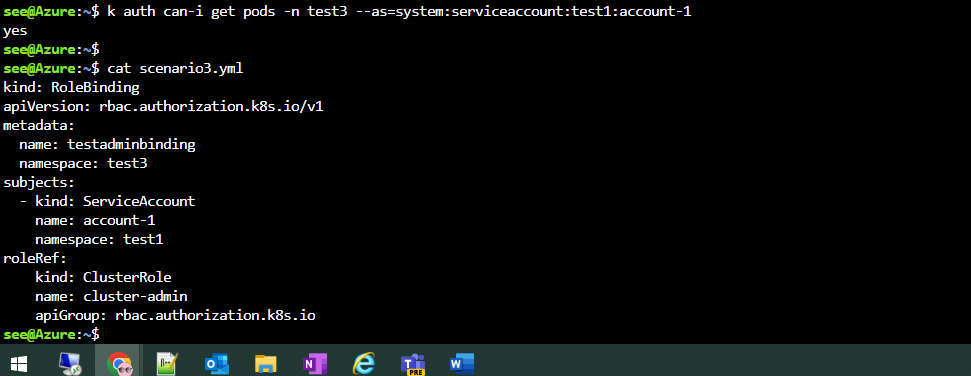
Account-1 service account able to access resources in another namespace (test2).

**RoleBinding can only reference a Role in the same namespace.**

**Scenario 3: ClusterRole with a RoleBinding**

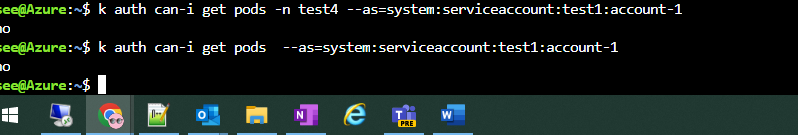
**------------------**

Create a RoleBinding in namespace test3 and link the Service Account (account-1) to the ClusterRole cluster-admin:

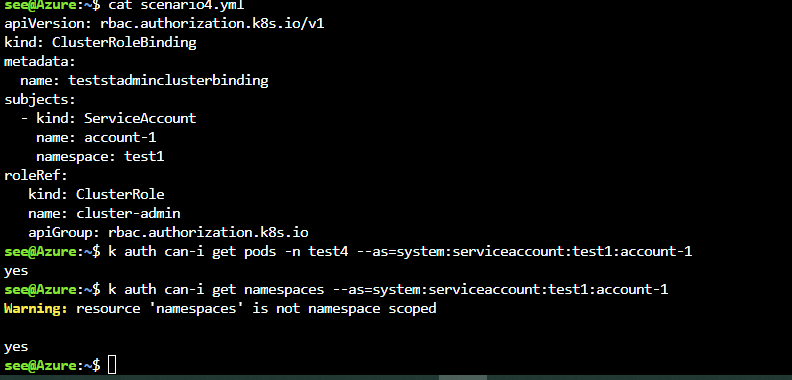


Let's test if the Service Account located in account-1  has access to the resources in test3:

But account-1 in namespace1 does not have access to resources in namespace4 and the default namespace.



neither the ClusterRole nor the ClusterRole binding defined any namespaces, the Service Account now has access to everything:



* Roles and RoleBindings must exist in the same namespace.
* RoleBindings can exist in separate namespaces to Service Accounts.
* RoleBindings can link ClusterRoles, but they only grant access to the namespace of the RoleBinding.
* ClusterRoleBindings link accounts to ClusterRoles and grant access across all resources.
* ClusterRoleBindings can not reference Roles.

Perhaps the most interesting implication here is that a ClusterRole can define common permissions expressed in a single namespace when referenced by a RoleBinding.

This removes the need to have duplicated roles in many namespaces.